1. Introduction

The objective of this report is to represent the implementation of integrated Citrus Expert system (CITEX4) including four sub expert systems: assessment, plant care, diagnosis, and treatment in addition to other two other sub systems: database and multimedia. This implementation is based on the integrated design report (TR/CLAES/211/2001.4). This system is implemented using KROL99 tools that support building the concepts, rules, inference, and task code.

The following eight sections represent the implementation code of the common knowledge base, assessment expert system, plant care expert system, diagnosis expert system, treatment expert system, database system, multimedia system, and user interface system.

2. Common Knowledge Base

File name: c_concept.pl

```
:-ensure_loaded('$KROL/lib/inferenc').
farm_data :: {
        concept_description('') &
        attributes([
        sid([]),
        gid([]),
        did([]),
        fid([]),
        month([])
        ]) &
        type(sid/1, integer) &
        ul(sid/1, 10) &
        ll(sid/1, 1) &
        prompt(sid/1, 'Enter the sector ID', []) &
        necessary(sid/1) &
        type(gid/1, integer) &
        ul(gid/1, 1000) &
        ll(gid/1, 1) &
        prompt(gid/1, 'Enter the governorate ID', []) &
        necessary(gid/1) &
        type(did/1, integer) &
        ul(did/1, 1000) &
        ll(did/1, 1) &
        prompt(did/1, 'Enter the directorate ID', []) &
        necessary(did/1) &
        type(fid/1, integer) &
        ul(fid/1, 1000) &
        ll(fid/1, 1) &
        prompt(fid/1, 'Enter the farm ID', []) &
        necessary(fid/1) &
        type(month/1, integer) &
        ul(month/1, 12) &
        ll(month/1, 1) &
        prompt(month/1, 'Enter the month', []) &
        necessary(month/1) &
        super(domain_class)
}.
variety :: {
        concept_description('') &
        attributes([
        value([])
        ]) &
        type(value/1, nominal) &
        source_of_value(value/1,
[database(citex4ds,farm_table(_157883,_157885,_157887,_157889,_157891,_157893,_1578
95,_157897,_157899,_157901,_157903,_157905,_157907,_157909,_157911,Vvariety_name
,_157915),Vvariety_name)]) &
        legal(value/1, [
                lime,
                navel,
                succar,
                valencia
        ]) &
        necessary(value/1) &
        super(domain_class)
}.
plantation :: {
        concept_description('') &
        attributes([
        existence([]),
        current_date([]),
        plantation_date([]),
        appearance([]),
        type([])
        ]) &
        type(existence/1, nominal) &
        prompt(existence/1, '', []) &
        legal(existence/1, [
                yes,
                no
        ]) &
        necessary(existence/1) &
        type(current_date/1, date) &
        prompt(current_date/1, '', []) &
        necessary(current_date/1) &
        type(plantation_date/1, date) &
        source_of_value(plantation_date/1,
[database(citex4ds,farm_table(_9234,_9236,_9238,_9240,_9242,_9244,Vplantation_date,_92
48,_9250,_9252,_9254,_9256,_9258,_9260,_9262,_9264,_9266),Vplantation_date)]) &
        type(appearance/1, nominal) &
        prompt(appearance/1, 'ÇáÙÇåÑå', []) &
        legal(appearance/1, [
        '\' æÌæÏ ÇÎÊáÇÝ ßÈíÑ Ýì ãäÓæÈ ÇáÊÑÈÉ\'',
        '\'ÇÓÊÎÏÇã ÇáÑì ÈÇáÊäÞíØ\'',
        '\'ÇáÇÔÌÇÑ Ýì ãÑÍáÉ ÇáÊÒåíÑ æÇáÚÞÏ\'',
        '\'Êã ÞØÝ ÇáËãÇÑ\'',
        '\'ÖÚÝ ÇäÊÇÌ ÇáÕäÝ\'',
        '\'æÌæÏ ãÇÁ Ñì íßÝì áÍÇÌÉ ÇáãÄÞÊÇÊ æÔÊáÇÊ ÇáãæÇÍá ãÚÇ\''   ]) &
    type(type/1, nominal) &
    prompt(type/1, 'äæÚ ÇáÈÓÊÇä', []) &
    legal(type/1, [
        '\'ÇÞÇãÉ ÈÓÊÇä ÍÏíË\'',
        '\'ÑÚÇíÉ ÇáÈÓÊÇä\''
    ]) &
    super(domain_class)
}.
soil :: {
    concept_description('') &
    attributes([
    s_status([]),
    ca_carbonate([]),
    ec([]),
    esp([]),
    texture([]),
    water_table_level([]),
    ph([])
    ]) &
    type(s_status/1, nominal) &
    multiple(s_status/1) &
    source_of_value(s_status/1, [[derived(s_p_p_v_det_s)]]) &
    legal(s_status/1, [
        alkaline,
        caliche,
        saline,
        suitable_soil,
        texture_def,
        unsuitable_soil,
        water_table_def,
        calcareous]) &
    type(ca_carbonate/1, real) &
    source_of_value(ca_carbonate/1,
[database(citex4ds,soil_assessment_table(_13952,_13954,_13956,_13958,_13960,_13962,_1
3964,_13966,Vca_carbonate,_13970,_13972,_13974),Vca_carbonate)]) &
    ul(ca_carbonate/1, 100) &
    ll(ca_carbonate/1, 0.1) &
    type(ec/1, real) &
    source_of_value(ec/1,
[database(citex4ds,soil_table(_15309,_15311,_15313,_15315,_15317,Vec,_15321,_15323,_
15325),Vec)]) &
    ul(ec/1, 10) &
    ll(ec/1, 0.1) &
    type(esp/1, real) &
```

```
        source_of_value(esp/1,
[database(citex4ds,soil_assessment_table(_16667,_16669,_16671,_16673,_16675,_16677,_1
6679,_16681,_16683,_16685,_16687,Vesp),Vesp)]) &
        ul(esp/1, 100) &
        ll(esp/1, 1) &
        type(texture/1, nominal) &
        source_of_value(texture/1,
[database(citex4ds,soil_table(_19327,_19329,_19331,Vtexture,_19335,_19337,_19339,_193
41,_19343),Vtexture)]) &
        legal(texture/1, [
                clay,
                clay_loam,
                coarse_sand,
                gravely,
                heavy_clay,
                loam,
                sand,
                sandy_clay_loam,
                sandy_loam,
                sily_clay,
                sily_clay_loam,
                sily_loam]) &
        type(water_table_level/1, real) &
        source_of_value(water_table_level/1,
[database(citex4ds,soil_table(_21686,_21688,_21690,_21692,Vwater_table_level,_21696,_2
1698,_21700,_21702),Vwater_table_level)]) &
        ul(water_table_level/1, 10) &
        ll(water_table_level/1, 0.1) &
        type(ph/1, real) &
        source_of_value(ph/1,
[database(citex4ds,soil_table(_56836,_56838,_56840,_56842,_56844,_56846,Vph,_56850,_
56852),Vph)]) &
        ul(ph/1, 14.0) &
        ll(ph/1, 0.1) &
        super(domain_class)
}.
water :: {
        concept_description('') &
        attributes([
        w_status([]),
        boron([]),
        eciw([]),
        rsc([]),
        sar([])]) &
        type(w_status/1, nominal) &
        multiple(w_status/1) &
        source_of_value(w_status/1, [[derived(w_p_p_det_w)]]) &
        legal(w_status/1, [
                alkaline,
                boron_def,
			suitable_water,
			unsuitable_water,
			saline1,
			saline2
		]) &
		type(boron/1, real) &
		source_of_value(boron/1,
[database(citex4ds,soil_assessment_table(_26226,_26228,_26230,Vboron,_26234,_26236,_2
6238,_26240,_26242,_26244,_26246,_26248),Vboron)]) &
		ul(boron/1, 5) &
		ll(boron/1, 0.01) &
		type(eciw/1, real) &
		source_of_value(eciw/1,
[database(citex4ds,water_table(_27514,_27516,_27518,Veciw),Veciw)]) &
		ul(eciw/1, 5) &
		ll(eciw/1, 0.01) &
		type(rsc/1, real) &
		source_of_value(rsc/1,
[database(citex4ds,soil_assessment_table(_28856,_28858,_28860,_28862,_28864,Vrsc,_288
68,_28870,_28872,_28874,_28876,_28878),Vrsc)]) &
		ul(rsc/1, 5) &
		ll(rsc/1, 0.01) &
		type(sar/1, real) &
		source_of_value(sar/1,
[database(citex4ds,soil_assessment_table(_30228,_30230,_30232,_30234,_30236,_30238,Vs
ar,_30242,_30244,_30246,_30248,_30250),Vsar)]) &
		ul(sar/1, 100) &
		ll(sar/1, 0.1) &
		super(domain_class)
}.
leaves :: {
		concept_description('') &
		attributes([
		l_color([]),
		l_shape([]),
		l_status([]),
		l_type([]),
		l_c_position([])]) &
		type(l_color/1, nominal) &
		multiple(l_color/1) &
		prompt(l_color/1, 'What is the leaves shape ?', []) &
		legal(l_color/1, [
			green,
			green_network,
			light_green,
			dark_green,
			geen_to_red,
			yellow,
			brown,
			black,
                purple,
                bronze]) &
        type(l_shape/1, nominal) &
        multiple(l_shape/1) &
        prompt(l_shape/1, 'What is the leaves shape?', []) &
        legal(l_shape/1, [
                normal,
                curled,
                webbed,
                honey_dew,
                cup_shape,
                unsimilar_blade_halves,
                zigzag_tunnels]) &
        type(l_status/1, nominal) &
        multiple(l_status/1) &
        prompt(l_status/1, 'What is the leaves status?', []) &
        legal(l_status/1, [
                normal,
                drop,
                insect_persent,
                small,
                wilted]) &
        type(l_type/1, nominal) &
        multiple(l_type/1) &
        prompt(l_type/1, 'What is the age of the infected leaves?', []) &
        legal(l_type/1, [new_leaves,  old_leaves]) &
        type(l_c_position/1, nominal) &
        multiple(l_c_position/1) &
        prompt(l_c_position/1, 'Where is the position of the infestation on the leaves?', [])&
        legal(l_c_position/1, [
                entire_leaf,
                inverted_v,
                'lower surface',
                'upper surface',
                'outer edge',
                'leaf base',
                'leaf margin',
                veins,
                'between veins',
                'main veins',
                'leaf tip'          ]) &
        super(domain_class)
}.
leaf_spots :: {
        concept_description('') &
        attributes([
        existence([]),
        l_s_color([]),
        l_s_shap([]),
        l_s_position([])
```

```
		]) &
		type(existence/1, nominal) &
		prompt(existence/1, 'Are they any sopts on leaves ?', []) &
		legal(existence/1, [
			yes,
			no
		]) &
		type(l_s_color/1, nominal) &
		multiple(l_s_color/1) &
		prompt(l_s_color/1, 'What is color of the spots on leaves?', []) &
		legal(l_s_color/1, [
			yellow,
			brown,
			dusty,
			silver,
			rust,
			black]) &
		type(l_s_shap/1, nominal) &
		multiple(l_s_shap/1) &
		prompt(l_s_shap/1, 'What is the shape of the spots on leaves ?', []) &
		legal(l_s_shap/1, [
			raised,
			sunken,
			necrotic,
			'zigzag tunnels',
			'concentric zones']) &
		type(l_s_position/1, nominal) &
		multiple(l_s_position/1) &
		prompt(l_s_position/1, 'What is the position of the spots on leaves?', []) &
		legal(l_s_position/1, [
			scattered,
			'upper surface',
			'lower surface',
			'between veins',
			'between veins of lower surface',
			'midrib upper surface']) &
		super(domain_class)
	}.
	fruits :: {
		concept_description('') &
		attributes([
		f_c_position([]),
		f_color([]),
		f_shape([]),
		f_r_status([])
		]) &
		type(f_c_position/1, nominal) &
		multiple(f_c_position/1) &
		prompt(f_c_position/1,'What is the position of the infestation on the fruit?', []) &
		legal(f_c_position/1, [
                'entire fruit',
                'styler end'       ]) &
        necessary(f_c_position/1) &
        type(f_color/1, nominal) &
        multiple(f_color/1) &
        prompt(f_color/1, 'What is the fruits color?', []) &
        legal(f_color/1, [
                black,
                green,
                'green styler end',
                normal,
                purple,
                rust,
                yellow,
                'yellow styler end',
                silver]) &
        necessary(f_color/1) &
        type(f_shape/1, nominal) &
        multiple(f_shape/1) &
        prompt(f_shape/1, 'What is the fruit shape?', []) &
        legal(f_shape/1, [
                asymatric,
                cracks,
                malformed,
                normal,
                small,
                soft,
                coarse]) &
        necessary(f_shape/1) &
        type(f_r_status/1, nominal) &
        multiple(f_r_status/1) &
        prompt(f_r_status/1, 'What is the fruits status?', []) &
        legal(f_r_status/1, [
                creasing,
                irregular,
                leathery,
                normal,
                reduced,
                rough,
                'rough and thickened',
                thickened,
                thin,
                drop]) &
        necessary(f_r_status/1) &
        super(domain_class)
}.
fruit_spots :: {
        concept_description('') &
        attributes([
        existence([]),
        f_s_color([]),
        f_s_position([]),
        f_s_shape([])
        ]) &
        type(existence/1, nominal) &
        multiple(existence/1) &
        prompt(existence/1, 'Are there spots on fruit? ', []) &
        legal(existence/1, [
                yes,
                no
        ]) &
        necessary(existence/1) &
        type(f_s_color/1, nominal) &
        multiple(f_s_color/1) &
        prompt(f_s_color/1, 'What is the color of the spots on the fruit?', []) &
        legal(f_s_color/1, [
                green,
                yellow,
                brown,
                red,
                sliver,
                bronze,
                'scabby patches']) &
        necessary(f_s_color/1) &
        type(f_s_position/1, nominal) &
        multiple(f_s_position/1) &
        prompt(f_s_position/1, 'What is the position of the spots on the fruit?', []) &
        legal(f_s_position/1, [
                scattered,
                'any position',
                rind,
                'stiller and stem ends',
                'fruits facing the sun']) &
        necessary(f_s_position/1) &
        type(f_s_shape/1, nominal) &
        multiple(f_s_shape/1) &
        prompt(f_s_shape/1, 'What is the shape of the spots on the fruit?', []) &
        legal(f_s_shape/1, [
                circular,
                irregular,
                raised,
                coarse,
                'large and circular',
                'gum pocket',
                'zigzag tunnels']) &
        necessary(f_s_shape/1) &
        super(domain_class)
}.
flowers :: {
        concept_description('') &
        attributes([
        fl_color([]),
        fl_status([]),
        f_l_shape([])]) &
        type(fl_color/1, nominal) &
        multiple(fl_color/1) &
        prompt(fl_color/1, 'What is the flowers color?', []) &
        legal(fl_color/1, [
                normal,
                brown,
                yellow]) &
        necessary(fl_color/1) &
        type(fl_status/1, nominal) &
        multiple(fl_status/1) &
        prompt(fl_status/1, 'What is the flowers color?', []) &
        legal(fl_status/1, [
                nromal,
                drop]) &
        necessary(fl_status/1) &
        type(f_l_shape/1, nominal) &
        multiple(f_l_shape/1) &
        prompt(f_l_shape/1, 'What is the flowers shape?', []) &
        legal(f_l_shape/1, [
                normal,
                aggregated,
                eaten]) &
        necessary(f_l_shape/1) &
        super(domain_class)
}.
branches :: {
        concept_description('') &
        attributes([
        b_type([]),
        b_status([]),
        b_color([])
        ]) &
        type(b_type/1, nominal) &
        multiple(b_type/1) &
        prompt(b_type/1, 'What is the age of the infected branches ?', []) &
        legal(b_type/1, [
                flushes,
                'old growth']) &
        necessary(b_type/1) &
        type(status/1, nominal) &
        multiple(status/1) &
        prompt(b_status/1, 'What is the branches status ?', []) &
        legal(b_status/1, [
                decline,
                'die back',
                dry,
                flattened,
                'insect present',
                normal,
                stunted,
                thickened,
                'gray fellvet']) &
        necessary(b_status/1) &
        type(b_color/1, nominal) &
        multiple(b_color/1) &
        prompt(b_color/1, 'What is the branches color?', []) &
        legal(b_color/1, [
                black,
                brown,
                normal,
                pale,
                rust,
                'spotted yellowish']) &
        necessary(b_color/1) &
        super(domain_class)
}.
trunk :: {
        concept_description('') &
        attributes([
        t_shape([]),
        t_position([])]) &
        type(t_shape/1, nominal) &
        multiple(t_shape/1) &
        prompt(t_shape/1, 'What is the trunk shape ?', []) &
        legal(t_shape/1, [
                normal,
                'fungal growths',
                'lichen growths',
                'bark scaling',
                'gum spots',
                dwarfing]) &
        necessary(t_shape/1) &
        type(t_position/1, nominal) &
        multiple(t_position/1) &
        prompt(t_position/1, 'What is the trunk position ?', []) &
        legal(t_position/1, [
                'basal part',
                'feeder roots']) &
        necessary(t_position/1) &
        super(domain_class)
}.
buds :: {
        concept_description('') &
        attributes([
        u_color([]),
        u_shape([]),
        u_status([])]) &
        type(u_color/1, nominal) &
        multiple(u_color/1) &
        prompt(u_color/1, 'What is the buds color ?', []) &
        legal(u_color/1, [
                normal,
                brown]) &
        necessary(u_color/1) &
        type(u_shape/1, nominal) &
        multiple(u_shape/1) &
        prompt(u_shape/1, 'What is the buds shape ?', []) &
        legal(u_shape/1, [
                rosette,
                deformed]) &
        necessary(u_shape/1) &
        type(u_status/1, nominal) &
        multiple(u_status/1) &
        prompt(u_status/1, 'What is the buds status?', []) &
        legal(u_status/1, [
                normal,
                abnormal]) &
        necessary(u_status/1) &
        super(domain_class)
}.
roots :: {
        concept_description('') &
        attributes([
        r_color([]),
        r_status([]),
        r_type([])]) &
        type(r_color/1, nominal) &
        multiple(r_color/1) &
        prompt(r_color/1, 'What is the root color ?', []) &
        legal(r_color/1, [
                normal,
                brown,
                black]) &
        necessary(r_color/1) &
        type(r_status/1, nominal) &
        multiple(r_status/1) &
        prompt(r_status/1, 'What is the root status ?', []) &
        legal(r_status/1, [
                normal,
                'fungal growth',
                sloughing,
                necrotic,
                adhesive]) &
        necessary(r_status/1) &
        type(r_type/1, nominal) &
        multiple(r_type/1) &
        prompt(r_type/1, 'What is the type of the infected roots ?', []) &
        legal(r_type/1, [
                'main roots',
                'feeder roots']) &
        necessary(r_type/1) &
        super(domain_class)
}.
twigs :: {
        concept_description('') &
        attributes([
        tw_color([]),
        tw_shape([]),
        tw_status([])]) &
        type(tw_color/1, nominal) &
        multiple(tw_color/1) &
        prompt(tw_color/1, 'What is the twigs color ?', []) &
        legal(tw_color/1, [
                brown,
                rust]) &
        necessary(tw_color/1) &
        type(tw_shape/1, nominal) &
        multiple(tw_shape/1) &
        prompt(tw_shape/1, 'What is the twigs shape?', []) &
        legal(tw_shape/1, [
                eaten]) &
        necessary(tw_shape/1) &
        type(tw_status/1, nominal) &
        multiple(tw_status/1) &
        prompt(tw_status/1, 'What is the twigs status ?', []) &
        legal(tw_status/1, [
                dieback]) &
        necessary(tw_status/1) &
        super(domain_class)
}.
plant :: {
        concept_description('') &
        attributes([
        current_date([]),
        age([]),
        season([]),
        current_week([]),
        current_month([]),
        yield([]),
        previous_yield_production([]),
        actual_yield([])]) &
        type(current_date/1, date) &
        type(age/1, real) &
        ul(age/1, 50) &
        ll(age/1, 0) &
        prompt(age/1, '', []) &
        type(yield/1, real) &
        type(season/1, nominal) &
        source_of_value(season/1, [table(plant_determine_plant)]) &
        legal(season/1, [
                spring,
                summer,
                autumn,
                winter]) &
        type(current_week/1, integer) &
        source_of_value(current_week/1, [derived(treated_by)]) &
        ul(current_week/1, 53) &
        ll(current_week/1, 1) &
        type(current_month/1, integer) &
        ul(current_month/1, 12) &
        ll(current_month/1, 1) &
        prompt(current_month/1, 'What is the current month?', []) &
        necessary(current_month/1) &
        type(yield/1, real) &
        source_of_value(yield/1, [derived(p_v_det_p)]) &
        ul(yield/1, 50) &
        ll(yield/1, 0) &
        type(previous_yield_production/1, real) &
        ul(previous_yield_production/1, 20) &
        ll(previous_yield_production/1, 0) &
        prompt(previous_yield_production/1, 'Enter the avaredge previous yield per Faddan
during the last three years', []) &
        necessary(previous_yield_production/1) &
        type(actual_yield/1, real) &
        source_of_value(actual_yield/1, [function(actual_yield)]) &
        ul(actual_yield/1, 20) &
        ll(actual_yield/1, 0) &
        super(domain_class)
}.
operation :: {
        concept_description('') &
        attributes([
        material_qty([]),
        unit([]),
        material_name([]),
        method([]),
        material_gr1([]),
        material_gr2([]),
        material_gr3([]),
        material_gr4([]),
        material_gr5([]),
        material_gr6([]),
        material_gr7([]),
        material_gr8([]),
        material_gr9([]),
        material_gr10([]),
        material_gr11([]),
        material_gr12([])]) &
        type(material_qty/1, real) &
        source_of_value(material_qty/1, [derived(treat_op_determine_treat_op)]) &
        ul(material_qty/1, 1000) &
        ll(material_qty/1, 0) &
        target(material_qty/1, '') &
        type(unit/1, nominal) &
        source_of_value(unit/1, [derived(treat_op_determine_treat_op)]) &
        legal(unit/1, [
                'L/100 l water',
                'gm/1 l water',
                'gm/100 l water',
                'gm/tree',
                'kg CuSo4 + 2 Kg CaO + 10 L water',
                'ml + 25 ml/100 l water',
                'ml/100 l water',
                'Kg Cu So4 + 1.5 CaO/100 l water',
                'ml + 150 ml/100 l water',
                'ml + 250 ml/100 L water',
                'kg/feddan',
                'L/feddan',
                'ml + L/100 l water',
                'ml + 500 ml/100 l water',
                'kg/100 l water',
                'as below']) &
        type(material_name/1, nominal) &
        multiple(material_name/1) &
        source_of_value(material_name/1, [[derived(treated_by)]]) &
        legal(material_name/1, [
                'K.Z. 95%',
                'Kimisol 95%',
                'actellic 50%',
                'agro oil 80%',
                aikaten,
                'ammonium nitrate',
                'anthio 33%',
                'bolum oil 80%',
                'bordeaux past',
                'calcium chloride',
                'calcium nitrate',
                'caprimex 98%',
                'copox 50%',
                copper_oxychloride,
                'cuprus K.Z 50%',
                'focal oil 82%',
                'furidan 10%',
                'halomac 65%',
                'libacid 50% + bominal',
                magnesium_sulfate,
                  'malathion 57%',
                  'malthion 57% + policure',
                  'misrona oil 80%',
                  'micro element mixture',
                  'neron 50%',
                  none,
                  'ortis 5% sc + kz oil',
                  'pory coper 50%',
                  potassium_nitrate,
                  potassium_permenganat,
                  potassium_sulfate,
                  pride,
                  'pro coper 50%',
                  'ragbi 10%',
                  'royal oil 80%',
                  'super aside',
                  'super masrona 94%',
                  'super royal 95%',
                  'temic 15%',
                  topsin,
                  'triple phosphate',
                  urea,
                  vaydete,
                  'vertimec + K.Z oil 95%',
                  'vertimec + Kimisol oil 95%',
                  'vertimec + super masrona 94%',
                  'vertimec + super royal oil 95%',
                  'vertimec 1.8%',
                  'vertimec 1.8% + kz oil']) &
        target(material_name/1, '') &
        type(method/1, nominal) &
        source_of_value(method/1, [derived(treated_by)]) &
        legal(method/1, [
                  advice,
                  'chemical spray',
                  disinfection,
                  'foliage nutrition',
                  painting,
                  'soil treatment']) &
        target(method/1, '') &
        type(material_gr1/1, nominal) &
        prompt(material_gr1/1, 'Select available material', []) &
        legal(material_gr1/1, [
                  'K.Z. 95%',
                  'Kimisol 95%',
                  'super masrona 94%',
                  'super royal 95%']) &
        necessary(material_gr1/1) &
        type(material_gr2/1, nominal) &
        prompt(material_gr2/1, 'Select available material', []) &
```

```
legal(material_gr2/1, [
        'actellic 50%',
        aikaten,
        'anthio 33%',
        'super aside']) &
necessary(material_gr2/1) &
type(material_gr3/1, nominal) &
prompt(material_gr3/1, 'Select available material', []) &
legal(material_gr3/1, [
        'caprimex 98%',
        'copox 50%',
        copper_oxychloride,
        'cuprus K.Z 50%',
        'halomac 65',
        'pory coper 50%',
        'pro coper 50%']) &
necessary(material_gr3/1) &
type(material_gr4/1, nominal) &
prompt(material_gr4/1, 'Select available material', []) &
legal(material_gr4/1, [
        'agro oil 80%',
        'bolum oil 80%',
        'focal oil 82%',
        'misrona oil 80%',
        'royal oil 80%']) &
necessary(material_gr4/1) &
type(material_gr5/1, nominal) &
prompt(material_gr5/1, 'Select available material', []) &
legal(material_gr5/1, [
        'vertimec + K.Z oil 95%',
        'vertimec + Kimisol oil 95%',
        'vertimec + super masrona 94%',
        'vertimec + super royal oil 95%']) &
necessary(material_gr5/1) &
type(material_gr6/1, nominal) &
prompt(material_gr6/1, 'Select available material', []) &
legal(material_gr6/1, [
        'neron 50%',
        'ortis 5% sc + kz oil',
        'vertimec 1.8% + kz oil']) &
necessary(material_gr6/1) &
type(material_gr7/1, nominal) &
prompt(material_gr7/1, 'Select available material', []) &
legal(material_gr7/1, [
        'ortis 5% sc + kz oil',
        pride,
        'vertimec 1.8% + kz oil']) &
necessary(material_gr7/1) &
type(material_gr8/1, nominal) &
prompt(material_gr8/1, 'Select available material', []) &
```

```
        legal(material_gr8/1, [
                'furidan 10%',
                'ragbi 10%',
                'temic 15%']) &
        necessary(material_gr8/1) &
        type(material_gr9/1, nominal) &
        prompt(material_gr9/1, 'Select available material', []) &
        legal(material_gr9/1, [
                urea,
                'ammonium nitrate']) &
        necessary(material_gr9/1) &
        type(material_gr10/1, nominal) &
        prompt(material_gr10/1, 'Select available material', []) &
        legal(material_gr10/1, [
                potassium_nitrate,
                potassium_sulfate]) &
        necessary(material_gr10/1) &
        type(material_gr11/1, nominal) &
        prompt(material_gr11/1, 'Select available material', []) &
        legal(material_gr11/1, [
                'calcium chloride',
                'calcium nitrate']) &
        necessary(material_gr11/1) &
        type(material_gr12/1, nominal) &
        prompt(material_gr12/1, 'Select available material', []) &
        legal(material_gr12/1, [
                'ibacid 50% + bominal',
                'malthion 57% + policure']) &
        necessary(material_gr12/1) &
        super(domain_class)
}.
treat_op :: {
        concept_description('') &
        attributes([
        tool([]),
        application_time([]),
        advice([]),
        date([]),
        number([]),
        special_date([])
        ]) &
        type(tool/1, nominal) &
        multiple(tool/1) &
        prompt(tool/1, '', []) &
        legal(tool/1, [
                manual,
                'sprayer motor']) &
        type(application_time/1, nominal) &
        multiple(application_time/1) &
        source_of_value(application_time/1, [[derived(treat_op_determine_treat_op)]]) &
        legal(application_time/1, [
                'any suitable time',
                'early morning or afternoon']) &
        target(application_time/1, '') &
        type(advice/1, nominal) &
        multiple(advice/1) &
        source_of_value(advice/1, [[derived(enhanced_by)]]) &
        legal(advice/1, [
                'Also, avoid excess irrigation water near the trunk',
                'Application of acaricides is recommended at 20 % infestation, in general.
Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application. Spraying should be as a mist, tacking
umbrella shape at lower pressure and as possible over the entire tree',
                'Avoid excess of nitrogen fertilizers and organic manure near the trunk. Also,
avoid excess irrigation water near the trunk.',
                'Collect infected fruits and bury it. Perform the suitable agriculture practices',
                'Collect infested fruits and bury it.',
                'Control the insects that produce the honey dew',
                'Cultivate plant tarps for scarab like faba-beans, turnip and cauliflower',
                'Good caring the diseased trees; i.e. better agriculture practices and
fertilization to extend the productive life of tree when yield becomes not economic, the
diseased trees must be replaced. Use certified transplants',
                'Improve the agriculture practices',
                'Improve the growth of trees to protect the fruits from direct sun light',
                'Increase quantity of fertilizer application by 25% and incrementally increased
up to 50% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system',
                'Increase quantity of fertilizer application by 50% and incrementally increased
up to 100% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system',
                'Infected young trees should be replaced by other healthy plants. Use certified
transplants',
                'It is important to check the soil salinity, and in case of high salinity the
leaching is recommended',
                'Lichens control includes good agricultural practices; i.e. pruning and avoid
excess irrigation water',
                'Manage the irrigation and increase the fertilization quantity of Potassium',
                'No foliage application during the flowering stage and fruit setting',
                'No foliage application during the fruits collecting period',
                'No significant response of trees to foliar application during winter. Therefore
treat your trees in the beginning of spring',
                'No treatment for this pest, such that it is not important economically',
                'No treatment for this phenomena where its economic importance is limited',
                'Picking up the insects twice a day',
                'Remove fungal growths and painting the wound by Bordeaux past then spray
the green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water',
                'Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application',
                'Spray the infested trees only',
                'Spray trees of entire farm',
                    'Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible',
                    'Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from downwards to up words and pointing to the core of tree',
                    'Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from upwards to downwards',
                    'Spraying two branches only in each tree and collects infested fruits and bury
it.',
                    'Spread watercolor traps at the rate 35 to 40 traps per feddan',
                    'Substitute the nitrogen quantity in the fertilization expert system
recommendation by its equivalence of calcium nitrate',
                    'The diseased trees must be replaced',
                    'The gum pocked must be removed with sharp knife, the wound and exposed
tissues must be disinfected with solution',
                    'The micro elements mixture is formulated, for every 100 lt water , as follow :
150 gm Iron Chelate (EDTA) + 30 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax',
                    'The micro elements mixture is formulated, for every 100 lt water, as follow :
30 gm Iron Chelate (EDTA) + 30 gm Zinc Chelate + 75 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax',
                    'The micro elements mixture is formulated, for every 100 lt water, as follow:
30 gm Iron Chelate (EDTA) + 150 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax',
                    'The pressure of spraying motor must not exceed 100 pound per square inch
without direct application',
                    'The treatment at this time is not recommended.',
                    'The treatment at this time is not recommended. Time of chemical control in
late of April, in case of infestation',
                    'The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation',
                    'The treatment at this time is not recommended. Time of chemical control in
late of May, in case of infestation',
                    'The trees must be completely washed',
                    'This operation used as shared treatment for aphids and citrus white fly',
                    'Use compost organic manure',
                    'Use fit spraying motor with good mixing. The trees must be completely
washed.',
                    'Use irrigation program to add leaching requirements',
                    'You must follow this operation by light irrigation to avoid application of fruit
bearing trees'
          ]) &
          target(advice/1, '') &
          type(date/1, date) &
          source_of_value(date/1, [derived(treated_by)]) &
          target(date/1, '') &
          type(number/1, integer) &
          source_of_value(number/1, [derived(treated_by)]) &
          ul(number/1, 50) &
          ll(number/1, 1) &
          target(number/1, '') &
```

```
        type(special_date/1, nominal) &
        source_of_value(special_date/1, [derived(treated_by)]) &
        legal(special_date/1, [
                'next 1/12',
                'next 1/2',
                'next 1/6',
                'next 22/6',
                'next 13/7',
                'next 22/2',
                'next summer',
                'next winter']) &
        super(operation)
}.
disorder :: {
        concept_description('') &
        attributes([
        suspected([]),
        confirmed([]),
        highly_confirmed([]),
        iron_def_sp([]),
        manganese_def_sp([]),
        zinc_def_sp([]),
        nitrogen_def_sp([]),
        salt_injury_sp([]),
        magnesium_def_sp([]),
        calcium_def_sp([]),
        potassium_def_sp([]),
        phosphorus_infestation([]),
        nitrogen_infestation([]),
        potassium_infestation([])]) &
        type(suspected/1, nominal) &
        multiple(suspected/1) &
        source_of_value(suspected/1, [[derived(caused_by_disoders)]]) &
        legal(suspected/1, [
                psorosis,
                impieetratura,
                stubborn,
                anthracnose,
                gummosis,
                sooty_mold,
                wilt_root_rot,
                black_root_rot,
                ganoderma_rot,
                brown_rot,
                alternaria_rot,
                armillaria_root_rot,
                alternaria_leaves_spot,
                gum_spots,
                sun_burn,
                fruit_cracking,
                fruit_creasing,
                lichens,
                rose_scarab,
                mediterranean_fruit_fly,
                citrus_white_fly,
                scales,
                aphids,
                citrus_flower_moth,
                mealy_bug,
                green_stink_bug,
                leafminer,
                rust_mite,
                bud_mite,
                brown_mite,
                flat_mite,
                citrus_nematude,
                nitrogen_def,
                phosphorus_def,
                potassium_def,
                magnesium_def,
                manganese_def,
                iron_def,
                calcium_def,
                zinc_def,
                salt_injury]) &
        type(confirmed/1, nominal) &
        multiple(confirmed/1) &
        source_of_value(confirmed/1, [[derived(confirm_disorders)]]) &
        legal(confirmed/1, [
                psorosis,
                impieetratura,
                stubborn,
                anthracnose,
                gummosis,
                sooty_mold,
                wilt_root_rot,
                black_root_rot,
                ganoderma_rot,
                brown_rot,
                alternaria_rot,
                armillaria_root_rot,
                alternaria_leaves_spot,
                gum_spots,
                sun_burn,
                fruit_cracking,
                fruit_creasing,
                lichens,
                rose_scarab,
                mediterranean_fruit_fly,
                citrus_white_fly,
                scales,
                aphids,
                citrus_flower_moth,
                mealy_bug,
                green_stink_bug,
                leafminer,
                rust_mite,
                bud_mite,
                brown_mite,
                flat_mite,
                citrus_nematude,
                nitrogen_def,
                phosphorus_def,
                potassium_def,
                magnesium_def,
                manganese_def,
                iron_def,
                calcium_def,
                zinc_def,
                salt_injury]) &
        type(highly_confirmed/1, nominal) &
        multiple(highly_confirmed/1) &
        source_of_value(highly_confirmed/1, [[derived(verify_disorders)]]) &
        legal(highly_confirmed/1, [
                psorosis,
                impieetratura,
                stubborn,
                anthracnose,
                gummosis,
                sooty_mold,
                wilt_root_rot,
                black_root_rot,
                ganoderma_rot,
                brown_rot,
                alternaria_rot,
                armillaria_root_rot,
                alternaria_leaves_spot,
                gum_spots,
                sun_burn,
                fruit_cracking,
                fruit_creasing,
                lichens,
                rose_scarab,
                mediterranean_fruit_fly,
                citrus_white_fly,
                scales,
                aphids,
                citrus_flower_moth,
                mealy_bug,
                green_stink_bug,
                leafminer,
                rust_mite,
                bud_mite,
                brown_mite,
                flat_mite,
                citrus_nematude,
                nitrogen_def,
                phosphorus_def,
                potassium_def,
                magnesium_def,
                manganese_def,
                iron_def,
                calcium_def,
                zinc_def,
                salt_injury]) &
        type(iron_def_sp/1, nominal) &
        source_of_value(iron_def_sp/1, [user]) &
        prompt(iron_def_sp/1, 'What is the spread range of the iron defection infestation ?',
[]) &
        legal(iron_def_sp/1, [
                'most trees']) &
        necessary(iron_def_sp/1) &
        type(manganese_def_sp/1, nominal) &
        source_of_value(manganese_def_sp/1, [user]) &
        prompt(manganese_def_sp/1, 'What is the spread range of the manganese defection
infestation ?', []) &
        legal(manganese_def_sp/1, [
                'most trees']) &
        necessary(manganese_def_sp/1) &
        type(zinc_def_sp/1, nominal) &
        source_of_value(zinc_def_sp/1, [user]) &
        prompt(zinc_def_sp/1, 'What is the spread range of the zinc defection infestation ?',
[]) &
        legal(zinc_def_sp/1, [
                'most trees']) &
        necessary(zinc_def_sp/1) &
        type(nitrogen_def_sp/1, nominal) &
        source_of_value(nitrogen_def_sp/1, [user]) &
        prompt(nitrogen_def_sp/1, 'What is the spread range of the nitrogen defection
infestation ?', []) &
        legal(nitrogen_def_sp/1, [
                'most trees']) &
        necessary(nitrogen_def_sp/1) &
        type(salt_injury_sp/1, nominal) &
        source_of_value(salt_injury_sp/1, [user]) &
        prompt(salt_injury_sp/1, 'What is the spread range of the salt_injury defection
infestation ?', []) &
        legal(salt_injury_sp/1, [
                'most trees']) &
        necessary(salt_injury_sp/1) &
```

```
	type(magnesium_def_sp/1, nominal) &
	source_of_value(magnesium_def_sp/1, [user]) &
	prompt(magnesium_def_sp/1, 'What is the spread range of the magnesium defection
infestation ?', []) &
	legal(magnesium_def_sp/1, [
		'most trees']) &
	necessary(magnesium_def_sp/1) &
	type(calcium_def_sp/1, nominal) &
	source_of_value(calcium_def_sp/1, [user]) &
	prompt(calcium_def_sp/1, 'What is the spread range of the calcium defection
infestation ?', []) &
	legal(calcium_def_sp/1, [
		'most trees']) &
	necessary(calcium_def_sp/1) &
	type(potassium_def_sp/1, nominal) &
	source_of_value(potassium_def_sp/1, [user]) &
	prompt(potassium_def_sp/1, 'What is the spread range of the potassium defection
infestation ?', []) &
	legal(potassium_def_sp/1, [
		'most trees']) &
	necessary(potassium_def_sp/1) &
	type(phosphorus_infestation/1, nominal) &
	source_of_value(phosphorus_infestation/1, [user]) &
	prompt(phosphorus_infestation/1, 'What is the Value of Phosphours Infestation? ', [])
&
	legal(phosphorus_infestation/1, [
		low,
		'very low']) &
	necessary(phosphorus_infestation/1) &
	type(nitrogen_infestation/1, nominal) &
	source_of_value(nitrogen_infestation/1, [user]) &
	prompt(nitrogen_infestation/1, 'What is the Value of Nitrogen Infestation? ', []) &
	legal(nitrogen_infestation/1, [
		low,
		'very low']) &
	type(potassium_infestation/1, nominal) &
	source_of_value(potassium_infestation/1, [user]) &
	prompt(potassium_infestation/1, 'What is the Value of Potassium Infestation? ', []) &
	legal(potassium_infestation/1, [
		low,
		'very low']) &
	necessary(potassium_infestation/1) &
	super(treat_op)
}.
insects :: {
	concept_description('') &
	attributes([
	i_color([]),
	i_status([])]) &
	type(i_color/1, nominal) &
```

```
        multiple(i_color/1) &
        prompt(i_color/1, 'What is the insects color ?', []) &
        legal(i_color/1, [
                green,
                black,
                white,
                red,
                purple]) &
        necessary(i_color/1) &
        type(i_status/1, nominal) &
        multiple(i_status/1) &
        prompt(i_status/1, 'What is the insects status ?', []) &
        legal(i_status/1, [
                stationary,
                flying,
                stucked,
                aggregated]) &
        necessary(i_status/1) &
        super(domain_class)
}.
insect :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
disease :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
lichens :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
mites :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
nematode :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
nutrition_def :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
```

```
virus :: {
        concept_description('') &
        attributes([]) &
        super(disease)
}.
fungal :: {
        concept_description('') &
        attributes([]) &
        super(disease)
}.
environmental :: {
        concept_description('') &
        attributes([]) &
        super(disease)
}.
psorosis :: {
        concept_description('') &
        attributes([]) &
        super(virus)
}.
impieetratura :: {
        concept_description('') &
        attributes([]) &
        super(virus)
}.
stubborn :: {
        concept_description('') &
        attributes([]) &
        super(virus)
}.
anthracnose :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
gummosis :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
sooty_mold :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
ganoderma_rot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
```

```
alternaria_rot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
armillaria_root_rot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
wilt_root_rot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
alternaria_leaves_spot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
gum_spots :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
sun_burn :: {
        concept_description('') &
        attributes([]) &
        super(environmental)
}.
fruit_cracking :: {
        concept_description('') &
        attributes([]) &
        super(environmental)
}.
fruit_creasing :: {
        concept_description('') &
        attributes([]) &
        super(environmental)
}.
salt_injury :: {
        concept_description('') &
        attributes([]) &
        super(environmental)
}.
rose_scarab :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
```

```
mediterranean_fruit_fly :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
citrus_white_fly :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
scales :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
aphids :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
citrus_flower_moth :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
mealy_bug :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
green_stink_bug :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
leafminer :: {
        concept_description('') &
        attributes([]) &
        super(insect)
}.
rust_mite :: {
        concept_description('') &
        attributes([]) &
        super(mites)
}.
bud_mite :: {
        concept_description('') &
        attributes([]) &
        super(mites)
}.
```

```
brown_mite :: {
          concept_description('') &
          attributes([]) &
          super(mites)
}.
flat_mite :: {
          concept_description('') &
          attributes([]) &
          super(mites)
}.
citrus_nematude :: {
          concept_description('') &
          attributes([]) &
          super(nematode)
}.
nitrogen_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
phosphorus_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
potassium_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
magnesium_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
manganese_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
iron_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
calcium_def :: {
          concept_description('') &
          attributes([]) &
          super(nutrition_def)
}.
```

```
zinc_def :: {
        concept_description('') &
        attributes([]) &
        super(nutrition_def)
}.
navel :: {
        concept_description('') &
        attributes([]) &
        super(variety)
}.
succar :: {
        concept_description('') &
        attributes([]) &
        super(variety)
}.
valencia :: {
        concept_description('') &
        attributes([]) &
        super(variety)
}.
lime :: {
        concept_description('') &
        attributes([]) &
        super(variety)
}.
ganoderma_rot_op1 :: {
        concept_description('') &
        attributes([]) &
        super(ganoderma_rot)
}.
ganoderma_rot_op2 :: {
        concept_description('') &
        attributes([]) &
        super(ganoderma_rot)
}.
wilt_root_rot_op1 :: {
        concept_description('') &
        attributes([]) &
        super(wilt_root_rot)
}.
wilt_root_rot_op2 :: {
        concept_description('') &
        attributes([]) &
        super(wilt_root_rot)
}.
leafminer_op1 :: {
        concept_description('') &
        attributes([]) &
        super(leafminer)
}.
```

```
leafminer_op2 :: {
        concept_description('') &
        attributes([]) &
        super(leafminer)
}.
leafminer_op3 :: {
        concept_description('') &
        attributes([]) &
        super(leafminer)
}.
rust_mite_op1 :: {
        concept_description('') &
        attributes([]) &
        super(rust_mite)
}.
rust_mite_op2 :: {
        concept_description('') &
        attributes([]) &
        super(rust_mite)
}.
bud_mite_op1 :: {
        concept_description('') &
        attributes([]) &
        super(bud_mite)
}.
bud_mite_op2 :: {
        concept_description('') &
        attributes([]) &
        super(bud_mite)
}.
brown_mite_op1 :: {
        concept_description('') &
        attributes([]) &
        super(brown_mite)
}.
brown_mite_op2 :: {
        concept_description('') &
        attributes([]) &
        super(brown_mite)
}.
flat_mite_op1 :: {
        concept_description('') &
        attributes([]) &
        super(flat_mite)
}.
flat_mite_op2 :: {
        concept_description('') &
        attributes([]) &
        super(flat_mite)
}.
```

```
citrus_nematude_op1 :: {
          concept_description('') &
          attributes([]) &
          super(citrus_nematude)
}.
citrus_nematude_op2 :: {
          concept_description('') &
          attributes([]) &
          super(citrus_nematude)
}.
```

3. Assessment subsystem

3.1 Concepts properties

File name: ass_concept.pl

```
:-ensure_loaded('$KROL/lib/inferenc').
climate :: {
          concept_description('') &
          attributes([
          c_status([]),
          max_d_tc_ss([]),
          min_d_rh_ss([])]) &
          type(c_status/1, nominal) &
          multiple(c_status/1) &
          source_of_value(c_status/1, [[derived(c_det_c)]]) &
          legal(c_status/1, [
                    suitable_climate,
                    critical,
                    usuitable_climate,
                    usuitable_for_navel]) &
          type(max_d_tc_ss/1, real) &
          source_of_value(max_d_tc_ss/1,
[database(citex4ds,soil_assessment_table(_34322,_34324,_34326,_34328,_34330,_34332,_3
4334,_34336,_34338,Vmax_d_tc_ss,_34342,_34344),Vmax_d_tc_ss)]) &
          ul(max_d_tc_ss/1, 50) &
          ll(max_d_tc_ss/1, 0) &
          type(min_d_rh_ss/1, real) &
          source_of_value(min_d_rh_ss/1,
[database(citex4ds,soil_assessment_table(_35706,_35708,_35710,_35712,_35714,_35716,_3
5718,_35720,_35722,_35724,Vmin_d_rh_ss,_35728),Vmin_d_rh_ss)]) &
          ul(min_d_rh_ss/1, 100) &
          ll(min_d_rh_ss/1, 0) &
          super(domain_class)
}.
conclusion :: {
          concept_description('') &
          attributes([
          text_sp([]),
          text_w([]),
          text_wp([]),
        text_cp([]),
        text_sw([])]) &
        type(text_sp/1, nominal) &
        multiple(text_sp/1) &
        source_of_value(text_sp/1, [[derived(s_p_det_con)]]) &
        legal(text_sp/1, [
                'improving sandy soil texture before cultivation',
                'improving sandy soil texture for existence plantation ',
                'improving clay soil texture before cultivation',
                'improving clay soil texture for existence plantation ',
                'improve drainage system to treat the water table level before cultivation',
                'improve drainage system to treat the water table level for existence
plantation',
                'reduce soil salinity  by leaching before cultivation',
                'reduce soil salinity  by leaching for existence plantation',
                'reduce soil alkaline by adding Gypsum to replace Sodium with Calcium
before cultivation',
                'reduce soil alkaline by adding Gypsum to replace Sodium with Calcium for
existence plantation ',
                'improve calcareous soil  before cultivation',
                'improve calcareous soil  for existence plantation',
                'your location is not suitable for orange cultivation because you have some
soil defect',
                'this plantain is not economic for citrus production because soil properties are
not valid']) &
        type(text_w/1, nominal) &
        multiple(text_w/1) &
        source_of_value(text_w/1, [[derived(w_det_con)]]) &
        legal(text_w/1, [
                'your water source needs to be mixed with another good quality water source ',
                'your water source needs to be mixed with another good quality water source
and use irrigation program to determine irrigation qty',
                'the water is  alkaline and you need to add agricultural Gypsum to the soil',
                'use irrigation program to determine irrigation qty']) &
        type(text_wp/1, nominal) &
        multiple(text_wp/1) &
        source_of_value(text_wp/1, [[derived(w_p_det_con)]]) &
        legal(text_wp/1, [
                'your water quality is not suitable for orange or lime cultivation',
                'this plantain is not economic for citrus production  because water properties is
not valid']) &
        type(text_cp/1, nominal) &
        multiple(text_cp/1) &
        source_of_value(text_cp/1, [[derived(c_p_det_con)]]) &
        legal(text_cp/1, [
                'Your location climate is critical. You have to prepare your location by  wend
break two years before plantation and follow narrow plant spacing',
                'your location climate is critical and you need to establish wend break',
                'your climate is not suitable for orange or lime cultivation',
                    'Your location climate is critical and you need to establish wend break. You
have also to install system to raise air humidity like green cultivation system']) &
          type(text_sw/1, nominal) &
          multiple(text_sw/1) &
          source_of_value(text_sw/1, [[derived(s_w_c_v_p_det_con)]]) &
          legal(text_sw/1, [
                    'Navel Orange is not suitable to be cultivated in your location but other seedy
oranges may be suitable',
                    'you may replace the scion variety with other seedy oranges or graft some
main branches with compatible bolynaier (i.e. grip fruit and mandaline)']) &
          super(domain_class)
}.
```

3.2 Relations between expressions

File name: ass_rules.pl

```
:- ensure_loaded('$KROL/lib/rule_exp').
s_p_p_v_det_s :: {
r1([s_status(suitable_soil)in soil]) if
          (         soil :: get_value(texture(clay_loam))
          ;         soil :: get_value(texture(loam))
          ;         soil :: get_value(texture(sandy_clay_loam))
          ;         soil :: get_value(texture(sandy_loam))
          ;         soil :: get_value(texture(sily_clay))
          ;         soil :: get_value(texture(sily_clay_loam))
          ;         soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_3018)),
          :(_3018>=1.2),
          soil :: get_value(ec(_3354)),
          :(_3354=<2),
          soil :: get_value(esp(_3682)),
          :(_3682=<10),
          soil :: get_value(ca_carbonate(_4010)),
          :(_4010=<10),
          (         variety :: get_value(value(navel))
          ;         variety :: get_value(value(succar))
          ;         variety :: get_value(value(valencia))
          ), ! &
r2([s_status(suitable_soil)in soil]) if
          (         soil :: get_value(texture(clay_loam))
          ;         soil :: get_value(texture(loam))
          ;         soil :: get_value(texture(sand))
          ;         soil :: get_value(texture(sandy_clay_loam))
          ;         soil :: get_value(texture(sandy_loam))
          ;         soil :: get_value(texture(sily_clay))
          ;         soil :: get_value(texture(sily_clay_loam))
          ;         soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_7088)),
          :(_7088>=1.2),
          soil :: get_value(ec(_7424)),
          :(_7424=<2.5),
          soil :: get_value(esp(_7760)),
          :(_7760=<15),
          soil :: get_value(ca_carbonate(_8088)),
          :(_8088=<15),
          variety :: get_value(value(lime)) &
r3([s_status(unsuitable_soil)in soil]) if
          (       soil :: get_value(texture(coarse_sand))
          ;       soil :: get_value(texture(gravely))
          ;       soil :: get_value(texture(heavy_clay))
          ), ! &
r4([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(water_table_level(_9762)),
          :(_9762<1) &
r5([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(ca_carbonate(_10334)),
          :(_10334>15),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), ! &
r6([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(ca_carbonate(_11594)),
          :(_11594>20),
          variety :: get_value(value(lime)) &
r7([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(esp(_12376)),
          :(_12376>15),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), ! &
r8([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(esp(_13636)),
          :(_13636>20),
          variety :: get_value(value(lime)) &
r9([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(ec(_14412)),
          :(_14412>4),
          plantation :: get_value(existence(no)) &
r10([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(ec(_15208)),
          :(_15208>4),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_15746)),
          :(_15746<5) &
r11([s_status(unsuitable_soil)in soil]) if
          soil :: get_value(ec(_16332)),
          :(_16332>8),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_16870)),
          :(_16870>=5) &
r12([s_status(texture_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(sand))
          ), !,
          soil :: get_value(water_table_level(_18036)),
          :(_18036>=1),
          soil :: get_value(ec(_18364)),
          :(_18364=<4),
          soil :: get_value(esp(_18692)),
          :(_18692=<10),
          soil :: get_value(ca_carbonate(_19020)),
          :(_19020=<10),
          plantation :: get_value(existence(no)),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), ! &
r13([s_status(texture_def)in soil]) if
          soil :: get_value(texture(clay)),
          soil :: get_value(water_table_level(_20800)),
          :(_20800>=1),
          soil :: get_value(ec(_21128)),
          :(_21128=<4),
          soil :: get_value(esp(_21456)),
          :(_21456=<15),
          soil :: get_value(ca_carbonate(_21784)),
          :(_21784=<15),
          plantation :: get_value(existence(no)),
          variety :: get_value(value(lime)) &
r14([s_status(texture_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(sand))
          ), !,
          soil :: get_value(water_table_level(_23390)),
          :(_23390>=1),
          soil :: get_value(ec(_23718)),
          :(_23718=<4),
          soil :: get_value(esp(_24046)),
          :(_24046=<10),
          soil :: get_value(ca_carbonate(_24374)),
          :(_24374=<10),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), !,
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_25606)),
          :(_25606<5) &
r15([s_status(texture_def)in soil]) if
          soil :: get_value(texture(clay)),
          soil :: get_value(water_table_level(_26502)),
          :(_26502>=1),
          soil :: get_value(ec(_26830)),
          :(_26830=<4),
          soil :: get_value(esp(_27158)),
          :(_27158=<15),
          soil :: get_value(ca_carbonate(_27486)),
          :(_27486=<15),
          variety :: get_value(value(lime)),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_28234)),
          :(_28234<5) &
r16([s_status(texture_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(sand))
          ), !,
          soil :: get_value(water_table_level(_29420)),
          :(_29420>=1),
          soil :: get_value(ec(_29748)),
          :(_29748=<8),
          soil :: get_value(esp(_30076)),
          :(_30076=<10),
          soil :: get_value(ca_carbonate(_30404)),
          :(_30404=<10),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_30942)),
          :(_30942>=5),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), ! &
r17([s_status(texture_def)in soil]) if
          soil :: get_value(texture(clay)),
          soil :: get_value(water_table_level(_32532)),
          :(_32532>=1),
          soil :: get_value(ec(_32860)),
          :(_32860=<8),
          soil :: get_value(esp(_33188)),
          :(_33188=<15),
          soil :: get_value(ca_carbonate(_33516)),
          :(_33516=<15),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_34054)),
          :(_34054>=5),
          variety :: get_value(value(lime)) &
r18([s_status(water_table_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(clay_loam))
          ;       soil :: get_value(texture(loam))
          ;       soil :: get_value(texture(sand))
          ;       soil :: get_value(texture(sandy_clay_loam))
          ;       soil :: get_value(texture(sandy_loam))
          ;       soil :: get_value(texture(sily_clay))
          ;       soil :: get_value(texture(sily_clay_loam))
          ;       soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_36896)),
          :(_36896>=1),
          soil :: get_value(water_table_level(_37224)),
          :(_37224<1.2),
          soil :: get_value(ec(_37560)),
          :(_37560=<4),
          soil :: get_value(esp(_37888)),
          :(_37888=<10),
          soil :: get_value(ca_carbonate(_38216)),
          :(_38216=<10),
          plantation :: get_value(existence(no)),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), ! &
r19([s_status(water_table_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(clay_loam))
          ;       soil :: get_value(texture(loam))
          ;       soil :: get_value(texture(sand))
          ;       soil :: get_value(texture(sandy_clay_loam))
          ;       soil :: get_value(texture(sandy_loam))
          ;       soil :: get_value(texture(sily_clay))
          ;       soil :: get_value(texture(sily_clay_loam))
          ;       soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_41746)),
          :(_41746>=1),
          soil :: get_value(water_table_level(_42074)),
          :(_42074<1.2),
          soil :: get_value(ec(_42410)),
          :(_42410=<4),
          plantation :: get_value(existence(no)),
          soil :: get_value(ca_carbonate(_42948)),
          :(_42948=<15),
          variety :: get_value(value(lime)),
          soil :: get_value(esp(_43486)),
          :(_43486=<15) &
r20([s_status(water_table_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(clay_loam))
          ;       soil :: get_value(texture(loam))
          ;       soil :: get_value(texture(sand))
          ;       soil :: get_value(texture(sandy_clay_loam))
          ;       soil :: get_value(texture(sandy_loam))
          ;       soil :: get_value(texture(sily_clay))
          ;       soil :: get_value(texture(sily_clay_loam))
          ;       soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_46138)),
          :(_46138>=1),
          soil :: get_value(water_table_level(_46466)),
          :(_46466<1.2),
          soil :: get_value(ec(_46802)),
          :(_46802=<4),
          soil :: get_value(esp(_47130)),
          :(_47130=<10),
          soil :: get_value(ca_carbonate(_47458)),
          :(_47458=<10),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
          ), !,
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_48690)),
          :(_48690<5) &
r21([s_status(water_table_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(clay_loam))
          ;       soil :: get_value(texture(loam))
          ;       soil :: get_value(texture(sand))
          ;       soil :: get_value(texture(sandy_clay_loam))
          ;       soil :: get_value(texture(sandy_loam))
          ;       soil :: get_value(texture(sily_clay))
          ;       soil :: get_value(texture(sily_clay_loam))
          ;       soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_51336)),
          :(_51336>=1),
          soil :: get_value(water_table_level(_51664)),
          :(_51664<1.2),
          soil :: get_value(ec(_52000)),
          :(_52000=<4),
          soil :: get_value(ca_carbonate(_52328)),
          :(_52328=<15),
          variety :: get_value(value(lime)),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_53076)),
          :(_53076<5),
          soil :: get_value(esp(_53404)),
          :(_53404=<15) &
r22([s_status(water_table_def)in soil]) if
          (       soil :: get_value(texture(clay))
          ;       soil :: get_value(texture(clay_loam))
          ;       soil :: get_value(texture(loam))
          ;       soil :: get_value(texture(sand))
          ;       soil :: get_value(texture(sandy_clay_loam))
          ;       soil :: get_value(texture(sandy_loam))
          ;       soil :: get_value(texture(sily_clay))
          ;       soil :: get_value(texture(sily_clay_loam))
          ;       soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_56056)),
          :(_56056>=1),
          soil :: get_value(water_table_level(_56384)),
          :(_56384<1.2),
          soil :: get_value(esp(_56720)),
          :(_56720=<10),
          soil :: get_value(ca_carbonate(_57048)),
          :(_57048=<10),
          (       variety :: get_value(value(navel))
          ;       variety :: get_value(value(succar))
          ;       variety :: get_value(value(valencia))
```

```
            ), !,
            plantation :: get_value(existence(yes)),
            soil :: get_value(ec(_58280)),
            :(_58280=<8),
            plant :: get_value(age(_58608)),
            :(_58608>=5) &
r23([s_status(water_table_def)in soil]) if
            (           soil :: get_value(texture(clay))
            ;           soil :: get_value(texture(clay_loam))
            ;           soil :: get_value(texture(loam))
            ;           soil :: get_value(texture(sand))
            ;           soil :: get_value(texture(sandy_clay_loam))
            ;           soil :: get_value(texture(sandy_loam))
            ;           soil :: get_value(texture(sily_clay))
            ;           soil :: get_value(texture(sily_clay_loam))
            ;           soil :: get_value(texture(sily_loam))
            ), !,
            soil :: get_value(water_table_level(_61254)),
            :(_61254>=1),
            soil :: get_value(water_table_level(_61582)),
            :(_61582<1.2),
            soil :: get_value(ca_carbonate(_61918)),
            :(_61918=<15),
            variety :: get_value(value(lime)),
            plantation :: get_value(existence(yes)),
            soil :: get_value(esp(_62666)),
            :(_62666=<15),
            soil :: get_value(ec(_62994)),
            :(_62994=<8),
            plant :: get_value(age(_63322)),
            :(_63322>=5) &
r24([s_status(saline)in soil]) if
            (           soil :: get_value(texture(clay))
            ;           soil :: get_value(texture(clay_loam))
            ;           soil :: get_value(texture(loam))
            ;           soil :: get_value(texture(sand))
            ;           soil :: get_value(texture(sandy_clay_loam))
            ;           soil :: get_value(texture(sandy_loam))
            ;           soil :: get_value(texture(sily_clay))
            ;           soil :: get_value(texture(sily_clay_loam))
            ;           soil :: get_value(texture(sily_loam))
            ), !,
            soil :: get_value(water_table_level(_65950)),
            :(_65950>=1),
            soil :: get_value(ec(_66278)),
            :(_66278>2),
            soil :: get_value(ec(_66606)),
            :(_66606=<4),
            soil :: get_value(esp(_66934)),
            :(_66934=<10),
            soil :: get_value(ca_carbonate(_67262)),
            :(_67262=<10),
            plantation :: get_value(existence(no)),
            (           variety :: get_value(value(navel))
            ;         variety :: get_value(value(succar))
            ;         variety :: get_value(value(valencia))
            ), ! &
r25([s_status(saline)in soil]) if
            (         soil :: get_value(texture(clay))
            ;         soil :: get_value(texture(clay_loam))
            ;         soil :: get_value(texture(loam))
            ;         soil :: get_value(texture(sand))
            ;         soil :: get_value(texture(sandy_clay_loam))
            ;         soil :: get_value(texture(sandy_loam))
            ;         soil :: get_value(texture(sily_clay))
            ;         soil :: get_value(texture(sily_clay_loam))
            ;         soil :: get_value(texture(sily_loam))
            ), !,
            soil :: get_value(water_table_level(_70792)),
            :(_70792>=1),
            soil :: get_value(ec(_71120)),
            :(_71120=<4),
            plantation :: get_value(existence(no)),
            soil :: get_value(ec(_71658)),
            :(_71658>2.5),
            soil :: get_value(esp(_71994)),
            :(_71994=<15),
            soil :: get_value(ca_carbonate(_72322)),
            :(_72322=<15),
            variety :: get_value(value(lime)) &
r26([s_status(saline)in soil]) if
            (         soil :: get_value(texture(clay))
            ;         soil :: get_value(texture(clay_loam))
            ;         soil :: get_value(texture(loam))
            ;         soil :: get_value(texture(sand))
            ;         soil :: get_value(texture(sandy_clay_loam))
            ;         soil :: get_value(texture(sandy_loam))
            ;         soil :: get_value(texture(sily_clay))
            ;         soil :: get_value(texture(sily_clay_loam))
            ;         soil :: get_value(texture(sily_loam))
            ), !,
            soil :: get_value(water_table_level(_75180)),
            :(_75180>=1),
            soil :: get_value(ec(_75508)),
            :(_75508>2),
            soil :: get_value(ec(_75836)),
            :(_75836=<4),
            soil :: get_value(esp(_76164)),
            :(_76164=<10),
            soil :: get_value(ca_carbonate(_76492)),
            :(_76492=<10),
            (         variety :: get_value(value(navel))
            ;         variety :: get_value(value(succar))
            ;         variety :: get_value(value(valencia))
            ), !,
            plantation :: get_value(existence(yes)),
            plant :: get_value(age(_77724)),
            :(_77724<5) &
```

```
r27([s_status(saline)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_80370)),
        :(_80370>=1),
        soil :: get_value(ec(_80698)),
        :(_80698=<4),
        soil :: get_value(ec(_81026)),
        :(_81026>2.5),
        soil :: get_value(esp(_81362)),
        :(_81362=<15),
        soil :: get_value(ca_carbonate(_81690)),
        :(_81690=<15),
        variety :: get_value(value(lime)),
        plantation :: get_value(existence(yes)),
        plant :: get_value(age(_82438)),
        :(_82438<5) &
r28([s_status(saline)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_85086)),
        :(_85086>=1),
        soil :: get_value(ec(_85414)),
        :(_85414>2),
        soil :: get_value(esp(_85742)),
        :(_85742=<10),
        soil :: get_value(ca_carbonate(_86070)),
        :(_86070=<10),
        (       variety :: get_value(value(navel))
        ;       variety :: get_value(value(succar))
        ;       variety :: get_value(value(valencia))
        ), !,
        plantation :: get_value(existence(yes)),
        soil :: get_value(ec(_87302)),
        :(_87302=<8),
        plant :: get_value(age(_87630)),
        :(_87630>=5) &
r29([s_status(saline)in soil]) if
```

```
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_90276)),
        :(_90276>=1),
        soil :: get_value(ec(_90604)),
        :(_90604>2.5),
        soil :: get_value(esp(_90940)),
        :(_90940=<15),
        soil :: get_value(ca_carbonate(_91268)),
        :(_91268=<15),
        variety :: get_value(value(lime)),
        plantation :: get_value(existence(yes)),
        soil :: get_value(ec(_92016)),
        :(_92016=<8),
        plant :: get_value(age(_92344)),
        :(_92344>=5) &
r30([s_status(alkaline)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_94972)),
        :(_94972>=1),
        soil :: get_value(ca_carbonate(_95300)),
        :(_95300=<10),
        (       variety :: get_value(value(navel))
        ;       variety :: get_value(value(succar))
        ;       variety :: get_value(value(valencia))
        ), !,
        soil :: get_value(esp(_96322)),
        :(_96322>10),
        soil :: get_value(esp(_96650)),
        :(_96650=<15),
        soil :: get_value(ec(_96978)),
        :(_96978=<4),
        plantation :: get_value(existence(no)) &
r31([s_status(alkaline)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
```

```
          ;         soil :: get_value(texture(sand))
          ;         soil :: get_value(texture(sandy_clay_loam))
          ;         soil :: get_value(texture(sandy_loam))
          ;         soil :: get_value(texture(sily_clay))
          ;         soil :: get_value(texture(sily_clay_loam))
          ;         soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_99836)),
          :(_99836>=1),
          soil :: get_value(ca_carbonate(_100164)),
          :(_100164=<10),
          (         variety :: get_value(value(navel))
          ;         variety :: get_value(value(succar))
          ;         variety :: get_value(value(valencia))
          ), !,
          soil :: get_value(esp(_101186)),
          :(_101186>10),
          soil :: get_value(esp(_101514)),
          :(_101514=<15),
          soil :: get_value(ec(_101842)),
          :(_101842=<4),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_102380)),
          :(_102380<5) &
r32([s_status(alkaline)in soil]) if
          (         soil :: get_value(texture(clay))
          ;         soil :: get_value(texture(clay_loam))
          ;         soil :: get_value(texture(loam))
          ;         soil :: get_value(texture(sand))
          ;         soil :: get_value(texture(sandy_clay_loam))
          ;         soil :: get_value(texture(sandy_loam))
          ;         soil :: get_value(texture(sily_clay))
          ;         soil :: get_value(texture(sily_clay_loam))
          ;         soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_105028)),
          :(_105028>=1),
          soil :: get_value(ca_carbonate(_105356)),
          :(_105356=<10),
          (         variety :: get_value(value(navel))
          ;         variety :: get_value(value(succar))
          ;         variety :: get_value(value(valencia))
          ), !,
          soil :: get_value(esp(_106378)),
          :(_106378>10),
          soil :: get_value(esp(_106706)),
          :(_106706=<15),
          plantation :: get_value(existence(yes)),
          soil :: get_value(ec(_107244)),
          :(_107244=<8),
          plant :: get_value(age(_107572)),
          :(_107572>=5) &
r33([s_status(alkaline)in soil]) if
          (         soil :: get_value(texture(clay))
          ;          soil :: get_value(texture(clay_loam))
          ;          soil :: get_value(texture(loam))
          ;          soil :: get_value(texture(sand))
          ;          soil :: get_value(texture(sandy_clay_loam))
          ;          soil :: get_value(texture(sandy_loam))
          ;          soil :: get_value(texture(sily_clay))
          ;          soil :: get_value(texture(sily_clay_loam))
          ;          soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_110194)),
          :(_110194>=1),
          soil :: get_value(ec(_110522)),
          :(_110522=<4),
          plantation :: get_value(existence(no)),
          soil :: get_value(esp(_111060)),
          :(_111060>15),
          soil :: get_value(esp(_111388)),
          :(_111388=<20),
          soil :: get_value(ca_carbonate(_111716)),
          :(_111716=<15),
          variety :: get_value(value(lime)) &
r34([s_status(alkaline)in soil]) if
          (          soil :: get_value(texture(clay))
          ;          soil :: get_value(texture(clay_loam))
          ;          soil :: get_value(texture(loam))
          ;          soil :: get_value(texture(sand))
          ;          soil :: get_value(texture(sandy_clay_loam))
          ;          soil :: get_value(texture(sandy_loam))
          ;          soil :: get_value(texture(sily_clay))
          ;          soil :: get_value(texture(sily_clay_loam))
          ;          soil :: get_value(texture(sily_loam))
          ), !,
          soil :: get_value(water_table_level(_114568)),
          :(_114568>=1),
          soil :: get_value(ec(_114896)),
          :(_114896=<4),
          soil :: get_value(esp(_115224)),
          :(_115224>15),
          soil :: get_value(esp(_115552)),
          :(_115552=<20),
          soil :: get_value(ca_carbonate(_115880)),
          :(_115880=<15),
          variety :: get_value(value(lime)),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_116628)),
          :(_116628<5) &
r35([s_status(alkaline)in soil]) if
          (          soil :: get_value(texture(clay))
          ;          soil :: get_value(texture(clay_loam))
          ;          soil :: get_value(texture(loam))
          ;          soil :: get_value(texture(sand))
          ;          soil :: get_value(texture(sandy_clay_loam))
          ;          soil :: get_value(texture(sandy_loam))
          ;          soil :: get_value(texture(sily_clay))
```

```
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_119270)),
        :(_119270>=1),
        soil :: get_value(esp(_119598)),
        :(_119598>15),
        soil :: get_value(esp(_119926)),
        :(_119926=<20),
        soil :: get_value(ca_carbonate(_120254)),
        :(_120254=<15),
        variety :: get_value(value(lime)),
        plantation :: get_value(existence(yes)),
        soil :: get_value(ec(_121002)),
        :(_121002=<8),
        plant :: get_value(age(_121330)),
        :(_121330>=5) &
r36([s_status(calcareous)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_123958)),
        :(_123958>=1),
        (       variety :: get_value(value(navel))
        ;       variety :: get_value(value(succar))
        ;       variety :: get_value(value(valencia))
        ), !,
        soil :: get_value(esp(_124980)),
        :(_124980=<15),
        soil :: get_value(ec(_125308)),
        :(_125308=<4),
        plantation :: get_value(existence(no)),
        soil :: get_value(ca_carbonate(_125846)),
        :(_125846>10),
        soil :: get_value(ca_carbonate(_126174)),
        :(_126174=<15) &
r37([s_status(calcareous)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_128822)),
        :(_128822>=1),
        (         variety :: get_value(value(navel))
        ;         variety :: get_value(value(succar))
        ;         variety :: get_value(value(valencia))
        ), !,
        soil :: get_value(esp(_129844)),
        :(_129844=<15),
        soil :: get_value(ec(_130172)),
        :(_130172=<4),
        soil :: get_value(ca_carbonate(_130500)),
        :(_130500>10),
        soil :: get_value(ca_carbonate(_130828)),
        :(_130828=<15),
        plantation :: get_value(existence(yes)),
        plant :: get_value(age(_131366)),
        :(_131366<5) &
r38([s_status(calcareous)in soil]) if
        (         soil :: get_value(texture(clay))
        ;         soil :: get_value(texture(clay_loam))
        ;         soil :: get_value(texture(loam))
        ;         soil :: get_value(texture(sand))
        ;         soil :: get_value(texture(sandy_clay_loam))
        ;         soil :: get_value(texture(sandy_loam))
        ;         soil :: get_value(texture(sily_clay))
        ;         soil :: get_value(texture(sily_clay_loam))
        ;         soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_134014)),
        :(_134014>=1),
        (         variety :: get_value(value(navel))
        ;         variety :: get_value(value(succar))
        ;         variety :: get_value(value(valencia))
        ), !,
        soil :: get_value(esp(_135036)),
        :(_135036=<15),
        soil :: get_value(ca_carbonate(_135364)),
        :(_135364>10),
        soil :: get_value(ca_carbonate(_135692)),
        :(_135692=<15),
        plantation :: get_value(existence(yes)),
        soil :: get_value(ec(_136230)),
        :(_136230=<8),
        plant :: get_value(age(_136558)),
        :(_136558>=5) &
r39([s_status(calcareous)in soil]) if
        (         soil :: get_value(texture(clay))
        ;         soil :: get_value(texture(clay_loam))
        ;         soil :: get_value(texture(loam))
        ;         soil :: get_value(texture(sand))
        ;         soil :: get_value(texture(sandy_clay_loam))
        ;         soil :: get_value(texture(sandy_loam))
        ;         soil :: get_value(texture(sily_clay))
        ;         soil :: get_value(texture(sily_clay_loam))
```

```
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_139180)),
        :(_139180>=1),
        soil :: get_value(ec(_139508)),
        :(_139508=<4),
        plantation :: get_value(existence(no)),
        soil :: get_value(esp(_140046)),
        :(_140046=<20),
        soil :: get_value(ca_carbonate(_140374)),
        :(_140374>15),
        soil :: get_value(ca_carbonate(_140702)),
        :(_140702=<20),
        variety :: get_value(value(lime)) &
r40([s_status(calcareous)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_143554)),
        :(_143554>=1),
        soil :: get_value(ec(_143882)),
        :(_143882=<4),
        soil :: get_value(esp(_144210)),
        :(_144210=<20),
        soil :: get_value(ca_carbonate(_144538)),
        :(_144538>15),
        soil :: get_value(ca_carbonate(_144866)),
        :(_144866=<20),
        variety :: get_value(value(lime)),
        plantation :: get_value(existence(yes)),
        plant :: get_value(age(_145614)),
        :(_145614<5) &
r41([s_status(calcareous)in soil]) if
        (       soil :: get_value(texture(clay))
        ;       soil :: get_value(texture(clay_loam))
        ;       soil :: get_value(texture(loam))
        ;       soil :: get_value(texture(sand))
        ;       soil :: get_value(texture(sandy_clay_loam))
        ;       soil :: get_value(texture(sandy_loam))
        ;       soil :: get_value(texture(sily_clay))
        ;       soil :: get_value(texture(sily_clay_loam))
        ;       soil :: get_value(texture(sily_loam))
        ), !,
        soil :: get_value(water_table_level(_148256)),
        :(_148256>=1),
        soil :: get_value(esp(_148584)),
        :(_148584=<20),
        soil :: get_value(ca_carbonate(_148912)),
        :(_148912>15),
        soil :: get_value(ca_carbonate(_149240)),
        :(_149240=<20),
        variety :: get_value(value(lime)),
        plantation :: get_value(existence(yes)),
        soil :: get_value(ec(_149988)),
        :(_149988=<8),
        plant :: get_value(age(_150316)),
        :(_150316>=5) &
r42([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(texture_def)),
        soil :: get_value(s_status(water_table_def)) &
r43([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(texture_def)),
        soil :: get_value(s_status(saline)) &
r44([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(texture_def)),
        soil :: get_value(s_status(alkaline)) &
r45([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(texture_def)),
        soil :: get_value(s_status(calcareous)),
        soil :: get_value(texture(clay)) &
r46([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(water_table_def)),
        soil :: get_value(s_status(saline)) &
r47([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(water_table_def)),
        soil :: get_value(s_status(alkaline)) &
r48([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(water_table_def)),
        soil :: get_value(s_status(calcareous)),
        soil :: get_value(texture(clay)) &
r49([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(saline)),
        soil :: get_value(s_status(calcareous)),
        soil :: get_value(texture(clay)) &
r50([s_status(unsuitable_soil)in soil]) if
        soil :: get_value(s_status(alkaline)),
        soil :: get_value(s_status(calcareous)),
        soil :: get_value(texture(clay)) &
super(rules)
}.
w_p_p_det_w :: {
r1([w_status(suitable_water)in water]) if
        water :: get_value(boron(_158179)),
        :(_158179=<0.67),
        water :: get_value(eciw(_158515)),
        :(_158515=<1.5),
        water :: get_value(sar(_158851)),
        :(_158851=<8),
        water :: get_value(rsc(_159179)),
        :(_159179=<1.25) &
r2([w_status(unsuitable_water)in water]) if
```

```
          water :: get_value(boron(_159737)),
          :(_159737>=1.0) &
r3([w_status(unsuitable_water)in water]) if
          water :: get_value(sar(_160291)),
          :(_160291>12) &
r4([w_status(unsuitable_water)in water]) if
          water :: get_value(rsc(_160841)),
          :(_160841>2.5) &
r5([w_status(unsuitable_water)in water]) if
          water :: get_value(eciw(_161419)),
          :(_161419>3.0),
          plantation :: get_value(existence(no)) &
r6([w_status(unsuitable_water)in water]) if
          water :: get_value(eciw(_162203)),
          :(_162203>5),
          plantation :: get_value(existence(yes)) &
r7([w_status(boron_def)in water]) if
          water :: get_value(boron(_163047)),
          :(_163047<1),
          water :: get_value(sar(_163375)),
          :(_163375=<12),
          water :: get_value(rsc(_163703)),
          :(_163703=<2.5),
          water :: get_value(boron(_164039)),
          :(_164039>0.67),
          water :: get_value(eciw(_164375)),
          :(_164375=<5) &
r8([w_status(saline1)in water]) if
          water :: get_value(boron(_165029)),
          :(_165029<1),
          water :: get_value(sar(_165357)),
          :(_165357=<12),
          water :: get_value(eciw(_165685)),
          :(_165685>1.5),
          water :: get_value(eciw(_166021)),
          :(_166021=<3),
          water :: get_value(rsc(_166349)),
          :(_166349=<2.5),
          plantation :: get_value(existence(no)) &
r9([w_status(saline1)in water]) if
          water :: get_value(boron(_167241)),
          :(_167241<1),
          water :: get_value(sar(_167569)),
          :(_167569=<12),
          water :: get_value(eciw(_167897)),
          :(_167897>1.5),
          water :: get_value(eciw(_168233)),
          :(_168233=<3),
          water :: get_value(rsc(_168561)),
          :(_168561=<2.5),
          plantation :: get_value(existence(yes)),
          plant :: get_value(age(_169107)),
          :(_169107<5) &
r10([w_status(saline2)in water]) if
```

```
            water :: get_value(boron(_169781)),
            :(_169781<1.0),
            water :: get_value(eciw(_170117)),
            :(_170117=<5),
            water :: get_value(sar(_170445)),
            :(_170445=<12),
            water :: get_value(rsc(_170773)),
            :(_170773=<2.5),
            water :: get_value(eciw(_171109)),
            :(_171109>3),
            plantation :: get_value(existence(yes)),
            plant :: get_value(age(_171647)),
            :(_171647>=5) &
r11([w_status(alkaline)in water]) if
            water :: get_value(boron(_172281)),
            :(_172281<1.0),
            water :: get_value(eciw(_172617)),
            :(_172617=<5),
            water :: get_value(sar(_172945)),
            :(_172945>8),
            water :: get_value(sar(_173273)),
            :(_173273=<12),
            water :: get_value(rsc(_173601)),
            :(_173601=<2.5) &
r12([w_status(alkaline)in water]) if
            water :: get_value(boron(_174247)),
            :(_174247<1.0),
            water :: get_value(eciw(_174583)),
            :(_174583=<5),
            water :: get_value(sar(_174911)),
            :(_174911=<12),
            water :: get_value(rsc(_175239)),
            :(_175239=<2.5),
            water :: get_value(rsc(_175575)),
            :(_175575>=1.25) &
r13([w_status(unsuitable_water)in water]) if
            water :: get_value(w_status(saline1)),
            water :: get_value(w_status(alkaline)) &
r14([w_status(unsuitable_water)in water]) if
            water :: get_value(w_status(saline2)),
            water :: get_value(w_status(alkaline)) &
r15([w_status(unsuitable_water)in water]) if
            water :: get_value(w_status(saline1)),
            water :: get_value(w_status(boron_def)) &
r16([w_status(unsuitable_water)in water]) if
            water :: get_value(w_status(saline2)),
            water :: get_value(w_status(boron_def)) &
r17([w_status(unsuitable_water)in water]) if
            water :: get_value(w_status(boron_def)),
            water :: get_value(w_status(alkaline)) &
super(rules)
}.
c_det_c :: {
r1([c_status(suitable_climate)in climate]) if
```

```
            climate :: get_value(max_d_tc_ss(_179699)),
            :(_179699<35),
            climate :: get_value(min_d_rh_ss(_180027)),
            :(_180027>=40) &
r2([c_status(usuitable_climate)in climate]) if
            climate :: get_value(max_d_tc_ss(_180653)),
            :(_180653>38),
            climate :: get_value(min_d_rh_ss(_180981)),
            :(_180981<26) &
r3([c_status(critical)in climate]) if
            climate :: get_value(max_d_tc_ss(_181567)),
            :(_181567=<38),
            climate :: get_value(max_d_tc_ss(_181895)),
            :(_181895>=35),
            climate :: get_value(min_d_rh_ss(_182223)),
            :(_182223>=26) &
r4([c_status(usuitable_for_navel)in climate]) if
            climate :: get_value(max_d_tc_ss(_182809)),
            :(_182809=<38),
            climate :: get_value(min_d_rh_ss(_183137)),
            :(_183137>=26),
            climate :: get_value(min_d_rh_ss(_183465)),
            :(_183465<40) &
super(rules)
}.
s_p_det_con :: {
r1([text_sp('improving sandy soil texture before cultivation')in conclusion]) if
            soil :: get_value(s_status(texture_def)),
            soil :: get_value(texture(sand)),
            plantation :: get_value(existence(no)) &
r2([text_sp('improving sandy soil texture for existence plantation ')in conclusion]) if
            soil :: get_value(s_status(texture_def)),
            soil :: get_value(texture(sand)),
            plantation :: get_value(existence(yes)) &
r3([text_sp('improving clay soil texture before cultivation')in conclusion]) if
            soil :: get_value(s_status(texture_def)),
            soil :: get_value(texture(clay)),
            plantation :: get_value(existence(no)) &
r4([text_sp('improving clay soil texture for existence plantation ')in conclusion]) if
            soil :: get_value(s_status(texture_def)),
            soil :: get_value(texture(clay)),
            plantation :: get_value(existence(yes)) &
r5([text_sp('improve drainage system to treat the water table level before cultivation')in conclusion])
if
            soil :: get_value(s_status(water_table_def)),
            plantation :: get_value(existence(no)) &
r6([text_sp('improve drainage system to treat the water table level for existence plantation')in
conclusion]) if
            soil :: get_value(s_status(water_table_def)),
            plantation :: get_value(existence(yes)) &
r7([text_sp('reduce soil salinity  by leaching before cultivation')in conclusion]) if
            soil :: get_value(s_status(saline)),
            plantation :: get_value(existence(no)) &
r8([text_sp('reduce soil salinity  by leaching for existence plantation')in conclusion]) if
```

```
        soil :: get_value(s_status(saline)),
        plantation :: get_value(existence(yes)) &
r9([text_sp('reduce soil alkaline by adding Gypsum to replace Sodium with Calcium before
cultivation')in conclusion]) if
        soil :: get_value(s_status(alkaline)),
        plantation :: get_value(existence(no)) &
r10([text_sp('reduce soil alkaline by adding Gypsum to replace Sodium with Calcium for existence
plantation ')in conclusion]) if
        soil :: get_value(s_status(alkaline)),
        plantation :: get_value(existence(yes)) &
r11([text_sp('improve calcareous soil  before cultivation')in conclusion]) if
        soil :: get_value(s_status(calcareous)),
        plantation :: get_value(existence(no)) &
r12([text_sp('improve calcareous soil  for existence plantation')in conclusion]) if
        soil :: get_value(s_status(calcareous)),
        plantation :: get_value(existence(yes)) &
r13([text_sp('your location is not suitable for orange cultivation because you have some soil defect')in
conclusion]) if
        soil :: get_value(s_status(unsuitable_soil)),
        plantation :: get_value(existence(no)) &
r14([text_sp('this plantain is not economic for citrus production because soil properties are not
valid')in conclusion]) if
        soil :: get_value(s_status(unsuitable_soil)),
        plantation :: get_value(existence(yes)) &
super(rules)
}.
w_det_con :: {
r1([text_w('your water source needs to be mixed with another good quality water source ')in
conclusion]) if
        water :: get_value(w_status(boron_def)) &
r2([text_w('use irrigation program to determine irrigation qty')in conclusion]) if
        water :: get_value(w_status(saline1)) &
r3([text_w('your water source needs to be mixed with another good quality water source and use
irrigation program to determine irrigation qty')in conclusion]) if
        water :: get_value(w_status(saline2)) &
r4([text_w('the water is  alkaline and you need to add agricultural Gypsum to the soil')in conclusion])
if
        water :: get_value(w_status(alkaline)) &
super(rules)
}.
w_p_det_con :: {
r1([text_wp('your water quality is not suitable for orange or lime cultivation')in conclusion]) if
        water :: get_value(w_status(unsuitable_water)),
        plantation :: get_value(existence(no)) &
r2([text_wp('this plantain is not economic for citrus production  because water properties is not
valid')in conclusion]) if
        water :: get_value(w_status(unsuitable_water)),
        plantation :: get_value(existence(yes)) &
super(rules)
}.
c_p_det_con :: {
r1([text_cp('Your location climate is critical. You have to prepare your location by  wend break two
years before plantation and follow narrow plant spacing')in conclusion]) if
        climate :: get_value(c_status(critical)),
        plantation :: get_value(existence(no)) &
r2([text_cp('your location climate is critical and you need to establish wend break')in conclusion]) if
        climate :: get_value(c_status(critical)),
        plantation :: get_value(existence(yes)) &
r3([text_cp('your climate is not suitable for orange or lime cultivation')in conclusion]) if
        climate :: get_value(c_status(usuitable_climate)),
        plantation :: get_value(existence(no)) &
r4([text_cp('Your location climate is critical and you need to establish wend break. You have also to
install system to raise air humidity like green cultivation system')in conclusion]) if
        climate :: get_value(c_status(usuitable_climate)),
        plantation :: get_value(existence(yes)) &
super(rules)
}.
s_w_c_v_p_det_con :: {
r1([text_sw('Navel Orange is not suitable to be cultivated in your location but other seedy oranges
may be suitable')in conclusion]) if
        climate :: get_value(c_status(usuitable_for_navel)),
        plantation :: get_value(existence(no)),
        variety :: get_value(value(navel)),
        soil :: get_value(s_status(_201813)),
        :(_201813\==unsuitable_soil),
        water :: get_value(w_status(_202141)),
        :(_202141\==unsuitable_water) &
r2([text_sw('you may replace the scion variety with other seedy oranges or graft some main branches
with compatible bolynaier (i.e. grip fruit and mandaline)')in conclusion]) if
        climate :: get_value(c_status(usuitable_for_navel)),
        soil :: get_value(s_status(_202965)),
        :(_202965\==unsuitable_soil),
        water :: get_value(w_status(_203293)),
        :(_203293\==unsuitable_water),
        variety :: get_value(value(navel)),
        plantation :: get_value(existence(yes)) &
super(rules)
}.
p_v_det_p :: {
r1([yield(0)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_204759)),
        :(_204759<5) &
r2([yield(3)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_205555)),
        :(_205555>=5),
        plant :: get_value(age(_205883)),
        :(_205883<8) &
r3([yield(7)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_206679)),
        :(_206679>=8),
        plant :: get_value(age(_207007)),
        :(_207007<12) &
r4([yield(9)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_207803)),
        :(_207803>=12),
        plant :: get_value(age(_208131)),
        :(_208131<17) &
r5([yield(10)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_208927)),
        :(_208927>=17),
        plant :: get_value(age(_209255)),
        :(_209255<25) &
r6([yield(7)in plant]) if
        variety :: get_value(value(navel)),
        plant :: get_value(age(_210031)),
        :(_210031>=25) &
r7([yield(0)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_210807)),
        :(_210807<5) &
r8([yield(3)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_211603)),
        :(_211603>=5),
        plant :: get_value(age(_211931)),
        :(_211931<8) &
r9([yield(6.5)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_212739)),
        :(_212739>=8),
        plant :: get_value(age(_213067)),
        :(_213067<12) &
r10([yield(8)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_213863)),
        :(_213863>=12),
        plant :: get_value(age(_214191)),
        :(_214191<17) &
r11([yield(8)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_214987)),
        :(_214987>=17),
        plant :: get_value(age(_215315)),
        :(_215315<25) &
r12([yield(6)in plant]) if
        variety :: get_value(value(succar)),
        plant :: get_value(age(_216091)),
        :(_216091>=25) &
r13([yield(0)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_216867)),
        :(_216867<5) &
r14([yield(3)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_217663)),
        :(_217663>=5),
        plant :: get_value(age(_217991)),
        :(_217991<8) &
r15([yield(6)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_218787)),
        :(_218787>=8),
        plant :: get_value(age(_219115)),
        :(_219115<12) &
r16([yield(8)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_219911)),
        :(_219911>=12),
        plant :: get_value(age(_220239)),
        :(_220239<17) &
r17([yield(8)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_221035)),
        :(_221035>=17),
        plant :: get_value(age(_221363)),
        :(_221363<25) &
r18([yield(6)in plant]) if
        variety :: get_value(value(valencia)),
        plant :: get_value(age(_222139)),
        :(_222139>=25) &
r19([yield(0)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_222915)),
        :(_222915<5) &
r20([yield(3)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_223711)),
        :(_223711>=5),
        plant :: get_value(age(_224039)),
        :(_224039<8) &
r21([yield(7)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_224835)),
        :(_224835>=8),
        plant :: get_value(age(_225163)),
        :(_225163<12) &
r22([yield(9)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_225959)),
        :(_225959>=12),
        plant :: get_value(age(_226287)),
        :(_226287<17) &
r23([yield(10)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_227083)),
        :(_227083>=17),
        plant :: get_value(age(_227411)),
        :(_227411<25) &
r24([yield(7)in plant]) if
        variety :: get_value(value(lime)),
        plant :: get_value(age(_228187)),
        :(_228187>=25) &
super(rules)
}.
```

File name: ass_function.pl

```
:-ensure_loaded('$KROL/lib/fun').
actual_yield :: {
        p(plant - actual_yield, previous_yield_production of plant+0.5) &
        super(function)
}.
```

3.3 Inference layer

File name: ass_inference.pl

```
:- ensure_loaded('$KROL/lib/krol_init').
ass_inference :: {
        input(select,[plant-age,variety-value] )&
        output(select,[plant-yield] )&
        input(abstract,[climate-max_d_tc_ss,climate-min_d_rh_ss,plant-age,plantation-
existence,soil-ca_carbonate,soil-ec,soil-esp,soil-texture,soil-water_table_level,variety-
value,water-boron,water-eciw,water-rsc,water-sar] )&
        output(abstract,[climate-c_status,soil-s_status,water-w_status] )&
        input(assign,[climate-c_status,plantation-existence,soil-s_status,soil-texture,water-
w_status] )&
        output(assign,[conclusion-text_cp,conclusion-text_sp,conclusion-text_sw,conclusion-
text_w,conclusion-text_wp] )&
        input(determine_actual_yield,[] )&
        output(determine_actual_yield,[] )&
        description(select, '') &
        select :-
                p_v_det_p :: conclude_all &
        description(abstract, '') &
        abstract :-
                s_p_p_v_det_s :: conclude_all ,
                w_p_p_det_w :: conclude_all ,
                c_det_c :: conclude_all &
        description(assign, '') &
        assign :-
                s_p_det_con :: conclude_all ,
                w_det_con :: conclude_all ,
                w_p_det_con :: conclude_all ,
                c_p_det_con :: conclude_all ,
                s_w_c_v_p_det_con :: conclude_all &
        description(determine_actual_yield, '') &
        determine_actual_yield :-
                actual_yield :: function &
super(krol_init)
}.
```

3.4 Task layer

File name: ass_task.pl

```
% This is to mark that is file is generated by task editor. Please do not delete
```

```
inference_task :: {
super(krol_init)
}.
inference_task_transfer :: {
super(inference_task)
}.
inference_task_uncondional :: {
start_inference :-
        inference_task_user :: init_inf,
        inference_task_user :: determine_exist,
        inference_task_user :: determine_age,
        inference_task_condional :: plantation_not_exist,
        inference_task_condional :: planation_exist_small_age,
        inference_task_condional :: planation_exist_old_age,
        inference_task_condional :: yield_small,
        inference_task_condional :: yield_large &
super(inference_task)
}.
inference_task_condional :: {
plantation_not_exist :-
        (
                plantation :: get_value(existence(_9824)),
                :(_9824=no) ->
                ass_inference :: abstract,
                ass_inference :: assign,
                inference_task_user :: present
        ;       :true
        ) &
planation_exist_small_age :-
        (
                plantation :: get_value(existence(_11575)),
                :(_11575=yes),
                plant :: get_value(age(_11971)),
                :(_11971=<5) ->
                ass_inference :: abstract,
                ass_inference :: assign,
                inference_task_user :: present
        ;       :true
        ) &
planation_exist_old_age :-
        (
                plantation :: get_value(existence(_13694)),
                :(_13694=yes),
                plant :: get_value(age(_14090)),
                :(_14090>5) ->
                ass_inference :: select
        ;       :true
        ) &
yield_small :-
        (
                plantation :: get_value(existence(_15297)),
                :(_15297=yes),
                plant :: get_value(age(_15709)),
                :(_15709>5),
        plant :: get_value(actual_yield(_16168)),
        plant :: get_value(yield(_16158)),
        :(_16168<_16158) ->
                ass_inference :: abstract,
                ass_inference :: assign,
                inference_task_user :: present
        ;       :true
        ) &
yield_large :-
        (
                plantation :: get_value(existence(_18018)),
                :(_18018=yes),
                plant :: get_value(age(_18430)),
                :(_18430>5),
        plant :: get_value(actual_yield(_18889)),
        plant :: get_value(yield(_18879)),
        :(_18889>=_18879) ->
                inference_task_user :: no_need_for_assessement
        ;       :true
        ) &
super(inference_task)
}.
inference_task_repetitive :: {
super(inference_task)
}.
inference_task_user :: {
determine_exist :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        (:compare_date(=<, [PD,PM,PY],[D,M,Y]) ->
                plantation :: set(existence(yes))
        ;       plantation :: set(existence(no))
        ) &
present :-
        assessment_dialog :: run,
        assessment_dialog :: tkwait &
no_need_for_assessement :-
krol_msgs :: show("Your Location is suitable for cultivation",[])&
init_inf :-
        krol_init :: init,
        utility :: restart &
determine_age :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        :dif([PD,PM,PY],[D,M,Y],[_,_,Age]),
        plant :: set(age(Age)) &
super(inference_task)
}.
```

File Name: ass_system.pl

```
:-use_module(library(system)).
:-ensure_loaded('$KROL/lib/messages').
:-ensure_loaded('$KROL/lib/database').
:-ensure_loaded('$KROL/lib/tk_user').
:-ensure_loaded('$KROL/Src/Krol/date').
:-ensure_loaded(ass_concept).
:-ensure_loaded(ass_rules).
:-ensure_loaded(ass_task).
:-ensure_loaded(ass_inference).
:-ensure_loaded(ass_diag).
:-ensure_loaded(ass_function).
ass_start :-
          tcl :: init,
          citex4ds :: open,
          select_table :: fetch([[SN,GN,DN,FN]]),
          farm_data :: set(sid(SN)),
          farm_data :: set(gid(GN)),
          farm_data :: set(did(DN)),
          farm_data :: set(fid(FN)),
          inference_task_uncondional :: start_inference,
          citex4ds :: close.
```

3.5 User Interface

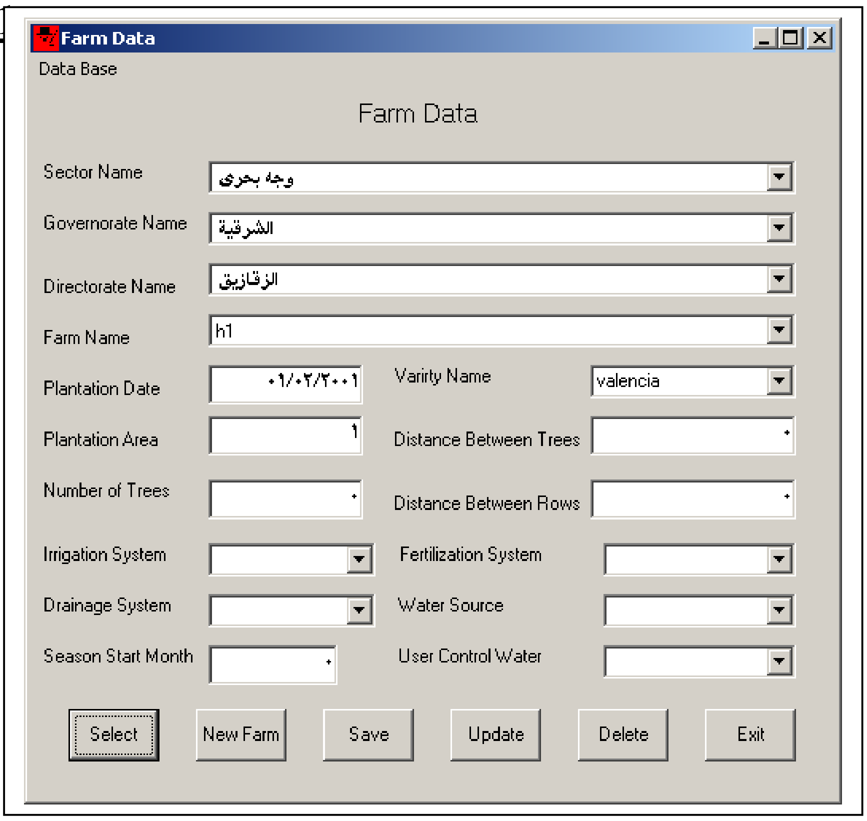
File Name: main.pl

```
:- ensure_loaded('$KROL/lib/buttonbox').
:- ensure_loaded('$KROL/lib/labelframe').
:- ensure_loaded('$KROL/lib/label').
:- use_module(library(system), [exec/3, file_exists/1, environ/2]).
:- use_module(library(charsio), [format_to_chars/3]).
assessment_dialog :-
          tcl :: init,
          assessment_dialog :: run,
          assessment_dialog :: tkwait,
          tcl :: end.
assessment_dialog :: {
widget(assessment_dialog, []) &
window_title('Assessment') &
components(Xs) :- self(D), :findall(X, D :: cs(_, X), Xs) &
pack(ass_frame, ['-side',top]) &
c(ass_frame, assessment_dialog) &
pack(ass_label, ['-side',top]) &
c(ass_label, ass_frame) &
pack(con_frame, ['-side',top]) &
c(con_frame, assessment_dialog) &
pack(con_label, ['-side',top]) &
c(con_label, con_frame) &
pack(ass_buttonboxassessment_dialog, []) &
c(ass_buttonboxassessment_dialog, assessment_dialog) &
super(dialog)
}.
ass_buttonboxassessment_dialog :: {
widget(ass_buttonboxassessment_dialog, ['-orient',horizontal], ['-padx','','-pady','']) &
button(back, ['-text','Back','-command','ass_buttonboxassessment_dialog :: destroy','-
underline',1], '') &
button(vedio, ['-text','MultiMedia Vedio','-command','ass_buttonboxassessment_dialog ::
action(vedio)',             '-underline', 1],'') &
button(text, ['-text','MultiMedia Text','-command','ass_buttonboxassessment_dialog ::
action(text)',    '-underline', 1],'') &
destroy :-
          assessment_dialog :: destroy &
action(vedio):-
          :mm_vedio &
action(text):-
          :mm_text &
super(buttonbox)
}.
ass_frame :: {
widget(ass_frame, ['-labelside',none], []) &
super(labelframe)
}.
ass_label :: {
widget(ass_label, ['-anchor',c,'-text','Assessment subsystem for Citrus cultivation','-padx',0,'-
pady',0,'-relief',raised,'-justify',center], []) &
super(label)
}.
con_frame :: {
widget(con_frame, ['-labelside',none], []) &
super(labelframe)
}.
con_label :: {
widget(con_label, ['-anchor',c,'-text',X,'-padx',0,'-pady',0,'-relief',raised,'-justify',center], []) :-
          conclusion :: get(text_sp(TSP)),
          conclusion :: get(text_w(TW)),
          conclusion :: get(text_wp(TWP)),
          conclusion :: get(text_cp(TCP)),
          conclusion :: get(text_sw(TSW)),
          :format_to_chars("~w~n~w~n~w~n~w~n~w",[TSP,TW,TWP,TCP,TSW],X1),
          :name(X,X1) &
super(label)
}.
mm_vedio:-
          conclusion :: get(text_sp(TSP)),
          ass_video(TSP,Vedio_ass),
        (Vedio_ass = [] ->      krol_msgs :: show('There is no vedio available', []) ;
                        (
                                ([Ved|Tail] = Vedio_ass
                                ),
                                mplay_mm(Ved),
                                (Tail = [] -> true;
                                        ([Ved1|_] = Tail
                                        ),
                                        mplay_mm(Ved1)
                                 )
                        )
        ),
        conclusion :: get(text_w(TW)),
        ass_video(TW,Vedio_ass),
        (Vedio_ass = [] ->      krol_msgs :: show('There is no vedio available', []) ;
                        (([Ved|_] = Vedio_ass) ,mplay_mm(Ved))
        ),
        conclusion :: get(text_wp(TWP)),
        ass_video(TWP,Vedio_ass),
        (Vedio_ass = [] ->      krol_msgs :: show('There is no vedio available', []) ;
                        (([Ved|_] = Vedio_ass) ,mplay_mm(Ved))
        ),
        conclusion :: get(text_cp(TCP)),
        ass_video(TCP,Vedio_ass),
        (Vedio_ass = [] ->      krol_msgs :: show('There is no vedio available', []) ;
                        (([Ved|_] = Vedio_ass) ,mplay_mm(Ved))
        ),
        conclusion :: get(text_sw(TSW)),
        ass_video(TSW,Vedio_ass),
        (Vedio_ass = [] ->      krol_msgs :: show('There is no vedio available', []) ;
                        (([Ved|_] = Vedio_ass) ,mplay_mm(Ved))
        ).
mplay_mm(File) :-
        environ('KROL', KROL),
        format_to_chars('~w/bin/mplayer2.exe ~w/multimedia/clip/~w', [KROL, KROL,
File], CS),
        name(C, CS),
        exec(C, [null,null,null], _),!.
mplay_mm(File) :-
        raise_exception(existance_error(File)).
mm_text:-
        conclusion :: get(text_sp(TSP)),
        ass_text(TSP,Text_ass),
        (Text_ass = [] ->       krol_msgs :: show('There is no text available', []) ;
                                mplay_htm(Text_ass)
        ),
        conclusion :: get(text_w(TW)),
        ass_text(TW,Text_ass),
        (Text_ass = [] ->       krol_msgs :: show('There is no text available', []) ;
                                mplay_htm(Text_ass)
        ),
        conclusion :: get(text_wp(TWP)),
        ass_text(TWP,Text_ass),
        (Text_ass = [] ->       krol_msgs :: show('There is no text available', []) ;
                                mplay_htm(Text_ass)
        ),
        conclusion :: get(text_cp(TCP)),
        ass_text(TCP,Text_ass),
        (Text_ass = [] ->       krol_msgs :: show('There is no text available', []) ;
                                mplay_htm(Text_ass)
        ),
        conclusion :: get(text_sw(TSW)),
        ass_text(TSW,Text_ass),
        (Text_ass = [] ->       krol_msgs :: show('There is no text available', []) ;
                                mplay_htm(Text_ass)
        ).
ass_video(['improving sandy soil texture for existence plantation '],['13.mpg']).
ass_video(['improving sandy soil texture for existence plantation '],['14.mpg']).
ass_video(['improve calcareous soil  for existence plantation'],['13.mpg']).
ass_video(['improve calcareous soil  for existence plantation'],['14.mpg']).
ass_video(['improving sandy soil texture before cultivation'],['13.mpg']).
ass_video(['improving clay soil texture before cultivation'],['13.mpg']).
ass_video(['improving clay soil texture for existence plantation '],[]).
ass_video(['improve drainage system to treat the water table level before cultivation'],[]).
ass_video(['improve drainage system to treat the water table level for existence
plantation'],[]).
ass_video(['reduce soil salinity  by leaching before cultivation'],[]).
ass_video(['reduce soil salinity  by leaching for existence plantation'],[]).
ass_video(['reduce soil alkaline by adding Gypsum to replace Sodium with Calcium before
cultivation'],[]).
ass_video(['reduce soil alkaline by adding Gypsum to replace Sodium with Calcium for
existence plantation '],[]).
ass_video(['improve calcareous soil  before cultivation'],[]).
ass_video(['your location is not suitable for orange cultivation because you have some soil
defect'],[]).
ass_video(['this plantain is not economic for citrus production because soil properties are not
valid'],[]).
ass_video(['your water source needs to be mixed with another good quality water source
'],[]).
ass_video(['use irrigation program to determine irrigation qty'],[]).
ass_video(['your water source needs to be mixed with another good quality water source and
use irrigation program to determine irrigation qty'],[]).
ass_video(['the water is  alkaline and you need to add agricultural Gypsum to the soil'],[]).
ass_video(['your water quality is not suitable for orange or lime cultivation'],[]).
ass_video(['this plantain is not economic for citrus production  because water properties is
not valid'],[]).
ass_video(['Your location climate is critical. You have to prepare your location by  wend
break two years before plantation and follow narrow plant spacing'],[]).
ass_video(['your location climate is critical and you need to establish wend break'],[]).
ass_video(['your climate is not suitable for orange or lime cultivation'],[]).
ass_video(['Your location climate is critical and you need to establish wend break. You have
also to install system to raise air humidity like green cultivation system'],[]).
ass_video(['Navel Orange is not suitable to be cultivated in your location but other seedy
oranges may be suitable'],[]).
ass_video(['you may replace the scion variety with other seedy oranges or graft some main
branches with compatible bolynaier (i.e. grip fruit and mandaline)'],[]).
ass_text(['improving sandy soil texture for existence plantation '],['Book(9).htm#s1']).
ass_text(['improving sandy soil texture for existence plantation '],['Book9.htm#s1']).
ass_text(['improving clay soil texture for existence plantation '],['Book1.htm#t60']).
ass_text(['reduce soil salinity  by leaching before cultivation'],['Book4.htm#w60']).
ass_text(['reduce soil salinity  by leaching for existence plantation'],['Book5.htm#a54']).
ass_text(['reduce soil alkaline by adding Gypsum to replace Sodium with Calcium before
cultivation'],['Book2.htm#l8']).
ass_text(['this plantain is not economic for citrus production because soil properties are not
valid'],['Book1.htm#t60']).
ass_text(['Your location climate is critical. You have to prepare your location by  wend break
two years before plantation and follow narrow plant spacing'],['Book1.htm#t61']).
ass_text(['Your location climate is critical and you need to establish wend break. You have
also to install system to raise air humidity like green cultivation
system'],['Book5.htm#a50','Book6.htm#2' ]).
ass_text(['your climate is not suitable for orange or lime cultivation'],['Book6.htm#q2']).
ass_text(['improve drainage system to treat the water table level before cultivation'],[]).
ass_text(['your location is not suitable for orange cultivation because you have some soil
defect'],[]).
ass_text(['you may replace the scion variety with other seedy oranges or graft some main
branches with compatible bolynaier (i.e. grip fruit and mandaline)'],[]).
ass_text(['improve calcareous soil  for existence plantation'],[]).
ass_text(['improve calcareous soil  for existence plantation'],[]).
ass_text(['improving sandy soil texture before cultivation'],[]).
ass_text(['improving clay soil texture before cultivation'],[]).
ass_text(['improve drainage system to treat the water table level before cultivation'],[]).
ass_text(['improve drainage system to treat the water table level for existence plantation'],[]).
ass_text(['reduce soil alkaline by adding Gypsum to replace Sodium with Calcium for
existence plantation '],[]).
ass_text(['improve calcareous soil  before cultivation'],[]).
ass_text(['improve calcareous soil  for existence plantation'],[]).
ass_text(['your location is not suitable for orange cultivation because you have some soil
defect'],[]).
ass_text(['your water source needs to be mixed with another good quality water source '],[]).
ass_text(['use irrigation program to determine irrigation qty'],[]).
ass_text(['your water source needs to be mixed with another good quality water source and
use irrigation program to determine irrigation qty'],[]).
ass_text(['the water is  alkaline and you need to add agricultural Gypsum to the soil'],[]).
ass_text(['your water quality is not suitable for orange or lime cultivation'],[]).
ass_text(['this plantain is not economic for citrus production  because water properties is not
valid'],[]).
ass_text(['Your location climate is critical and you need to establish wend break. You have
also to install system to raise air humidity like green cultivation system'],[]).
ass_text(['Navel Orange is not suitable to be cultivated in your location but other seedy
oranges may be suitable'],[]).
```

ass_text(['you may replace the scion variety with other seedy oranges or graft some main branches with compatible bolynaier (i.e. grip fruit and mandaline)'],[]).

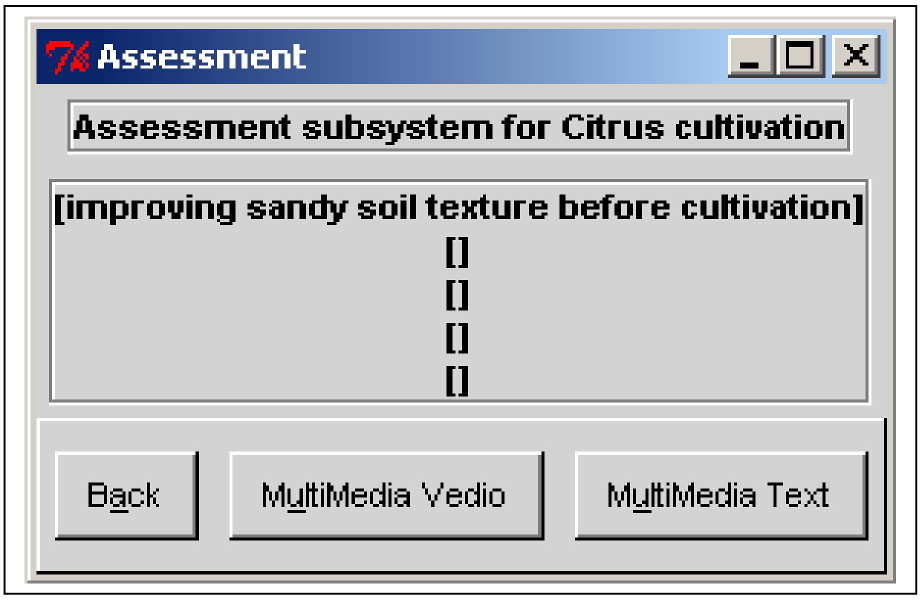
3.6 Test Cases

Case1



Farm Data

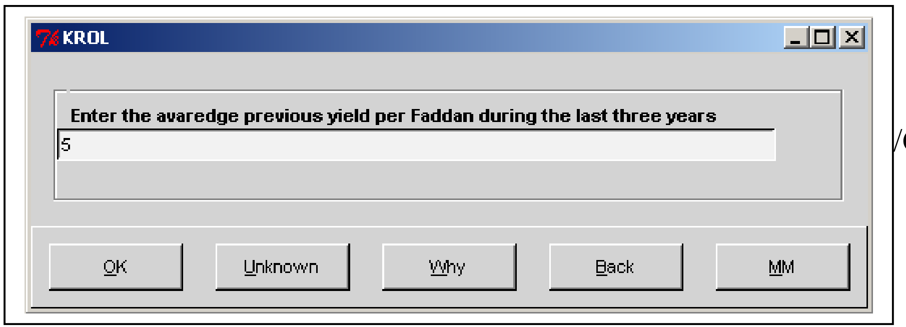
Sector Name	وجه بحرى		
Governorate Name	الشرقية		
Directorate Name	الزقازيق		
Farm Name	h1		
Plantation Date	٠١/٠٢/٢٠٠١	Varirty Name	valencia
Plantation Area	١	Distance Between Trees	٠
Number of Trees	٠	Distance Between Rows	٠
Irrigation System		Fertilization System	
Drainage System		Water Source	
Season Start Month	٠	User Control Water	



Assessment subsystem for Citrus cultivation

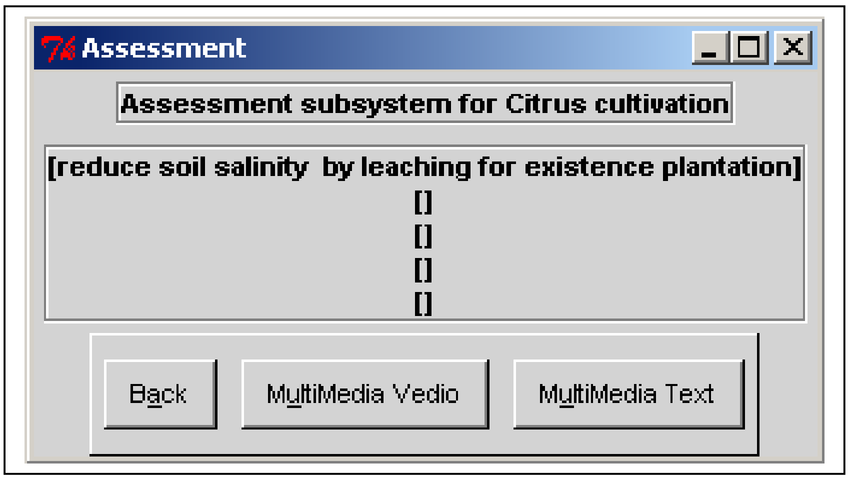
[improving sandy soil texture before cultivation]
[]
[]
[]
[]

Case2



Enter the avaredge previous yield per Faddan during the last three years

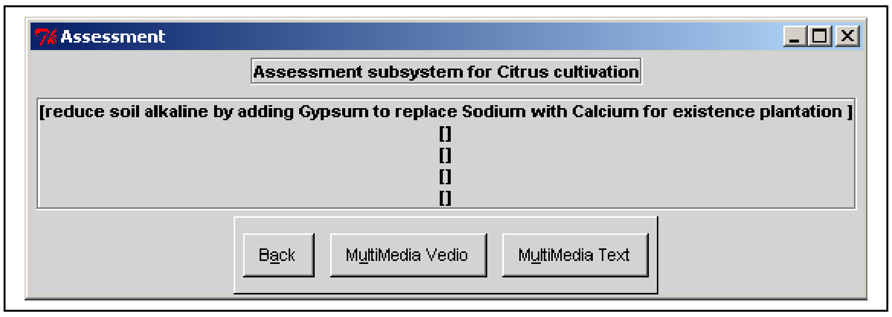
5



Assessment subsystem for Citrus cultivation

[reduce soil salinity by leaching for existence plantation]
[]
[]
[]
[]

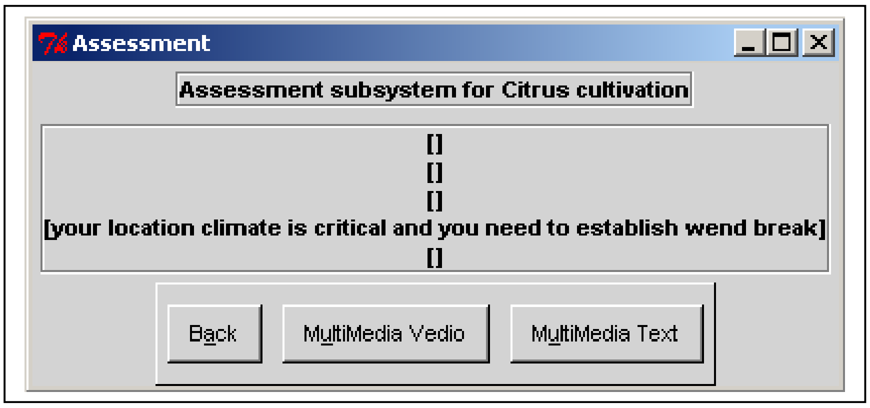
Case 3



Assessment subsystem for Citrus cultivation

[reduce soil alkaline by adding Gypsum to replace Sodium with Calcium for existence plantation]
[]
[]
[]
[]

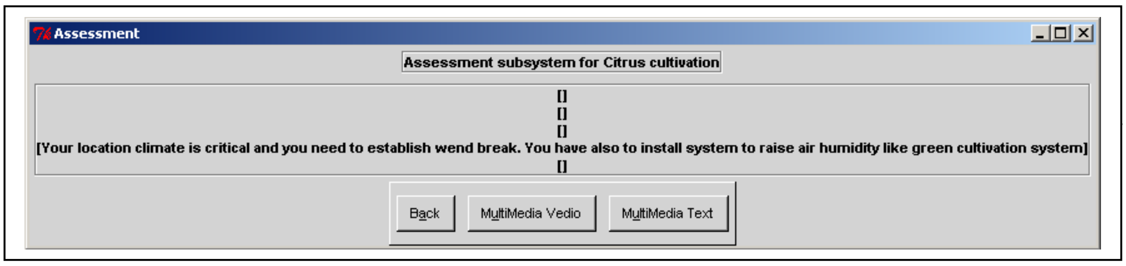
Case4



Assessment subsystem for Citrus cultivation

[]
[]
[]
[your location climate is critical and you need to establish wend break]
[]

Case 5



Assessment subsystem for Citrus cultivation

[]
[]
[]
[Your location climate is critical and you need to establish wend break. You have also to install system to raise air humidity like green cultivation system]
[]

14/2001.5

4. Plant Care_subsystem

4.1. Concepts properties

File name: plcareconcept.pl

```
-ensure_loaded('$KROL/lib/inferenc').
oper :: {
        concept_description('') &
        attributes([
        status([]),
        occurrence([]),
        importance([]),
        material([]),
        video([]),
        method([]),
        text([])]) &
        type(status/1, nominal) &
        source_of_value(status/1, [derived(suggestion_model)]) &
        legal(status/1, [suggested]) &
        type(occurrence/1, nominal) &
        prompt(occurrence/1, '\"هل تم تنفيذ العمليه\"', []) &
        legal(occurrence/1, [ '\"تم عملها\"','\"لم يتم عملها بعد\"']) &
        type(importance/1, nominal) &
        source_of_value(importance/1, [derived(suggestion_model)]) &
        legal(importance/1, [ 'إجبارية','اختيارية' ]) &
        type(material/1, atom) &
        source_of_value(material/1, [derived(assignment_model)]) &
        type(video/1, atom) &
        source_of_value(video/1, [derived(assignment_model)]) &
        type(method/1, atom) &
        source_of_value(method/1, [derived(assignment_model)]) &
        type(text/1, atom) &
        source_of_value(text/1, [derived(assignment_model)]) &
        super(domain_class)
}.
'\"تسوية التربة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ تسوية التربة\"' , []) &
        super(oper)
}.
'\"تقسيم الارض للزراع\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ تقسيم الارض للزراع\"', []) &
        super(oper)
}.
'\"اقامة شبكة الرى بالتنقيط\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ اقامة شبكة الرى بالتنقيط\"', []) &
        super(oper)
}.
'\"زراعة مصدات الرياح\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ زراعة مصدات الرياح\"', []) &
        super(oper)
}.
'\"حفر جور الزراعة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ حفر جور الزراعة\"', []) &
        super(oper)
}.
'\"اضافة الاسمدة العضوية والمعدنية\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ اضافة الاسمدة العضوية والمعدنية\"', []) &
        super(oper)
}.
'\"زراعة الشتلات\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ زراعة الشتلات\"', []) &
        super(oper)
}.
'\"رى الشتلات فى عام الزراعة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ رى الشتلات فى عام الزراعة\"', []) &
        super(oper)
}.
'\"زراعة المحاصيل المؤقتة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ زراعة المحاصيل المؤقتة\"', []) &
        super(oper)
}.
'\"تقليم أشجار الموالح المثمرة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ تقليم أشجار الموالح المثمرة\"', []) &
        super(oper)
}.
'\"استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\"', []) &
        super(oper)
}.
'\"مقاومة الحشائش فى حدائق الموالح المثمرة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ مقاومة الحشائش فى حدائق الموالح المثمرة\"', []) &
        super(oper)
}.
'\"العزيق فى حدائق الموالح المثمرة\"' :: {
        concept_description('') &
        attributes([]) &
        prompt(occurrence/1, '\"هل تم تنفيذ العزيق فى حدائق الموالح المثمرة\"' , []) &
        super(oper)
}.
```

4.2. Relations between expressions

File name: plcarerule.pl

```
:- use_module(library(lists), [memberchk/2]).
:- ensure_loaded('$KROL/lib/rule_exp').
suggestion_model :: {
r1([    status(suggested)in '\"تسوية التربة\"',
        importance('اختيارية')in '\"تسوية التربة\"']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,1,Y],[D,M,Y]) ,:compare_date(>, [1,3,Y],[D,M,Y]);
        :compare_date(=<, [1,7,Y],[D,M,Y]) ,:compare_date(>, [1,9,Y],[D,M,Y])),
        type('\"اقامة بستان حديث\"') in plantation,
        appearance('\"وجود اختلاف كبير فى منسوب التربة \"') in plantation,
        appearance(_16934) in plantation,     :(_16934\=='\"استخدام الرى بالتنقيط\"'),
        occurrence('\"لم يتم عملها بعد\"') in '\"تسوية التربة\"' &
r2([    status(suggested)in '\"تقسيم الارض للزراع\"',
        importance('إجبارية')in '\"تقسيم الارض للزراع\"']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]);
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\"اقامة بستان حديث\"') in plantation ,
        occurrence('\"تم عملها\"') in '\"تسوية التربة\"',
        occurrence('\"لم يتم عملها بعد\"') in '\"تقسيم الارض للزراع\"' &
r3([    status(suggested)in '\"تقسيم الارض للزراع\"',
        importance('إجبارية')in '\"تقسيم الارض للزراع\"']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]);
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\"اقامة بستان حديث\"') in plantation ,
        appearance(_19813) in plantation,     :(_19813\=='\"وجود اختلاف كبير فى منسوب التربة \"'),
        occurrence('\"لم يتم عملها بعد\"') in '\"تسوية التربة\"',
        occurrence('\"لم يتم عملها بعد\"') in '\"تقسيم الارض للزراع\"' &
r4([    status(suggested)in '\"تقسيم الارض للزراع\"',
        importance('إجبارية')in '\"تقسيم الارض للزراع\"']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation,
        appearance(_21553) in plantation,    :(_21553\=='\'استخدام الرى بالتنقيط\''),
        occurrence('\'لم يتم عملها بعد\'') in '\'تسوية التربة\'',
        occurrence('\'لم يتم عملها بعد\'') in '\'تقسيم الارض للزراع\'' &
r5([     status(suggested)in '\'اقامة شبكة الرى بالتنقيط\'',
        importance('إجبارية')in '\'اقامة شبكة الرى بالتنقيط\'']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation ,
        occurrence('\'تم عملها\'') in '\'تقسيم الارض للزراع\'',
        occurrence('\'لم يتم عملها بعد\'') in '\'اقامة شبكة الرى بالتنقيط\'' &
r6([     status(suggested)in '\'زراعة مصدات الرياح\'',
        importance('إجبارية')in '\'زراعة مصدات الرياح\'']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation ,
        occurrence('\'لم يتم عملها بعد\'') in '\'زراعة مصدات الرياح\'' &
r7([     status(suggested)in '\'حفر جور الزراعة\'',
        importance('إجبارية')in '\'حفر جور الزراعة\'']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation ,
        occurrence('\'تم عملها\'') in '\'اقامة شبكة الرى بالتنقيط\'',
        occurrence('\'لم يتم عملها بعد\'') in '\'حفر جور الزراعة\'' &
r8([     status(suggested)in '\'اضافة الاسمدة العضوية والمعدنية\'',
        importance('إجبارية')in '\'اضافة الاسمدة العضوية والمعدنية\'']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,2,Y],[D,M,Y]) ,:compare_date(>, [1,4,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation ,
        occurrence('\'تم عملها\'') in '\'حفر جور الزراعة\'',
        occurrence('\'لم يتم عملها بعد\'') in '\'اضافة الاسمدة العضوية والمعدنية\'' &
r9([     status(suggested)in '\'زراعة الشتلات\'',
        importance('إجبارية')in '\'زراعة الشتلات\'']) if
        plantation :: get(current_date([D,M,Y])),
        (:compare_date(=<, [1,3,Y],[D,M,Y]) ,:compare_date(>, [1,5,Y],[D,M,Y]) ;
        :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,11,Y],[D,M,Y])),
        type('\'اقامة بستان حديث\'') in plantation ,
        occurrence('\'تم عملها\'') in '\'اضافة الاسمدة العضوية والمعدنية\'',
        occurrence('\'لم يتم عملها بعد\'') in '\'زراعة الشتلات\'' &
r10([    status(suggested)in '\'رى الشتلات فى عام الزراعة\'',
        importance('إجبارية')in '\'رى الشتلات فى عام الزراعة\'']) if
        type('\'اقامة بستان حديث\'') in plantation &
r11([    status(suggested)in '\'زراعة المحاصيل المؤقتة\'',
        importance('اختيارية')in '\'زراعة المحاصيل المؤقتة\'']) if
```

```
         plantation :: get(current_date([D,M,Y])),
         (:compare_date(=<, [1,4,Y],[D,M,Y]) ,:compare_date(>, [1,5,Y],[D,M,Y]) ;
         :compare_date(=<, [1,10,Y],[D,M,Y]) ,:compare_date(>, [1,11,Y],[D,M,Y])),
         type('\'اقامة بستان حديث\'') in plantation ,
         appearance('\'وجود ماء رى يكفى لحاجة المؤقتات وشتلات الموالح معا\'') in plantation,
         occurrence('\'تم عملها\'') in '\'زراعة الشتلات\'' &
r12([     status(suggested)in '\'تقليم أشجار الموالح المثمرة\'',
         importance('إجبارية')in '\'تقليم أشجار الموالح المثمرة\'']) if
         appearance('\'تم قطف الثمار\'') in plantation,
         type('\'رعاية البستان\'') in plantation ,
         occurrence('\'لم يتم عملها بعد\'') in '\'تقليم أشجار الموالح المثمرة\'' &
r13([     status(suggested)in '\'استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\'',
         importance('اختيارية')in '\'استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\'']) if
         plantation :: get(current_date([D,M,Y])),
         (:compare_date(=<, [1,1,Y],[D,M,Y]) ,:compare_date(>, [1,5,Y],[D,M,Y]) ;
         :compare_date(=<, [1,9,Y],[D,M,Y]) ,:compare_date(>, [1,10,Y],[D,M,Y])),
         appearance('\'ضعف انتاج الصنف\'') in plantation,
         type('\'رعاية البستان\'') in plantation ,
         occurrence('\'لم يتم عملها بعد\'') in '\'استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\'' &
r14([     status(suggested)in '\'مقاومة الحشائش فى حدائق الموالح المثمرة\'',
         importance('إجبارية')in '\'مقاومة الحشائش فى حدائق الموالح المثمرة\'']) if
         plantation :: get(current_date([D,M,Y])),
         (:compare_date(=<, [1,1,Y],[D,M,Y]) ,:compare_date(>, [1,2,Y],[D,M,Y]);
         :compare_date(=<, [1,4,Y],[D,M,Y]) ,:compare_date(>, [1,11,Y],[D,M,Y]);
         :compare_date(=<, [1,12,Y],[D,M,Y]) ,:compare_date(>=, [31,12,Y],[D,M,Y])),
         appearance(_35517) in plantation,    :(_35517\=='\'الاشجار فى مرحلة التزهير والعقد\''),
         type('\'رعاية البستان\'') in plantation ,
         occurrence('\'لم يتم عملها بعد\'') in '\'مقاومة الحشائش فى حدائق الموالح المثمرة\'' &
r15([     status(suggested)in '\'العزيق فى حدائق الموالح المثمرة\'',
         importance('إجبارية')in '\'العزيق فى حدائق الموالح المثمرة\'']) if
         plantation :: get(current_date([D,M,Y])),
         (:compare_date(=<, [1,1,Y],[D,M,Y]) ,:compare_date(>, [1,2,Y],[D,M,Y]) ;
         :compare_date(=<, [1,4,Y],[D,M,Y]) ,:compare_date(>, [1,11,Y],[D,M,Y]);
         :compare_date(=<, [1,12,Y],[D,M,Y]) ,:compare_date(>=, [31,12,Y],[D,M,Y])),
         type('\'رعاية البستان\'') in plantation ,
         appearance(_37075) in plantation,    :(_37075\=='\'الاشجار فى مرحلة التزهير والعقد\''),
         occurrence('\'لم يتم عملها بعد\'') in '\'العزيق فى حدائق الموالح المثمرة\'' &
super(rules)
}.

assignment_model :: {
r1([      material('لا توجد مادة')in '\'تسوية التربة\'',
         method(m1)in '\'تسوية التربة\'',
         text(t1)in '\'تسوية التربة\'',
         video(v1)in '\'تسوية التربة\'']) if
         status(suggested) in '\'تسوية التربة\'' &
r2([      material('لا توجد مادة')in '\'تقسيم الارض للزراع\'',
         method(m2)in '\'تقسيم الارض للزراع\'',
         text(t2)in '\'تقسيم الارض للزراع\'',
         video(v2)in '\'تقسيم الارض للزراع\'']) if
```

status(suggested) in '\تقسيم الارض للزراع'\" &
r3([material(' مواسير مختلفة الاقطار ٤-٦-٨-١٠-١٢-١٤-١٦ بوصة (٣٠٠ م تقريبا) خراطيم (٢٠٠ م تقريبا) نقاطات (٣٥٠ نقاط) فلاتر-محابس امان- محابس هواء- محابس ضغط- محابس عدم الرجوع-عداد ضغط - فانشورى- - اسياد-تنكات.')in '\اقامة شبكة الرى بالتنقيط'\",
method(m3)in '\اقامة شبكة الرى بالتنقيط'\",
text(t3)in '\اقامة شبكة الرى بالتنقيط'\",
video(v3)in '\اقامة شبكة الرى بالتنقيط'\"]) if
status(suggested) in '\اقامة شبكة الرى بالتنقيط'\" &
r4([material(' شتلات الكازورينا - سماد بلدى - سوبر فوسفات ١٥ %')in '\زراعة مصدات الرياح'\",
method(m4)in '\زراعة مصدات الرياح'\",
text(t4)in '\زراعة مصدات الرياح'\",
video(v4)in '\زراعة مصدات الرياح'\"]) if
status(suggested) in '\زراعة مصدات الرياح'\" &
r5([material('لا توجد مادة')in '\حفر جور الزراعة'\",
method(m5)in '\حفر جور الزراعة'\",
text(t5)in '\حفر جور الزراعة'\",
video(v5)in '\حفر جور الزراعة'\"]) if
status(suggested) in '\حفر جور الزراعة'\" &
r6([material('30 م٣ سماد بلدى قديم كامل التحلل + ١٠٠ كجم سلفات النشادر ٢٠ % + ٥٠ كجم سلفات بوتاسيوم ٤٨ % + ١٠٠ كجم سوبر فوسفات ١٥ %\n')in '\اضافة الاسمدة العضوية والمعدنية'\",
method(m6)in '\اضافة الاسمدة العضوية والمعدنية'\",
text(v6)in '\اضافة الاسمدة العضوية والمعدنية'\",
video(v6)in '\اضافة الاسمدة العضوية والمعدنية'\"]) if
status(suggested) in '\اضافة الاسمدة العضوية والمعدنية'\" &
r7([material('170 شتلة للصنف المراد زراعته\n')in '\زراعة الشتلات'\",
method(m7)in '\زراعة الشتلات'\",
text(t7)in '\زراعة الشتلات'\",
video(v8)in '\زراعة الشتلات'\"]) if
status(suggested) in '\زراعة الشتلات'\" &
r8([material('الماء بمعدل ١٥٠٠ - ١٧٠٠ م٣ / فدان / سنة')in '\رى الشتلات فى عام الزراعة'\",
method(m8)in '\رى الشتلات فى عام الزراعة'\",
text(t8)in '\رى الشتلات فى عام الزراعة'\",
video(v8)in '\رى الشتلات فى عام الزراعة'\"]) if
status(suggested) in '\رى الشتلات فى عام الزراعة'\" &
r9([material(' التقاوى حسب نوع المحصول ومخصبات زراعية واسمدة عضوية ومعدنية .')in '\زراعة المحاصيل المؤقتة'\",
method(m9)in '\زراعة المحاصيل المؤقتة'\",
text(t9)in '\زراعة المحاصيل المؤقتة'\",
video(v9)in '\زراعة المحاصيل المؤقتة'\"]) if
status(suggested) in '\زراعة المحاصيل المؤقتة'\" &
r10([material('لا توجد مادة')in '\تقليم أشجار الموالح المثمرة'\",
method(m10)in '\تقليم أشجار الموالح المثمرة'\",
text(t10)in '\تقليم أشجار الموالح المثمرة'\",
video(v10)in '\تقليم أشجار الموالح المثمرة'\"]) if
status(suggested) in '\تقليم أشجار الموالح المثمرة'\" &
r11([text(t11)in '\استبدال صنف بصنف آخر فى حدائق الموالح المثمرة'\",
method(m11)in '\استبدال صنف بصنف آخر فى حدائق الموالح المثمرة'\",
material(' عيون طعم أو أقلام طعم من أشجار قوية عالية الجودة والإنتاج خالية من الأمراض الفيروسية \n')in '\استبدال صنف بصنف آخر فى حدائق الموالح المثمرة'\",
video(v11)in '\استبدال صنف بصنف آخر فى حدائق الموالح المثمرة'\"]) if

status(suggested) in '\"استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\"' &
r12([material('لا توجد مادة')in '\"مقاومة الحشائش فى حدائق الموالح المثمرة\"',
method(m12)in '\"مقاومة الحشائش فى حدائق الموالح المثمرة\"',
text(t12)in '\"مقاومة الحشائش فى حدائق الموالح المثمرة\"',
video(v12)in '\"مقاومة الحشائش فى حدائق الموالح المثمرة\"']) if
status(suggested) in '\"مقاومة الحشائش فى حدائق الموالح المثمرة\"' &
r13([material('لا توجد مادة')in '\"العزيق فى حدائق الموالح المثمرة\"',
method(m13)in '\"العزيق فى حدائق الموالح المثمرة\"',
text(t13)in '\"العزيق فى حدائق الموالح المثمرة\"',
video(v13)in '\"العزيق فى حدائق الموالح المثمرة\"']) if
status(suggested) in '\"العزيق فى حدائق الموالح المثمرة\"' &
super(rules)
}.

File name: methvedtext.pl

method(m1,'فى حالة استخدام الجرارات و اللودر وسيارات نقل مع الميزان يتم تحديد منسوب التربة ثم نقل الاجزاء الزائدة عن المنسوب المطلوب الى الاجزاء الناقصة عن المنسوب بما يحقق كفاءة عالية فى عملية الرى. وفى حالة استخدام اشعة الليزر يتم التسوية من خلال الآلة').

method(m2,'تقسم الارض الى مسافات متساوية تلائم طبيعة النمو الخضرى للصنف المراد زراعته وهى : ٥ × ٥ م للبرتقال والليمون البذرة والمطعومة
4 × 4 م لليوسفى و الجريب فروت . ').

method(m3,'1 - تركيب خط رئيسى من مواسير P.V.C حسب المساحة ، البئر ، طلمبة الضخ بأقطار تبدأ من ١٦ ، ١٤ ، ١٢ ، ١٠ ، ٨ ، ٦ ،الى ٤ بوصة .
2 - تركيب خطوط فرعية من مادة P.V.C تتحمل ضغط تشغيل حتى ٦ ضغط جوى .
3 - تركيب خراطيم توضع بجوار الاشجار بأقطار من ١٨ - ٢٠ مم مصنوعة من مادة البولي اثيلين عالى الكثافة و مقاوم لاشعة الشمس و التشقق و يتحمل ضغط تشغيل حتى ٤ ضغط جوى .
4 - وضع نقاطات مصنوعة من مادة البولى ايثيلين عال الكثافة شبه الصلب ذات تصرف مائى (٤ ، ٦ ، ٨ لتر / ساعة).
5 - وضع فلاتر سلك ذات سعات مختلفة من ٢٥ - ١٠٠ م٣ / ساعة و مزودة بسلك استانلس ستيل لا يصدأ ذات سعات مختلفة من ٨٠ - ١٢٥ ثقب فى البوصة المربعة لتنقية ماء الرى من الشوائب .
6- وضع اجهزة الآمان للشبكة و هى محبس أمان - محبس هواء - محبس منظم ضغط - محبس عدم رجوع - عداد ضغط - محابس عادية .
7 - وضع اجهزة تسميد مع ماء الرى و هى فانشورى من ٤/٣ - ٢ بوصة ، اسياد حتى ٢ بوصة ، تنكات عادية حتى ٢٠٠ لتر .').

method(m4,'طتزرع شتلات الكازورينا فى اتجاهات الجنوب و الغرب بمعدل صنية فى كل اتجاه بين الشتلة و الاخرى ٢ م و بين الصف و الآخر ٢ م بالتبادل ، و فى اتجاه الشرق و الشمال يزرع صف واحد فى كل اتجاه على مسافة ٢ م بين كل شتلة والاخرى ويفضل اضافة مادة عضوية بمعدل واحد مقطف سماد بلدى لكل شتلة مع ١٠٠ جم سوبر فوسفات (١٥ %).
الأداة المستخدمة فى اجراء العملية : ديتشر لعمل خطوط الزراعة ، فأس لزراعة الشتلات .').

method(m5,'بعد تحديد مسافة الزراعة المناسبة
الطريقة الاولى : تحفر الجور بأبعاد هى : ٦٠ سم طول .
60 سم عرض. 60 سم عمق
مع ضرورة امزاج تراب الحفر حول الجورة
الطريقة الثانية : عمل فنادق بطول صفوف الاشجار بابعاد ٧٠ × ٧٠ × ٧٠ سم . ').

method(m6,'الطريقة الاولى
يوضع لتراب الحفر ١ كجم سوبر فوسفات ١٥ % ، ٢/١ كجم سلفات نشادر ٢٠ % + ٤/١ كجم سلفات بوتاسيوم ٤٨ % + ٢ - ٣ مقطف سماد بلدى قديم كامل التحلل مع خلط المواد المذكورة جيدا.
الطريقة الثانية - وعند الزراعة فى خنادق يوضع للفدان ٣٠ م٣ سماد بلدى قديم كامل التحلل + ١٠٠ كجم سلفات النشادر ٢٠ % + ٥٠ كجم سلفات بوتاسيوم ٤٨ % + ١٠٠ كجم سوبر فوسفات ١٥ % وتخلط الاسمدة مع تراب الحفر.').

method(m7,'توضع الشتلة فى مركز الجورة و تكون منطقة التطعيم عكس اتجاه الريح ثم الردم بمخلوط الاسمدة مع تراب الحفر وداخل الخندق ثم الرى. ').

method(m8,'تروى الاشجار يوميا حسب الاتى

```
. لتر / شجرة / يوم ٢٤ مارس
. لتر / شجرة / يوم ٣٢ ابريل - سبتمبر
. لتر / شجرة / يوم ٢٤ اكتوبر
. لتر / شجرة / يوم ١٢ نوفمبر - فبراير ').
method(m9,'الزراعة، عمل شبكة الرى ، تخطيط التربة ما بين صفوف الاشجار حسب نوع المحصول .').
method(m10, ': بعد جمع المحصول يتم التقليم كالآتى
1-      إزالة السرطانات و الأفرخ المائية و الأفرع الجافة
2-      إزالة الأجزاء الجافة من الأفرع
3-      فتح قلب الشجرة بدرجة متوسطة
4-      إزالة الأفرع المصابة بالحشرات والأمراض
5-      قص قمم الأفرع المرتفعة بحيث لا يزيد ارتفاع الشجرة عن ٣-٣,٥ م
6-      قص قمم الأفرع المتداخلة بين صفوف الأشجار لتوفير مساحة كافية بين صفوف الأشجار لإجراء عمليات الخدمة الزراعية وتوفير التهوية والإضاءة اللازمة للأشجار
الأشجار التى لا تقلم فى ديسمبر ويناير يمكن تقليمها فى يوليو وأغسطس').
method(m11,'فى حالة التطعيم بالقلم :قرط الأشجار المراد تغيير صنفها أسفل منطقة التطعيم بحوالى ١٠سم ثم يطعم الأصل (نارنج) بأقلام الصنف الجديد حيث يتم تركيب ٣-٤ أقلام على محيط الجذع .
فى حالة التطعيم بالعين :بعد قرط الأشجار بعد منطقة التطعيم بحوالى ١٠سم يربى ٣-٤ سرطانات (الأفرع النامية من اصل النارنج) ثم تطعيم هذه الأفرع بالعين بالصنف الجديد بعد وصولها إلى نمو ملائم للتطعيم وغالبا فى ديسمبر إذا تم القطع فى يناير أو فبراير .
و فى الحالتين يجب إزالة السرطانات التى تنمو على الأصل مع الاهتمام بخدمة الحديقة وبالتسميد والرى ومقاومة الحشائش والآفات').
method(m12,'
1-      إجراء عزقة رئيسية عميقة خلال ديسمبر أو يناير
2-      إجراء عزيق سطحى للحديقة إذا دعت الحاجة خلال أشهر أبريل - مايو أو حش الحشائش و تركها فوق سطح التربة خلال هذان الشهران.
3-      إجراء عزيق سطحى فى حالة وجود حشائش خلال الفترة من يوليو و حتى أكتوبر (عزيق خربشة).
4-      يمكن استخدام مبيدات الحشائش فى مقاومة الحشائش ابتداء من يوليو و حتى نهاية أكتوبر كالآتى:
أ - حشائش حولية بنوعيها عريضة و ضيقة الأوراق و يستخدم مبيد الجرامكسون بمعدل ١لتر/ ٢٠٠ لتر ماء/ فدان من ٢-٣ مرات بفاصل شهر واحد بين الرشة و الأخرى أو مبيد الباستا ٢٠ بمعدل من ٢-٤ لتر/٢٠٠لتر/ فدان دفعة واحدة (٤ لتر/فدان) أو على دفعتين (٢لتر/فدان) بفاصل من ١-٢ شهر بين الرشة والأخرى .
ب- الرجلة والحشائش الحولية العريضة يستخدم مبيد الجيسابريم ¾ كجم +٢٠٠سم جرامكسون /٢٠٠ لترماء / فدان.
ج- حشائش معمرة (نجيل- سعد-حلفا-هجنة-عليق) يستخدم راوند آب بمعدل (٢٠سم٣ +١٠جم سلفات نوشادر+١/٢سم٣ زيت طعام) /١لتر ماء على أن ترش الأماكن المصابة فقط بالمبيد ويمكن تكرار العملية مرة ثانية بعد مرور شهر من موعد الرشة الأولى و ذلك فى حالة عدم القضاء على الحشائش.
اما إذا كانت الحديقة موبوءة بالحشائش المعمرة ترش التربة بصفة عامة فى وجود الحشائش بالرواند آب بمعدل ٤لتر مبيد + ٢ كجم سلفات نوشادر + ١٠٠سم٣ زيت طعام و ذلك لكل ٢٠٠ لتر ماء/فدان.
يجب أن يسبق عملية المقاومة بالمبيدات رى الحديقة ولا تروى إلا بعد مرور ٥-٧ أيام من الرش و يكون الرش فى الصباح الباكر بعد تطاير الندى أو فى فترة ما بين الظهيرة.        ').
method(m13,'
1-      عزقة رئيسية عميقة بالفأس خلال ديسمبر أو يناير
2-      عزيق سطحى قبل التزهير إذا دعت الحاجة لذلك
3-      عزيق خربشة أو حش الحشائش خلال الفترة من ابريل - يونيو
4-      عزيق سطحى خلال الفترة من يوليو و حتى أكتوبر
الحدائق التى تروى بالتنقيط يكتفى بالعزيق حول الأشجار فقط أى المنطقة التى يوجد بها الحشائش').
video(v1,['14.mpg']).
video(v2,['15.mpg']).          %video(v2,['15.mpg','16.mpg']).
video(v3,[]).
video(v5,['17.mpg']).
video(v4,[]).
video(v6,['24.mpg']).          %video(v6,['0.mpg','24.mpg']).
video(v7,['4.mpg']).              %video(v7,[4,5,6,18,19,21,22]).
video(v8,['9.mpg']).              %video(v8,[9,23]).
video(v9,['25.mpg']).
video(v10,['31.mpg']).
video(v11,[]).
video(v12,['26.mpg']).            %video(v12,[26,35,37]).
video(v13,[]).
text(t1,[]).
text(t2,[]).
text(t3,[]).
text(t4,[]).
text(t5,['Book4#w13.htm','Book5#A13.htm','Book9#S9.htm']).
text(t6,[]).
text(t7,[]).
text(t8,[]).
text(t9,[]).
text(t10,['Book3#Q6', 'Book4#W16', 'Book5#A17', 'Book9#S12','Book11#T3']).
text(t11,[]).
text(t12,['Book5#A18', 'Book9#S13', 'Book10#video3']).
text(t13,[]).
```

File name: Knowledge.pl

```
knowledge(   '\"تسوية التربة\"' ,'
اسم العملية : تسوية التربة .
الشروط اللازمة لاجراء العملية : وجود اختلافات كبيرة فى منسوب التربة .
موانع اجراء العملية : استخدام الرى بالتنقيط.
تأثير اجراء العملية على الانتاج : زيادة كفاءة عملية الرى و زيادة الانتاج عند استخدام الرى بالغمر او الرش .
تأثير اجراء العملية على البيئة : لا يوجد.
اهمية اجراء العملية : (اختيارية) تسوية سطح التربة للتحكم فى توزيع ماء الرى بالغمر او الرش.
عدد مرات اجراء العملية : واحدة.
مسلسل العملية : ١
تصنيف العملية : قبل الزراعة.
العملية السابقة : لا يوجد.
موعد اجراء العملية : يناير وفبراير / يوليو واغسطس .
الوقت اللازم بين موعد اجراء العملية و العملية السابقة : لا يوجد .
فترة اجراء العملية عامل (يوم / فدان) : ٥ عامل / فدان / يوم .
المادة المستخدمة و كميتها : لا يوجد .
وقت اجراء العملية خلال اليوم : طوال اليوم .
 الطريقة المستخدمة فى اجراء العملية : فى حالة استخدام الجرارات و اللودر وسيارات نقل مع الميزان يتم تحديد منسوب
 التربة ثم نقل الاجزاء الزائدة عن المنسوب المطلوب الى الاجزاء الناقصة عن المنسوب بما يحقق كفاءة عالية فى عملية
الرى. وفى حالة استخدام اشعة الليزر يتم التسوية من خلال الآلة .
 الأداة المستخدمة فى اجراء العملية : الميزان - قوائم لتحديد منسوب التربة المطلوب - اشعة الليزر او جرارات و لودر
وسيارات نقل.
').
knowledge(   '\"تقسيم الارض للزراع\"','
اسم العملية : تقسيم الارض للزراعة .
الشروط اللازمة لاجراء العملية : عدم وجود اختلافات كبيرة فى منسوب التربة.
موانع اجراء العملية : وجود اختلافات كبيرة فى منسوب التربة.
تأثير اجراء العملية على الانتاج : زيادة الانتاج .
تأثير اجراء العملية على البيئة : لا يوجد .
```

. اهمية اجراء العملية : (اجبارية) الزراعة على مسافات منتظمة ملائمة لنمو الصنف
عدد مرات اجراء العملية : واحدة.
مسلسل العملية : ٢
. تصنيف العملية : قبل الزراعة
العملية السابقة : التسوية اذا كانت ضرورية.
. موعد اجراء العملية : فبراير / سبتمبر
. الوقت اللازم بين موعد اجراء العملية والعملية السابقة : مباشرة
. فترة اجراء العملية عامل (يوم / فدان) : ٥ عمال / فدان / يوم
المادة المستخدمة و كميتها : ٥٠ كجم جير مطفى.
. وقت اجراء العملية خلال اليوم : طوال فترة النهار
الطريقة المستخدمة فى اجراء العملية : تقسم الارض الى مسافات متساوية تلائم طبيعة النمو الخضرى للصنف المراد زراعته وهى :
. م للبرتقال والليمون البذرة والمطعومة 5 × 5
. م لليوسفى و الجريب فروت 4 × 4
الأداة المستخدمة فى اجراء العملية : واحد شريط مقاس ٥٠ او ١٠٠ متر ، اوناد خشبية (١٧٠ م) ، فأس ، يمكن استخدام جرار زراعى مع او تستخدم الفأس.').
knowledge('\'اقامة شبكة الرى بالتنقيط\'',"
اسم العملية : اقامة شبكة الرى بالتنقيط.
الشروط اللازمة لاجراء العملية : مراعاة الاحتياجات المائية لاشجار الموالح عند الاثمار التجارى.
موانع اجراء العملية : لا يوجد.
. تأثير اجراء العملية على الانتاج : الحصول على الانتاج الاقتصادى المربح
. تأثير اجراء العملية على البيئة : لا يوجد
. اهمية اجراء العملية : (اجبارية) توفير الاحتياجات المائية للاشجار
. عدد مرات اجراء العملية : واحدة
مسلسل العملية : ٣.
تصنيف العملية : قبل الزراعة.
. العملية السابقة : تقسيم الارض للزراعة
. موعد اجراء العملية : يناير وفبراير / يوليو واغسطس
الوقت اللازم بين موعد اجراء العملية و العملية السابقة : بعد التقسيم مباشرة.
. فترة اجراء العملية عامل (يوم / فدان) : ١٠ عامل / فدان / يوم
المادة المستخدمة و كميتها : مواسير مختلفة الاقطار ٤-٦-٨-١٠-١٢-١٤-١٦ بوصة (٣٠٠ م تقريبا) خراطيم (٢٠٠ م تقريبا) نقاطات (٣٥٠ نقاط) فلاتر-محابس امان- محابس هواء- محابس ضغط- محابس عدم الرجوع-عداد ضغط - فانشورى- - اسياد-تنكات.
. وقت اجراء العملية خلال اليوم : طوال فترة النهار
الطريقة المستخدمة فى اجراء العملية :
1 - تركيب خط رئيسى من مواسير P.V.C ، ١٠ ، ١٢ ، ١٤ ، ١٦ حسب المساحة ، البئر ، طلمبة الضخ بأقطار تبدأ من ٨ ، ٦ ،الى ٤ بوصة .
2 - تركيب خطوط فرعية من مادة P.V.C تتحمل ضغط تشغيل حتى ٦ ضغط جوى .
3 - تركيب خراطيم توضع بجوار الاشجار بأقطار من ١٨ - ٢٠ مم مصنوعة من مادة البولي اثيلين عالى الكثافة و مقاوم لاشعة الشمس و التشقق و يتحمل ضغط تشغيل حتى ٤ ضغط جوى .
4 - وضع نقاطات مصنوعة من مادة البولى ايثيلين عال الكثافة شبه الصلب ذات تصرف مائى (٤ ، ٦ ، ٨ لتر / ساعة)
.
5 - وضع فلاتر سلك ذات سعات مختلفة من ٢٥ - ١٠٠ م٣ / ساعة و مزودة بسلك استانلس ستيل لا يصدأ ذات سعات مختلفة من ٨٠ - ١٢٥ ثقب فى البوصة المربعة لتنقية ماء الرى من الشوائب .
6- وضع اجهزة الآمان للشبكة و هى محبس أمان - محبس هواء - محبس منظم ضغط - محبس عدم رجوع - عداد ضغط - محابس عادية .
7 - وضع اجهزة تسميد مع ماء الرى و هى فانشورى من ٣/٤ - ٢ بوصة ، اسياد حتى ٢ بوصة ، تنكات عادية حتى ٢٠٠ لتر .
الأداة المستخدمة فى اجراء العملية : حفار تربة - فأس - مقاطف - رمل - اسمنت - طوب .').
knowledge('\'زراعة مصدات الرياح\'',"

. اسم العملية : زراعة مصدات الرياح
. موانع اجراء العملية : لا يوجد. الشروط اللازمة لاجراء العملية : استخدام شتلات قوية
. تأثير اجراء العملية على الانتاج : حماية الاشجار من تأثير العواصف يسبب زيادة الانتاج
. تأثير اجراء العملية على البيئة : لا يوجد
. اهمية اجراء العملية : (اجبارية) توفير الحماية للاشجار من العواصف الرملية والرياح الساخنة
. عدد مرات اجراء العملية : واحدة
. مسلسل العملية : ٤
. تصنيف العملية : قبل الزراعة
. العملية السابقة : اقامة شبكة الرى
. موعد اجراء العملية : الربيع / الخريف
. الوقت اللازم بين موعد اجراء العملية و العملية السابقة : لا يوجد
. فترة اجراء العملية عامل (يوم / فدان) : ٣ عمال / فدان / يوم
%. المادة المستخدمة و كميتها : شتلات الكازورينا - سماد بلدى - سوبر فوسفات ١٥
. وقت اجراء العملية خلال اليوم : طوال ساعات النهار
الطريقة المستخدمة فى اجراء العملية : تزرع شتلات الكازورينا فى اتجاهات الجنوب و الغرب بمعدل صنية فى كل اتجاه بين الشتلة و الاخرى ٢ م و بين الصف و الآخر ٢ م بالتبادل ، و فى اتجاه الشرق و الشمال يزرع صف واحد فى كل اضافة مادة عضوية بمعدل واحد مقطف سماد بلدى لكل شتلة مع اتجاه على مسافة ٢ م بين كل شتلة والاخرى ويفضل
.(١٥% ١٠٠ جم سوبر فوسفات
.('. الأداة المستخدمة فى اجراء العملية : ديتشر لعمل خطوط الزراعة ، فأس لزراعة الشتلات
knowledge('\حفر جور الزراعة'\",'
. اسم العملية : حفر جور الزراعة
. الشروط اللازمة لاجراء العملية : ابعاد مناسبة للجورة فى الطول والعرض والعمق
. موانع اجراء العملية : لا يوجد
. تأثير اجراء العملية على الانتاج : الحصول على نمو جيد للشتلات
. تأثير اجراء العملية على البيئة : لا يوجد
. اهمية اجراء العملية : (اجبارية) توفير مهد مناسب لنمو الشتلات الصغيرة
. عدد مرات اجراء العملية : واحدة
. مسلسل العملية : ٥
. تصنيف العملية : قبل الزراعة
. العملية السابقة : تقسيم الارض
. موعد اجراء العملية : فبراير / سبتمبر
. الوقت اللازم بين موعد اجراء العملية و العملية السابقة : لا يوجد
. فترة اجراء العملية عامل / يوم / فدان : ١٠ عمال / فدان / يوم
. المادة المستخدمة و كميتها : لا يوجد
. وقت اجراء العملية خلال اليوم : طوال اليوم
الطريقة المستخدمة فى اجراء العملية : بعد تحديد مسافة الزراعة المناسبة
: الطريقة الاولى : تحفر الجور بأبعاد هى
. سم طول 60
. سم عرض 60
. سم عمق 60
مع ضرورة امزاج تراب الحفر حول الجورة
. الطريقة الثانية : عمل فنادق بطول صفوف الاشجار بابعاد ٧٠ × ٧٠ × ٧٠ سم
.الأداة المستخدمة فى اجراء العملية : ميكنة حفر الجور او الفأس وفى حالة الخنادق يستخدم الديتشر
').
knowledge('\اضافة الاسمدة العضوية والمعدنية'\" ,'
.اسم العملية : اضافة الاسمدة العضوية والمعدنية
: الشروط اللازمة لاجراء العملية
. موانع اجراء العملية : لا يوجد
.تأثير اجراء العملية على الانتاج : انتاجية مرتفعة
. تأثير اجراء العملية على البيئة : لا يوجد

. اهمية اجراء العملية : (اجبارية) انشاء مزرعة موالح
. عدد مرات اجراء العملية : واحدة
. مسلسل العملية : ٦
. تصنيف العملية : الزراعة
. العملية السابقة : اعداد الجور للزراعة
. موعد اجراء العملية : مارس وابريل / سبتمبر واكتوبر
.الوقت اللازم بين موعد اجراء العملية والعملية السابقة : مباشرة
. فترة اجراء العملية عامل (يوم / فدان) : ٣ عامل / فدان / يوم
المادة المستخدمة و كميتها : ٣٠ م٣ سماد بلدى قديم كامل التحلل + ١٠٠ كجم سلفات النشادر ٢٠ % + ٥٠ كجم سلفات بوتاسيوم ٤٨ % + ١٠٠ كجم سوبر فوسفات ١٥ %.
. وقت اجراء العملية خلال اليوم : خلال ساعات النهار
: الطريقة المستخدمة فى اجراء العملية
الطريقة الاولى - يوضع لتراب الحفر ١ كجم سوبر فوسفات ١٥ % ، ٢/١ كجم سلفات نشادر ٢٠ % + ٤/١ كجم سلفات بوتاسيوم ٤٨ % + ٢ - ٣ مقطف سماد بلدى قديم كامل التحلل مع خلط المواد المذكورة جيدا.
الطريقة الثانية - وعند الزراعة فى خنادق يوضع للفدان ٣٠ م٣ سماد بلدى قديم كامل التحلل + ١٠٠ كجم سلفات النشادر ٢٠ % + ٥٠ كجم سلفات بوتاسيوم ٤٨ % + ١٠٠ كجم سوبر فوسفات ١٥ % وتخلط الاسمدة مع تراب الحفر.
.'). الأداة المستخدمة فى اجراء العملية : الفأس
knowledge('\"زراعة الشتلات\"', '
. اسم العملية : زراعة الشتلات
. الشروط اللازمة لاجراء العملية : شتلات قوية - ممثلة للصنف - خالية من الافات
. موانع اجراء العملية : لا يوجد
. تأثير اجراء العملية على الانتاج : انتاجية مرتفعة
. تأثير اجراء العملية على البيئة : لا يوجد
. اهمية اجراء العملية : (اجبارية) انشاء مزرعة موالح
. عدد مرات اجراء العملية : واحدة
. مسلسل العملية : ٧
. تصنيف العملية : الزراعة
. العملية السابقة : اضافة الاسمدة العضوية والمعدنية
. موعد اجراء العملية : مارس وابريل / سبتمبر واكتوبر
. الوقت اللازم بين موعد اجراء العملية و العملية السابقة : مباشرة
. فترة اجراء العملية عامل (يوم / فدان) : ٣ عامل / فدان / يوم
.المادة المستخدمة و كميتها : ١٧٠ شتلة للصنف المراد زراعته
. وقت اجراء العملية خلال اليوم : خلال ساعات النهار
: الطريقة المستخدمة فى اجراء العملية
توضع الشتلة فى مركز الجورة و تكون منطقة التطعيم عكس اتجاه الريح ثم الردم بمخلوط الاسمدة مع تراب الحفر وداخل الخندق ثم الرى.
.').الأداة المستخدمة فى اجراء العملية : الفأس
knowledge('\"رى الشتلات فى عام الزراعة\"', '
. اسم العملية : رى الشتلات فى عام الزراعة
. الشروط اللازمة لاجراء العملية : توفر ماء صالح للرى ، شبكة رى بالتنقيط
. موانع اجراء العملية : لا يوجد
.تأثير اجراء العملية على الانتاج : نمو جيد للشتلات يعطى اشجار جيدة
. تأثير اجراء العملية على البيئة : لا يوجد
. اهمية اجراء العملية : (اجبارية) امداد الشتلات بالماء
.عدد مرات اجراء العملية : يوميا
. مسلسل العملية : ٨
. تصنيف العملية : بعد الزراعة مباشرة
. العملية السابقة : لا يوجد
. موعد اجراء العملية : طوال العام
. الوقت اللازم بين موعد اجراء العملية و العملية السابقة : بعد الزراعة مباشرة

. فترة اجراء العملية عامل (يوم / فدان) : عامل / يوم / فدان
.المادة المستخدمة و كميتها : الماء بمعدل ١٥٠٠ - ١٧٠٠ م٣ / فدان / سنة
. وقت اجراء العملية خلال اليوم : فترة ما قبل الظهيرة او بعد فترة الظهيرة
: الطريقة المستخدمة فى اجراء العملية : تروى الاشجار يوميا حسب الاتى

. لتر / شجرة / يوم ٢٤ مارس
. لتر / شجرة / يوم ٣٢ابريل - سبتمبر
. لتر / شجرة / يوم ٢٤ اكتوبر
. لتر / شجرة / يوم ١٢ نوفمبر - فبراير

.('. الأداة المستخدمة فى اجراء العملية : النقاطات
knowledge('\"زراعة المحاصيل المؤقتة\"' ,'
.اسم العملية : زراعة المحاصيل المؤقتة
. الشروط اللازمة لاجراء العملية : زراعة محاصيل خضر لا تؤثر على نمو الاشجار
.موانع اجراء العملية : عدم وجود ماء رى يكفى بحاجة المؤقتات وشتلات الموالح معا
.تأثير اجراء العملية على الانتاج : استغلال مسافات الزراعة بين صفوف الاشجار وزيادة عائد الفدان
. تأثير اجراء العملية على البيئة : لا يوجد
.اهمية اجراء العملية : تحسين صفات التربة و خدمة الحديقة
. (عدد مرات اجراء العملية : اثنين (محصول شتوى وآخر صيفى
. مسلسل العملية : ٩
. تصنيف العملية : (اختيارية) بعد زراعة شتلات الموالح
. العملية السابقة : اعداد الارض لزراعة المؤقتات
. موعد اجراء العملية : ابريل / اكتوبر
. الوقت اللازم بين موعد اجراء العملية و العملية السابقة : اسبوع
. فترة اجراء العملية عامل (يوم / فدان) : ٥ عامل / فدان / يوم
. المادة المستخدمة و كميتها : التقاوى حسب نوع المحصول ومخصبات زراعية واسمدة عضوية ومعدنية
. وقت اجراء العملية خلال اليوم : طوال ساعات النهار
الطريقة المستخدمة فى اجراء العملية : تخطيط التربة ما بين صفوف الاشجار حسب نوع المحصول ، عمل شبكة الرى ، الزراعة .
.('. الأداة المستخدمة فى اجراء العملية : جرار زراعى - خطاطة - شبكة رى - فأس
knowledge('\"تقليم أشجار الموالح المثمرة\"','
(اسم العملية : تقليم أشجار الموالح المثمرة (الوادى / الأراضى المستصلحة
الشروط اللازمة لإجراء العملية : بعد قطف الثمار
موانع إجراء العملية : لا يوجد
تأثير إجراء العملية على الإنتاج : تحسين الصفات الكمية والنوعية للثمار
تأثير إجراء العملية على البيئة : لا يوجد
أهمية إجراء العملية : (إجبارية)- الموازنة بين كل من النمو الخضرى و الثمرى
عدد مرات إجراء العملية : مرة واحدة فى العام
:مسلسل العملية
تصنيف العملية : الأشجار المثمرة
العملية السابقة: قطف الثمار
موعد إجراء العملية : ديسمبر ويناير / يوليو وأغسطس
الوقت اللازم بين موعد إجراء العملية و العملية السابقة : مباشرة بعد قطف الثمار
فترة إجراء العملية عامل (يوم / فدان) : ١٠ عامل / فدان / يوم
المادة المستخدمة و كميتها : لا يوجد
وقت إجراء العملية خلال اليوم :طوال ساعات النهار
: الطريقة المستخدمة فى إجراء العملية : بعد جمع المحصول يتم التقليم كالآتى

7- إزالة السرطانات و الأفرخ المائية و الأفرع الجافة
8- إزالة الأجزاء الجافة من الأفرع
9- فتح قلب الشجرة بدرجة متوسطة
10- إزالة الأفرع المصابة بالحشرات والأمراض
11- قص قمم الأفرع المرتفعة بحيث لا يزيد ارتفاع الشجرة عن ٣-٣,٥ م

12- قص قمم الأفرع المتداخلة بين صفوف الأشجار لتوفير مساحة كافية بين صفوف الأشجار لإجراء عمليات الخدمة الزراعية وتوفير التهوية والإضاءة اللازمة للأشجار
13- الأشجار التى لا تقلم فى ديسمبر ويناير يمكن تقليمها فى يوليو وأغسطس
'الأداة المستخدمة فى إجراء العملية : مقص تقليم - سراق').
knowledge('\"استبدال صنف بصنف آخر فى حدائق الموالح المثمرة\"',
اسم العملية : استبدال صنف بصنف آخر فى حدائق الموالح المثمرة (الوادى / الأراضى المستصلحة)
الشروط اللازمة لإجراء العملية : ضعف إنتاج الصنف - كبر عمر الأشجار - عدم اقتصادية الصنف
موانع إجراء العملية : الإصابة بالأمراض الفيروسية و شبه الفيروسية
تأثير إجراء العملية على الإنتاج :تجديد نمو وإنتاج الأشجار وزيادته
تأثير إجراء العملية على البيئة : لا يوجد
أهمية إجراء العملية : اختيارية (لتجديد شباب الأشجار أو تغيير الصنف)
عدد مرات إجراء العملية : واحدة
مسلسل العملية:
تصنيف العملية : الأشجار المثمرة
العملية السابقة:
موعد إجراء العملية :تطعيم بالقلم (يناير وفبراير) تطعيم بالعين (مارس وابريل / سبتمبر)
الوقت اللازم بين موعد إجراء العملية و العملية السابقة : لايوجد
فترة إجراء العملية عامل (يوم / فدان) : ٦ عمال / فدان / يوم (للقرط)
6 عمال / فدان / يوم (للتطعيم)
المادة المستخدمة و كميتها : عيون طعم أو أقلام طعم من أشجار قوية عالية الجودة والإنتاج خالية من الأمراض الفيروسية
وقت إجراء العملية خلال اليوم : مبكرات الصباح أو فى فترة ما بين الظهر
الطريقة المستخدمة فى إجراء العملية :
فى حالة التطعيم بالقلم :
قرط الأشجار المراد تغيير صنفها أسفل منطقة التطعيم بحوالى ١٠سم ثم يطعم الأصل (نارنج) بأقلام الصنف الجديد حيث يتم تركيب ٣-٤ أقلام على محيط الجذع .
فى حالة التطعيم بالعين :
بعد قرط الأشجار بعد منطقة التطعيم بحوالى ١٠سم يربى ٣-٤ سرطانات (الأفرع النامية من اصل النارنج) ثم تطعيم هذه الأفرع بالعين بالصنف الجديد بعد وصولها إلى نمو ملائم للتطعيم وغالبا فى ديسمبر إذا تم القطع فى يناير أو فبراير .
و فى الحالتين يجب إزالة السرطانات التى تنمو على الأصل مع الاهتمام بخدمة الحديقة وبالتسميد والرى ومقاومة الحشائش والآفات
'الأداة المستخدمة فى إجراء العملية : سراق - مقص تقليم - مطواة تطعيم - قيرط بلاستيك').
knowledge('\"مقاومة الحشائش فى حدائق الموالح المثمرة\"',
اسم العملية : مقاومة الحشائش فى حدائق الموالح المثمرة (الوادى / الأراضى المستصلحة)
الشروط اللازمة لإجراء العملية : مراعاة حالة التزهير و العقد النمو للأشجار
موانع إجراء العملية : التزهير والعقد
تأثير إجراء العملية على الإنتاج : الحصول على إنتاج مربح
تأثير إجراء العملية على البيئة :لا يوجد
أهمية إجراء العملية : إجبارية (للتخلص من الحشائش)
عدد مرات إجراء العملية : ٤-٥ مرات فى العام
مسلسل العملية:
تصنيف العملية : الأشجار المثمرة
العملية السابقة:
موعد إجراء العملية : ديسمبر - يناير ثم من ابريل حتى أكتوبر
الوقت اللازم بين موعد إجراء العملية و العملية السابقة :
فترة إجراء العملية عامل (يوم / فدان) : ١٠- ١٥ عامل / فدان / يوم
المادة المستخدمة و كميتها : لا يوجد
وقت إجراء العملية خلال اليوم : مبكرات الصباح أو بعد فترة الظهيرة
الطريقة المستخدمة فى إجراء العملية :
5- إجراء عزقة رئيسية عميقة خلال ديسمبر أو يناير

6- إجراء عزيق سطحى للحديقة إذا دعت الحاجة خلال أشهر أبريل - مايو أو حش الحشائش و تركها فوق سطح التربة خلال هذان الشهران.
7- (إجراء عزيق سطحى فى حالة وجود حشائش خلال الفترة من يوليو و حتى أكتوبر (عزيق خربشة.
8- يمكن استخدام مبيدات الحشائش فى مقاومة الحشائش ابتداء من يوليو و حتى نهاية أكتوبر كالآتى:
أ - حشائش حولية بنوعيها عريضة و ضيقة الأوراق و يستخدم مبيد الجرامكسون بمعدل ١لتر/ ٢٠٠ لتر ماء/ فدان من ٢-٣ مرات بفاصل شهر واحد بين الرشة و الأخرى أو مبيد الباستا ٢٠ بمعدل من ٢-٤ لتر/٢٠٠لتر/ فدان دفعة واحدة (٤ شهر بين الرشة والأخرى لتر/فدان) أو على دفعتين (٢لتر/فدان) بفاصل من ١-٢ .
ب- الرجلة والحشائش الحولية العريضة يستخدم مبيد الجيسابريم ¾ كجم +٢٠٠سم جرامكسون /٢٠٠ لترماء / فدان.
ج- حشائش معمرة (نجيل- سعد-حلفا-هجنة-عليق) يستخدم راوند آب بمعدل (٢٠سم٣ +١٠جم سلفات نوشادر+١/٢سم٣ زيت طعام) /١لتر ماء على أن ترش الأماكن المصابة فقط بالمبيد ويمكن تكرار العملية مرة ثانية بعد مرور شهر من موعد الرشة الأولى و ذلك فى حالة عدم القضاء على الحشائش.
اما إذا كانت الحديقة موبوءة بالحشائش المعمرة ترش التربة بصفة عامة فى وجود الحشائش بالرواند آب بمعدل ٤لتر مبيد + ٢ كجم سلفات نوشادر + ١٠٠سم٣ زيت طعام و ذلك لكل ٢٠٠ لتر ماء/فدان.
يجب أن يسبق عملية المقاومة بالمبيدات رى الحديقة ولا تروى إلا بعد مرور ٥-٧ أيام من الرش و يكون الرش فى الصباح الباكر بعد تطاير الندى أو فى فترة ما بين الظهيرة.
'). الأداة المستخدمة فى إجراء العملية : فأس - رشاشة ظهر- غراقة
knowledge('\"العزيق فى حدائق الموالح المثمرة\"' ,'
(اسم العملية : العزيق فى حدائق الموالح المثمرة (الوادى / الأراضى المستصلحة
الشروط اللازمة لإجراء العملية : عدم إجراء العزيق أثناء مرحلة التزهير والعقد
موانع إجراء العملية : التزهير والعقد
تأثير إجراء العملية على الإنتاج : تحسين صفات الثمار و زيادة الإنتاج
تأثير إجراء العملية على البيئة : لا يوجد
أهمية إجراء العملية (إجبارية) لمقاومة الحشائش
عدد مرات إجراء العملية : ٤-٥ مرات فى العام
مسلسل العملية.
تصنيف العملية : أشجار مثمرة
العملية السابقة:رى الحديقة
(موعد إجراء العملية : عزقة رئيسية (ديسمبر / يناير) عزيق سطحى (ابريل - أكتوبر
الوقت اللازم بين موعد إجراء العملية و العملية السابقة :
فترة إجراء العملية عامل (يوم / فدان) : ١٠-١٥ عامل / فدان / يوم
المادة المستخدمة و كميتها :
وقت إجراء العملية خلال اليوم :طوال اليوم باستثناء فترة الظهيرة
الطريقة المستخدمة فى إجراء العملية :
1- عزقة رئيسية عميقة بالفأس خلال ديسمبر أو يناير
2- عزيق سطحى قبل التزهير إذا دعت الحاجة لذلك
3- عزيق خربشة أو حش الحشائش خلال الفترة من ابريل - يونيو
4- عزيق سطحى خلال الفترة من يوليو و حتى أكتوبر
5- الحدائق التى تروى بالتنقيط يكتفى بالعزيق حول الأشجار فقط أى المنطقة التى يوجد بها الحشائش
الأداة المستخدمة فى إجراء العملية : عزاقة ميكانيكية أو الفأس
').

4.3. Inference layer

File name: pc_inf.pl

```
:- ensure_loaded('$KROL/lib/krol_init').
pc_inf :: {
        input(obtain_plantation_status,[] )&
        output(obtain_plantation_status,[] )&
        input(suggest,[] )&
        output(suggest,[] )&
```

```
          input(assign,[] )&
          output(assign,[] )&
          description(obtain_plantation_status, '') &
          obtain_plantation_status :-
                    input :: run,
                    input :: tkwait&

          description(suggest, '') &
          suggest :-
                    suggestion_model :: conclude_all &

          description(assign, '') &
          assign :-
                    assignment_model :: conclude_all &
super(krol_init)
}.
```

4.4. Task layer

File name: pc_task.pl

```
%task([pc_task]).
% This is to mark that is file is generated by task editor. Please do not delete
pc_task :: {
super(krol_init)
}.
pc_task_transfer :: {
obtain_plantation_status :-
          input :: display,
          input :: tkwait &
super(pc_task)}.
pc_task_uncondional :: {
super(pc_task)}.
pc_task_condional :: {
super(pc_task)}.
pc_task_repetitive :: {
super(pc_task)}.
pc_task_user :: {
start :-
          :datime(datime(Y,M,D,_,_,_)),
          plantation :: set(current_date([D,M,Y])),
          pc_task_transfer :: obtain_plantation_status,
          suggestion_model  :: conclude_all,
          assignment_model :: conclude_all,
          :result,
          (output :: get(operList([]))->
                    (krol_msgs :: show('لا يوجد عمليات', []),
                    krol_msgs :: tkwait)
                    ;
                    (output :: display,:setvalue,output :: tkwait)
           )&
super(pc_task)
}.
```

File Name: pl_main.pl

```
:-use_module(library(system)).
:-ensure_loaded('$KROL/lib/messages').
:-ensure_loaded('$KROL/lib/database').
:-ensure_loaded('$KROL/lib/tk_user').
:-ensure_loaded('$KROL/lib/date').
:- ensure_loaded('$KROL/lib/buttonbox').
:- ensure_loaded('$KROL/lib/ComboBox').
:- ensure_loaded('$KROL/lib/frame').
:- ensure_loaded('$KROL/lib/HList').
:- ensure_loaded('$KROL/lib/labelframe').
:- ensure_loaded('$KROL/lib/messages').
:- ensure_loaded('$KROL/lib/msgs').
:- ensure_loaded('$KROL/lib/txtw').
:-ensure_loaded(plcareconcept).
:-ensure_loaded(plcarerule).
:-ensure_loaded(dialog).
:-ensure_loaded(output).
:-ensure_loaded(pc_inf).
:-ensure_loaded(methvedtext).
:-ensure_loaded(knowledge).
:-ensure_loaded(pc_task).
plant_main :-
        tcl :: init,
        krol_init :: init,
        listbox_button :: set(back(0)),
        entry_buttons ::  set(back(0)),
        utility :: restart,
        krol_init :: init,
        pc_task_user :: start.
```

4.5. User Interface

File name: dialog.pl

```
dialogue(input).
:- ensure_loaded('$KROL/lib/buttonbox').
:- ensure_loaded('$KROL/lib/labelframe').
frame1 :: {
        belong_to(input) &
        pack(['-side top -expand true -fill both']) &
        widget(frame1, ['-labelside',right], []) &
        super(labelframe)
}.
:- ensure_loaded('$KROL/lib/combobox').
'plantation-type' :: {
        belong_to(frame1) &
        widget(typeplantation, ['-label','نوع البستان','-labelside',right,'-editable',false,'-
dropdown',true], ['label.width',10,'entry.width',45,'-anchor',e,'-value',command]) &
```

```
        content(['\'اقامة بستان حديث\"'،'\'رعاية البستان\'']) &
        pack(['-anchor w']) &
        super(combobox)
}.
:- ensure_loaded('$KROL/lib/combobox').
'plantation-appearance' :: {
        belong_to(frame1) &
        widget(appearanceplantation, ['-label','الظاهره','-labelside',right,'-editable',false,'-
dropdown',true], ['label.width',10,'entry.width',45,'-anchor',e,'-value',command]) &
        content(['\' وجود اختلاف كبير فى منسوب التربة\"'،'\'استخدام الرى بالتنقيط\"'،'\'وجود ماء رى يكفى لحاجة
المؤقتات وشتلات الموالح معا\"'،'\'تم قطف الثمار\"'،'\'ضعف انتاج الصنف\"'،'\'الاشجار فى مرحلة التزهير والعقد\"'])&
        pack(['-anchor w']) &
        super(combobox)}.
buttonbox_input :: {
        belong_to(input) &
        widget(buttonbox_input, ['-orient', horizontal], []) &
        pack(['-side bottom -fill both']) &
        button(ok, ['-text','Ok','-command','buttonbox_input ::action(1)',
                        '-underline 0', '-width 10'],'<Control-o>') &
        default(ok) &
        action(1) :-
                'plantation-type'::fetch(A),plantation::set(type(A)),
                'plantation-appearance'::fetch(B),plantation::set(appearance(B)),
                input :: destroy &
        super(buttonbox)}.
input :: {
window_title('Input screen') &
widget(input, []) &
position(10, 10) &
size(300, 125) &
components([buttonbox_input,frame1,'plantation-type','plantation-appearance']) &
super(dialog)}.
```

File name: output.pl

```
opername :: {
        belong_to(output) &
        default_var(opernameV)&
        widget(opername, ['-label',' العملية الزراعية التالية هى','-state',disabled,'-labelside',
right],['label.width', 25,'entry.width', 65])&
        super(labelentry)}.

ftype :: {
        belong_to(output) &
        default_var(ftypeV)&
        widget(ftype, ['-label','              نوع البستان ','-state',disabled,'-labelside',
right],['label.width', 25,'entry.width', 65])&
        pack(['-padx 5 -pady 5']) &
        super(labelentry)}.
impor :: {
        belong_to(output) &
        default_var(imporV)&
        widget(impor, ['-label','أهمية العملية','-state',disabled,'-labelside',
right],['label.width', 25,'entry.width', 65])&
        pack(['-padx 5 -pady 5']) &
        super(labelentry)}.
mater :: {
        belong_to(output) &
        default_var(materV)&
        widget(mater, ['-label','المادة المستخدمة','-state',disabled,'-labelside',
right],['label.width', 25,'entry.width', 65])&
        pack(['-padx 5 -pady 5']) &
        super(labelentry)}.
method :: {
        belong_to(output) &
        widget(method, ['-text','طريقة التطبيق','-padx',5,'-pady',5,'-anchor',e], ['-
justify',right]) &
        pack(['-padx 5 -pady 5']) &
        super(label)}.
mtxt :: {
belong_to(output) &
widget(mtxt, ['-height', 150, '-width', 550], []) &
super(textwindow)}.
buttonbox_output :: {
        belong_to(output) &
        widget(buttonbox_output, ['-orient', horizontal], []) &
        pack(['-side bottom -fill both']) &
        button(occur, ['-text','تمت','-command','buttonbox_output ::action(occur)',
                '-underline 0', '-width 10'],'<Control-c>') &
        button(vedio, ['-text','لقطة فيديو','-command','buttonbox_output ::action(vedio)',
                '-underline 0', '-width 10'],'<Control-c>') &
        button(text, ['-text','نص مرتبط','-command','buttonbox_output ::action(text)',
                '-underline 0', '-width 10'],'<Control-c>') &
        button(know, ['-text','المعرفة الخاصة بهذه العملية','-command','buttonbox_output
::action(know)',
                '-underline 0', '-width 20'],'<Control-c>') &
        button(next, ['-text','العملية التالية','-command','buttonbox_output ::action(next)',
                '-underline 0', '-width 15'],'<Control-c>') &
        button(exit, ['-text','خروج ','-command','buttonbox_output ::action(exit)',
                '-underline 0', '-width 10'],'<Control-o>') &
        default(exit) &
        action(exit) :-
                output :: destroy &
        action(next):-
                output :: get(operList([_Name|SubList])),
                (SubList = [] -> krol_msgs :: show('لا يوجد عمليه تاليه', []);
                        (output :: set(operList(SubList)),:setvalue))&
        action(know):-
                output :: get(name(Name)),
                :knowledge(Name,Know),
                krol_msgs :: show(Know, [])&
        action(text):-
                output :: get(name(Name)),
                Name :: get(text(T)),
                :text(T,Text),
                (Text = [] ->    krol_msgs :: show('لا يوجد نص مرتبط لهذه العمليه', []) ;
                        :mplay_htm(Text))&
        action(vedio):-
                output :: get(name(Name)),
                Name :: get(video(V)),
                :video(V,Vedio),
                (Vedio = [] -> krol_msgs :: show('لا يوجد فيديو مرتبط بهذه العمليه', []) ;
                        (:([Ved|_] = Vedio) ,:mplay_mm(Ved)))&
        action(occur):-
                output :: get(name(Name)),
                Name :: set(occurrence('\"تم عملها\"')),
                suggestion_model  :: conclude_all,
                assignment_model :: conclude_all,
                :result&
        super(buttonbox)
}.
output :: {
attributes([operList(''),name('')]) &
window_title('Output Sceen') &
widget(output, []) &
position(0, 0) &
size(600, 300) &
components([buttonbox_output,opername,ftype,impor,mater,method,mtxt])&
super(dialog)
}.
result:-
        findall(X, (oper :: sub(X), X :: get(status(suggested))),SubList),
        output :: set(operList(SubList)).
setvalue:-
        output :: get(operList([Name|_SubList])),
        output :: set(name(Name)),
        plantation :: get(type(Type)),
        Name :: get(importance(Impor)),
        Name :: get(material(Mater)),
        Name :: get(method(M)),
        method(M,Method),
        opername :: set_default(Name),
        ftype :: set_default(Type),
        impor :: set_default(Impor),
        mater :: set_default(Mater),
        mtxt :: insert('                    '),
        mtxt :: delete('1.0', end),
        mtxt :: insert(Method).
mplay_htm([]).
mplay_htm([File|Tail]):-
```

```
        environ('KROL', KROL),
        format_to_chars('~w/bin/IEXPLORE.EXE ~w/multimedia/~w', [KROL, KROL,
File], CS),
        name(C, CS),
        exec(C, [null,null,null], _),
        mplay_htm(Tail).
```

4.6. Test Cases

Case 1

Input

Session date: 1/8/2000

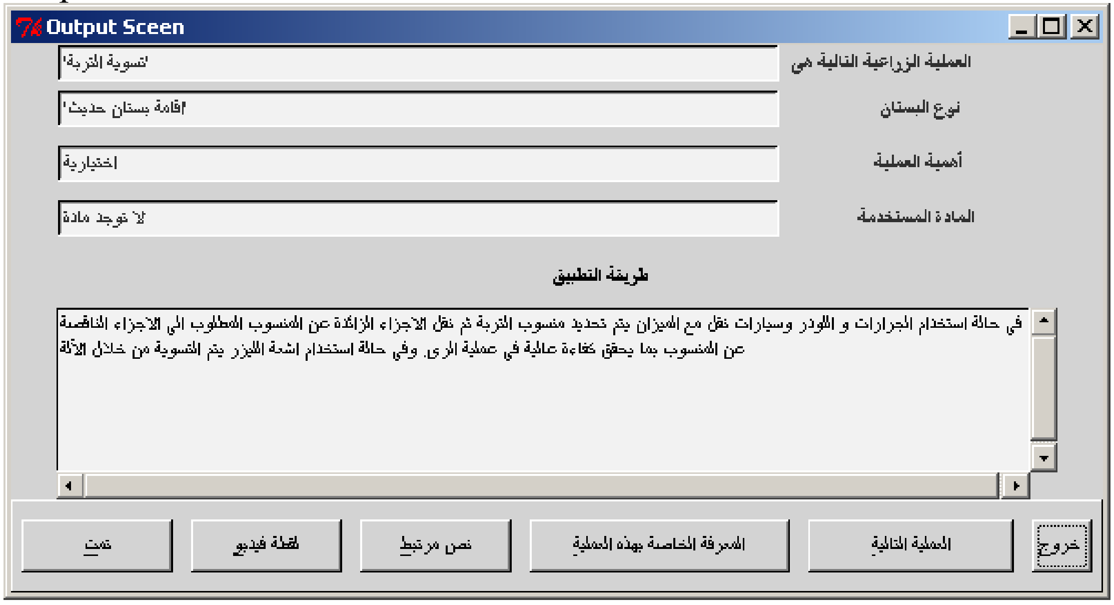
Plantation: type = إقامة بستان حديث

plantation : appearance = وجود اختلاف كبير فى منسوب التربة

تسوية التربة: occurrence = لم يتم عملها بعد

زراعه مصدات الرياح: occurrence = لم يتم عملها بعد

Output



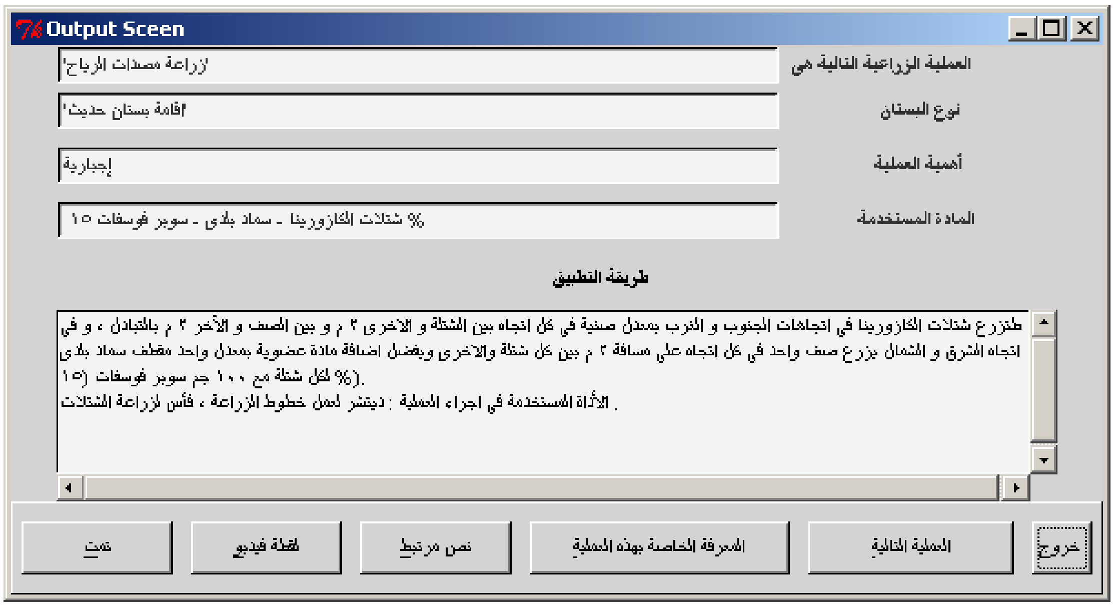
Output Sceen

العملية الزراعية التالية هي	'تسوية التربة'
نوع البستان	'اقامة بستان حديث'
أهمية العملية	اختيارية
المادة المستخدمة	لا توجد مادة

طريقة التطبيق

في حالة استخدام الجرارات و اللودر وسيارات نقل مع الميزان يتم تحديد منسوب التربة ثم نقل الاجزاء الزائدة عن المنسوب المطلوب الي الاجزاء الناقصة عن المنسوب بما يحقق كفاءة عالية في عملية الري. وفي حالة استخدام اشعة الليزر يتم التسوية من خلال الآلة

عند الضغط على " العمليه التاليه



العملية الزراعية التالية هى 'زراعة مصدات الرياح'

نوع البستان 'اقامة بستان حديث'

أهمية العملية إجبارية

المادة المستخدمة شتلات الكازورينا - سماد بلدى - سوبر فوسفات ١٥ %

طريقة التطبيق

طتزرع شتلات الكازورينا في اتجاهات الجنوب و الغرب بمعدل صنفية في كل اتجاه بين الشتلة و الاخرى ٢ م و بين الصف و الآخر ٢ م بالتبادل ، و في اتجاه الشرق و الشمال يزرع صف واحد في كل اتجاه على مسافة ٢ م بين كل شتلة والاخرى ويفضل اضافة مادة عضوية بمعدل واحد مقطف سماد بلدى لكل شتلة مع ١٠٠ جم سوبر فوسفات (١٥ %).
الأداة المستخدمة في اجراء العملية : ديفتر لعمل خطوط الزراعة ، فأس لزراعة الشتلات .

Case 2

Input

Session date: ٣٠/٩/2000

Plantation: type =إقامة بستان حديث

plantation : appearance = **وجود اختلاف كبير فى منسوب التربة**

تسوية التربة: occurrence = تم عملها

تقسيم الأرض للزراع: occurrence = تم عملها

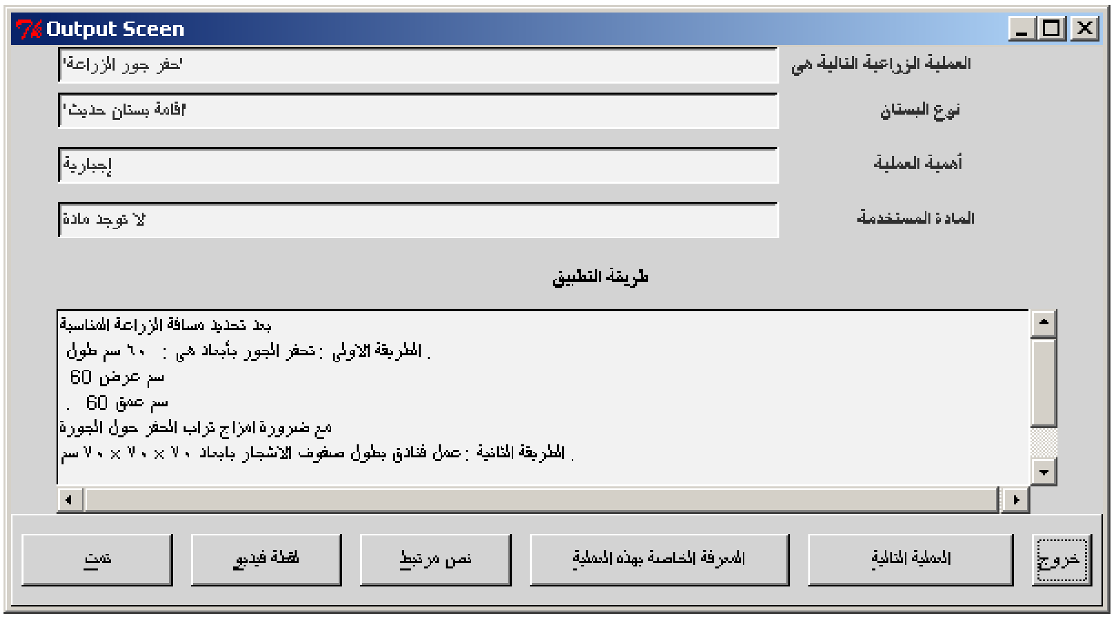
اقامة شبكة الرى بالتنقيط: occurrence = تم عملها

زراعة مصدات الرياح: occurrence = تم عملها

حفر جور الزراعة: occurrence = لم يتم عملها بعد

اضافة الأسمدة العضوية و المعدنية : occurrence = تم عملها

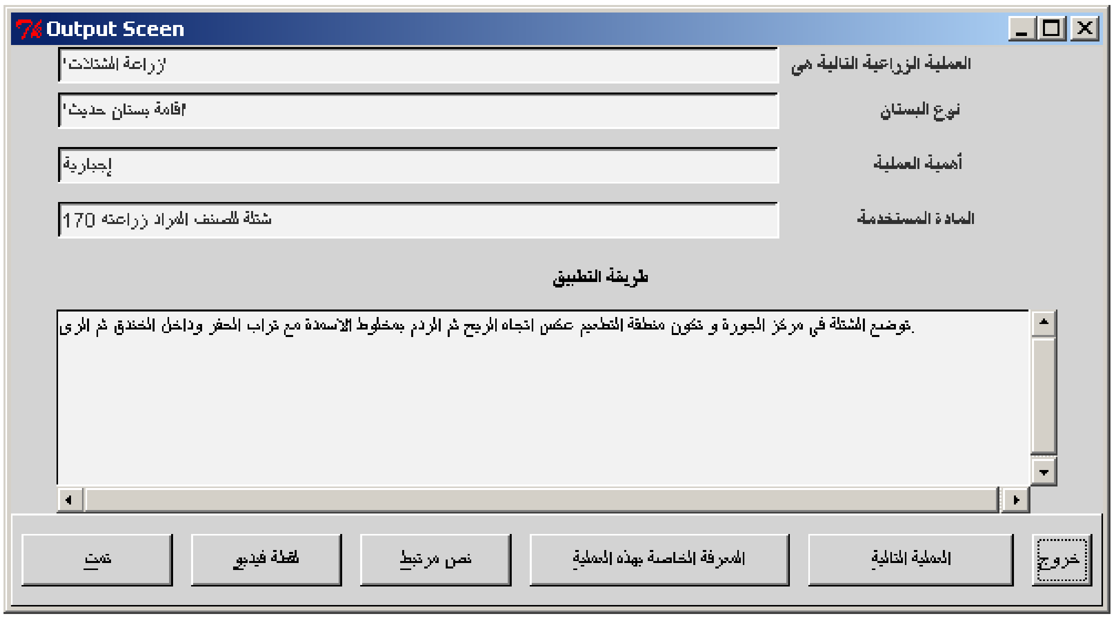
تنفيذ زراعة الشتلات: occurrence = لم يتم عملها بعد



العملية الزراعية التالية هي	'حفر جور الزراعة'
نوع البستان	'اقامة بستان حديث'
أهمية العملية	إجبارية
المادة المستخدمة	لا توجد مادة

طريقة التطبيق

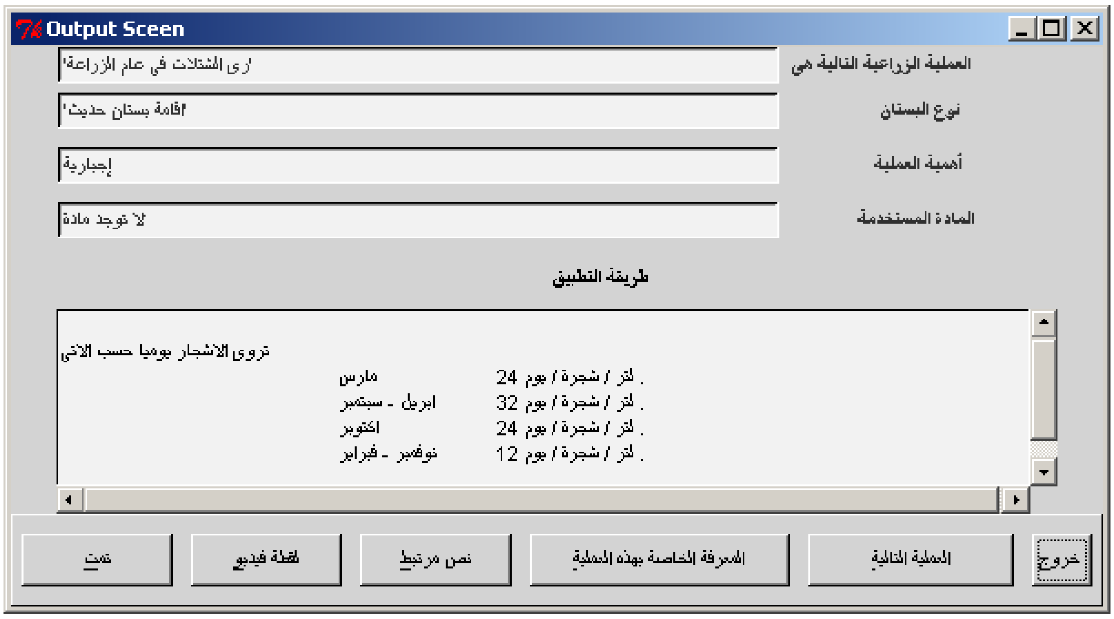
بعد تحديد مسافة الزراعة المناسبة
. الطريقة الاولى : تحفر الجور بأبعاد هي : ٦٠ سم طول
60 سم عرض
60 سم عمق .
مع ضرورة امزاج تراب الحفر حول الجورة
. الطريقة الثانية : عمل فنادق بطول صفوف الاشجار بابعاد ٧٠ × ٧٠ × ٧٠ سم



العملية الزراعية التالية هي	'زراعة المشتلات'
نوع البستان	'اقامة بستان حديث'
أهمية العملية	إجبارية
المادة المستخدمة	شتلة للصنف المراد زراعته 170

طريقة التطبيق

.توضع الشتلة في مركز الجورة و تكون منطقة التطعيم عكس اتجاه الريح ثم الردم بمخلوط الاسمدة مع تراب الحفر وداخل الخندق ثم الرى



العملية الزراعية التالية هي: 'رى الشتلات في عام الزراعة'

نوع البستان: 'اقامة بستان حديث'

أهمية العملية: إجبارية

المادة المستخدمة: لا توجد مادة

طريقة التطبيق

```
تروى الاشجار يوميا حسب الاتى
        24 لتر / شجرة / يوم .        مارس
        32 لتر / شجرة / يوم .        ابريل - سبتمبر
        24 لتر / شجرة / يوم .        اكتوبر
        12 لتر / شجرة / يوم .        نوفمبر - فبراير
```

Case 3

Input

Session date: 22/10/2000

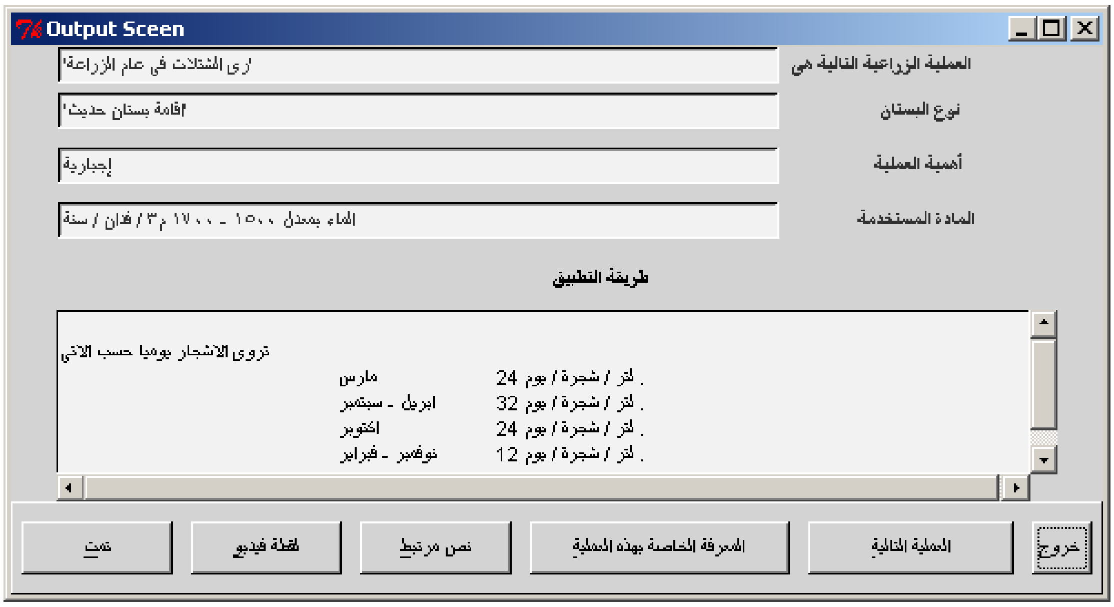
Plantation: type =إقامة بستان حديث

plantation : appearance = وجود اختلاف كبير فى منسوب التربة

اضافة الأسمدة العضوية و المعدنية: occurrence = تم عملها

زراعة الشتلات: occurrence = تم عملها

Output



العملية الزراعية التالية هي: 'رى الشتلات في عام الزراعة'

نوع البستان: 'اقامة بستان حديث'

أهمية العملية: إجبارية

المادة المستخدمة: الماء بمعدل ١٥٠٠ - ١٧٠٠ م٣ / فدان / سنة

طريقة التطبيق

```
تروى الاشجار يوميا حسب الاتى
        24 لتر / شجرة / يوم .        مارس
        32 لتر / شجرة / يوم .        ابريل - سبتمبر
        24 لتر / شجرة / يوم .        اكتوبر
        12 لتر / شجرة / يوم .        نوفمبر - فبراير
```

Case 4
Input

Session date: 22/1/2000

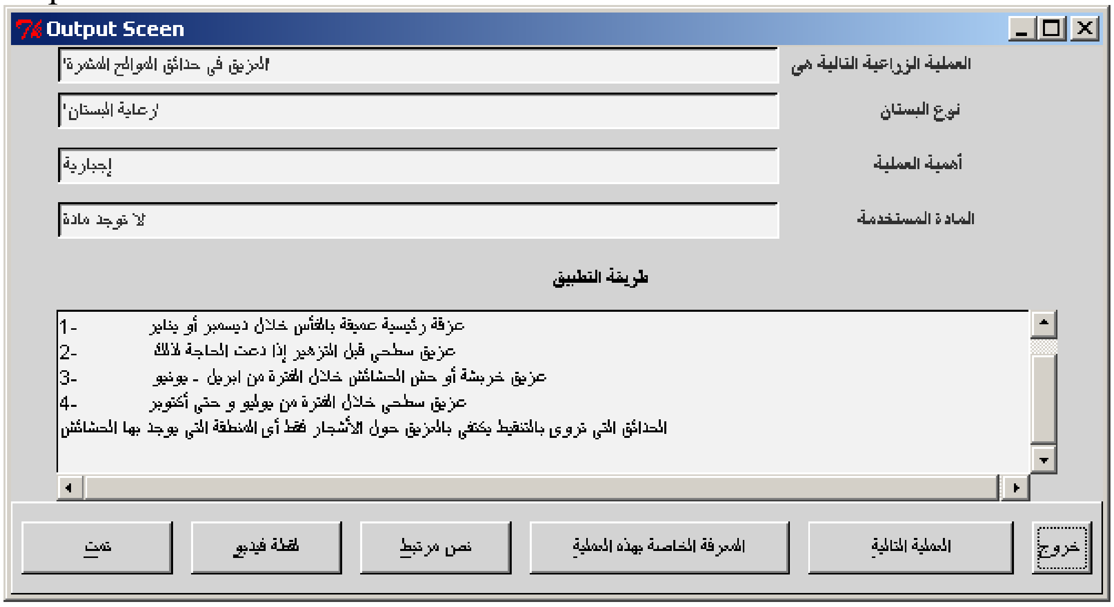
Plantation: type = رعاية البستان

plantation : appearance = وجود اختلاف كبير فى منسوب التربة

مقاومه الحشائش فى حدائق الموالح المثمره: occurrence = تم عملها

العزيق فى حدائق الموالح المثمرة: occurrence =لم يتم عملها بعد

Output


Output Sceen

العملية الزراعية التالية هي	'العزيق فى حدائق الموالح المثمرة'
نوع البستان	'رعاية البستان'
أهمية العملية	إجبارية
المادة المستخدمة	لا توجد مادة

طريقة التطبيق

```
1-        عزقة رئيسية عميقة بالفأس خلال ديسمبر أو يناير
2-        عزيق سطحي قبل التزهير إذا دعت الحاجة لذلك
3-        عزيق خربشة أو حش الحشائش خلال الفترة من ابريل - يونيو
4-        عزيق سطحي خلال الفترة من يوليو و حتى أكتوبر
الحدائق التي تروى بالتنقيط يكتفي بالعزيق حول الأشجار فقط أي المنطقة التي يوجد بها الحشائش
```

Case 5
Input

Session date: 22/12/2000

Plantation: type = رعاية البستان

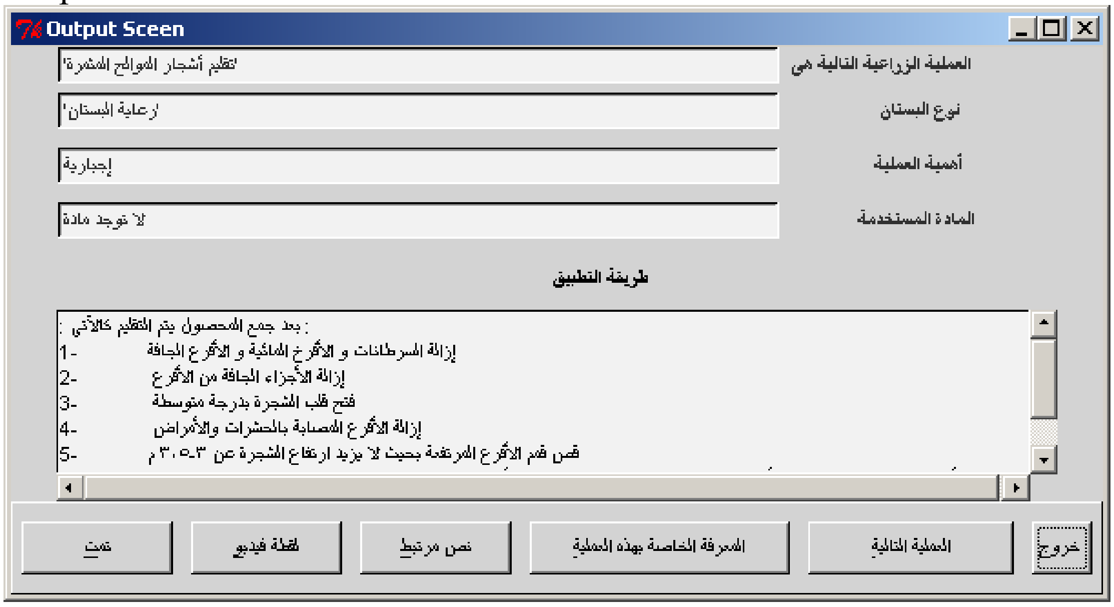
plantation : appearance = تم قطف الثمار

تقليم أشجار الموالح المثمرة: occurrence = لم يتم عملها بعد

مقاومه الحشائش فى حدائق الموالح المثمره: occurrence = تم عملها

العزيق فى حدائق الموالح المثمره: occurrence = تم عملها

Output



العملية الزراعية التالية هي: 'تقليم أشجار الموالح المثمرة'

نوع البستان: 'رعاية البستان'

أهمية العملية: إجبارية

المادة المستخدمة: لا توجد مادة

طريقة التطبيق

بعد جمع المحصول يتم التقليم كالآتي :

1- إزالة السرطانات و الأفرخ المائية و الأفرع الجافة
2- إزالة الأجزاء الجافة من الأفرع
3- فتح قلب الشجرة بدرجة متوسطة
4- إزالة الأفرع المصابة بالحشرات والأمراض
5- قص قمم الأفرع المرتفعة بحيث لا يزيد ارتفاع الشجرة عن ٣-٣.٥ م

5. Diagnosis subsystem

5.1 Relations between expressions

Notes:
The following rules are added to the confirm_disorders relation in the implemntation instead of the verify_disorder relations in the design

- Rules for armillaria_root_rot and sooty_mold in "Disorder & Observation & Variety VERIFY Disorder".
- Rules for fruit_crackin, fruit_creasing, sooty mold, gummosis, and psorosis Disorder & Plant & Observation & Variety VERIFY Disorder".
- Rules for ganoderma_rot in "Disorder & Plant & Observation VERIFY Disorder".

File name: Diag_rules.pl

```
:- use_module(library(lists), [memberchk/2]).
:- ensure_loaded('$KROL/lib/rule_exp').
caused_by_disoders :: {
r1([suspected(gummosis)in disorder,suspected(citrus_nematude)in disorder,suspected(nitrogen_def)in
disorder,suspected(potassium_def)in disorder,suspected(magnesium_def)in
disorder,suspected(manganese_def)in disorder,suspected(iron_def)in
disorder,suspected(calcium_def)in disorder,suspected(zinc_def)in disorder,suspected(salt_injury)in
disorder]) if
        l_color(yellow) in leaves &
r2([suspected(citrus_white_fly)in disorder,suspected(aphids)in disorder,suspected(mealy_bug)in
disorder]) if
        l_color(black) in leaves &
r3([suspected(phosphorus_def)in disorder,suspected(salt_injury)in disorder]) if
        l_color(purple) in leaves &
r4([suspected(psorosis)in disorder,suspected(aphids)in disorder]) if
        l_color(green) in leaves &
r5([suspected(rust_mite)in disorder]) if
        l_color(brown) in leaves &
r6([suspected(phosphorus_def)in disorder]) if
        l_color(dark_green) in leaves &
r7([suspected(potassium_def)in disorder]) if
        l_color(bronze) in leaves &
r8([suspected(iron_def)in disorder]) if
        l_color(green_network) in leaves &
r9([suspected(gummosis)in disorder]) if
        l_color(light_green) in leaves &
r10([suspected(leafminer)in disorder]) if
        l_shape(zigzag_tunnels) in leaves &
r11([suspected(scales)in disorder,suspected(leafminer)in disorder,suspected(rust_mite)in
disorder,suspected(brown_mite)in disorder,suspected(flat_mite)in
disorder,suspected(manganese_def)in disorder]) if
        existence(yes) in leaf_spots &
r12([suspected(calcium_def)in disorder]) if
        l_shape(cup_shape) in leaves &
r13([suspected(aphids)in disorder]) if
        l_shape(curled) in leaves &
r14([suspected(citrus_white_fly)in disorder,suspected(aphids)in disorder,suspected(mealy_bug)in
disorder]) if
        l_shape(honey_dew) in leaves &
r15([suspected(bud_mite)in disorder]) if
```

```
          u_color(brown) in buds,
          (         value(lime) in variety
          ;         value(navel) in variety
          ), ! &
r16([suspected(wilt_root_rot)in disorder]) if
          l_color(yellow) in leaves,
          (         value(navel) in variety
          ;         value(succar) in variety
          ;         value(valencia) in variety
          ), ! &
r17([suspected(anthracnose)in disorder,suspected(alternaria_leaves_spot)in
disorder,suspected(gum_spots)in disorder]) if
          existence(yes) in leaf_spots,
          (         value(navel) in variety
          ;         value(succar) in variety
          ;         value(valencia) in variety
          ), ! &
r18([suspected(armillaria_root_rot)in disorder]) if
          r_status('fungal growth') in roots,
          (         value(navel) in variety
          ;         value(succar) in variety
          ;         value(valencia) in variety
          ), ! &
r19([suspected(zinc_def)in disorder]) if
          age(_11364) in plant,    :(_11364>=5),
          t_shape(dwarfing) in trunk &
r20([suspected(lichens)in disorder]) if
          age(_12012) in plant,    :(_12012>=5),
          t_shape('lichen growths') in trunk &
r21([suspected(lichens)in disorder]) if
          age(_12660) in plant,    :(_12660>=5),
          b_color('spotted yellowish') in branches &
r22([suspected(gummosis)in disorder]) if
          age(_13308) in plant,    :(_13308>=5),
          t_shape('gum spots') in trunk &
r23([suspected(stubborn)in disorder]) if
          (         season(autumn) in plant
          ;         season(winter) in plant
          ), !,
          (         f_color(normal) in fruits
          ;         f_color('green styler end') in fruits
          ), ! &
r24([suspected(citrus_flower_moth)in disorder]) if
          season(spring) in plant,
          age(_15153) in plant,    :(_15153>=5),
          f_l_shape(aggregated) in flowers &
r25([suspected(citrus_flower_moth)in disorder]) if
          season(spring) in plant,
          age(_15977) in plant,    :(_15977>=5),
          l_color(geen_to_red) in leaves &
r26([suspected(rose_scarab)in disorder,suspected(citrus_flower_moth)in disorder]) if
          season(spring) in plant,
          age(_16852) in plant,    :(_16852>=5),
          f_l_shape(eaten) in flowers &
```

```
r27([suspected(potassium_def)in disorder,suspected(salt_injury)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_17967) in plant,    :(_17967>=5),
        f_shape(small) in fruits &
r28([suspected(phosphorus_def)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_19031) in plant,    :(_19031>=5),
        f_r_status('rough and thickened') in fruits &
r29([suspected(ganoderma_rot)in disorder]) if
        age(_19705) in plant,    :(_19705>=5),
        t_shape('fungal growths') in trunk,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r30([suspected(psorosis)in disorder]) if
        age(_20931) in plant,    :(_20931>=5),
        t_shape('bark scaling') in trunk,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r31([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        l_color(black) in leaves,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r32([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        b_color(black) in branches &
r33([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        f_color(black) in fruits &
```

```
r34([suspected(brown_mite)in disorder,suspected(green_stink_bug)in
disorder,suspected(impieetratura)in disorder,suspected(mediterranean_fruit_fly)in
disorder,suspected(sun_burn)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r35([suspected(fruit_creasing)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_28213) in plant,   :(_28213>=5),
        f_r_status(creasing) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r36([suspected(fruit_cracking)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_29855) in plant,   :(_29855>=5),
        f_shape(cracks) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r37([suspected(alternaria_rot)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        value(navel) in variety,
        f_color('yellow styler end') in fruits &
super(rules)
}.

confirm_disorders :: {
r1([confirmed(stubborn)in disorder]) if
        suspected(stubborn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_color('green styler end') in fruits &
r2([confirmed(citrus_flower_moth)in disorder]) if
        suspected(citrus_flower_moth) in disorder,
        season(spring) in plant,
        l_color(geen_to_red) in leaves &
r3([confirmed(citrus_flower_moth)in disorder]) if
        suspected(citrus_flower_moth) in disorder,
        season(spring) in plant,
        (       f_l_shape(aggregated) in flowers
        ;       f_l_shape(eaten) in flowers
        ), ! &
r4([confirmed(rose_scarab)in disorder]) if
        suspected(rose_scarab) in disorder,
        season(spring) in plant,
        f_l_shape(eaten) in flowers &
r5([confirmed(phosphorus_def)in disorder]) if
        suspected(phosphorus_def) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_r_status('rough and thickened') in fruits &
r6([confirmed(potassium_def)in disorder]) if
        suspected(potassium_def) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_shape(small) in fruits,
        (       f_r_status(creasing) in fruits
        ;       f_r_status(thin) in fruits
        ), ! &
r7([confirmed(salt_injury)in disorder]) if
        suspected(salt_injury) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_shape(small) in fruits &
r8([confirmed(gummosis)in disorder]) if
        suspected(gummosis) in disorder,
        age(_39646) in plant,    :(_39646>=5),
        t_shape('gum spots') in trunk,
        (       t_position('basal part') in trunk
        ;       t_position('feeder roots') in trunk
        ), ! &
r9([confirmed(gummosis)in disorder]) if
        suspected(gummosis) in disorder,
        age(_40886) in plant,    :(_40886>=5),
        (       l_color(light_green) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        l_c_position('main veins') in leaves &
r10([confirmed(lichens)in disorder]) if
        suspected(lichens) in disorder,
        age(_42102) in plant,    :(_42102>=5),
        t_shape('lichen growths') in trunk &
r11([confirmed(lichens)in disorder]) if
        suspected(lichens) in disorder,
        age(_42946) in plant,
        :(_42946>=5),
        b_color('spotted yellowish') in branches,
        b_status('gray fellvet') in branches &
r12([confirmed(mediterranean_fruit_fly)in disorder]) if
        suspected(mediterranean_fruit_fly) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        (       f_s_color(red) in fruit_spots
        ;       f_s_color(yellow) in fruit_spots
        ), !,
        f_s_position('any position') in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r13([confirmed(green_stink_bug)in disorder]) if
        suspected(green_stink_bug) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        f_s_color(yellow) in fruit_spots,
        f_s_position(scattered) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r14([confirmed(impieetratura)in disorder]) if
        suspected(impieetratura) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        f_s_color(green) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r15([confirmed(salt_injury)in disorder]) if
        suspected(salt_injury) in disorder,
        l_color(purple) in leaves &
r16([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        existence(yes) in leaf_spots,
        (       l_s_color(brown) in leaf_spots
        ;       l_s_color(rust) in leaf_spots
        ), !,
        l_s_position(scattered) in leaf_spots &
r17([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        l_color(brown) in leaves,
        (       tw_color(brown) in twigs
        ;       tw_color(rust) in twigs
        ), ! &
r18([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        (       tw_color(brown) in twigs
        ;       tw_color(rust) in twigs
        ), !,
        existence(yes) in leaf_spots &
r19([confirmed(brown_mite)in disorder]) if
        suspected(brown_mite) in disorder,
        existence(yes) in fruit_spots,
        f_s_color(brown) in fruit_spots,
        f_s_position('stiller and stem ends') in fruit_spots &
r20([confirmed(brown_mite)in disorder]) if
        suspected(brown_mite) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(dusty) in leaf_spots,
        l_s_position('midrib upper surface') in leaf_spots &
r21([confirmed(flat_mite)in disorder]) if
        suspected(flat_mite) in disorder,
        (       l_s_color(brown) in leaf_spots
        ;       l_s_color(silver) in leaf_spots
        ), !,
        l_s_shap(sunken) in leaf_spots,
        l_s_position('between veins of lower surface') in leaf_spots &
r22([confirmed(citrus_nematude)in disorder]) if
        suspected(citrus_nematude) in disorder,
        l_color(yellow) in leaves,
        l_c_position(entire_leaf) in leaves,
        b_status('die back') in branches,
        b_type(flushes) in branches &
r23([confirmed(leafminer)in disorder]) if
        suspected(leafminer) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(silver) in leaf_spots &
r24([confirmed(leafminer)in disorder]) if
        suspected(leafminer) in disorder,
        l_shape(zigzag_tunnels) in leaves &

r25([confirmed(aphids)in disorder]) if
        suspected(aphids) in disorder,
        (       l_shape(curled) in leaves
        ;       l_shape(honey_dew) in leaves
        ), !,
        l_status(insect_persent) in leaves,
        l_type(new_leaves) in leaves &

r26([confirmed(aphids)in disorder]) if
        suspected(aphids) in disorder,
        l_status(insect_persent) in leaves,
        l_type(new_leaves) in leaves,
        (       l_color(black) in leaves
        ;       l_color(green) in leaves
        ), ! &

r27([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_color(black) in leaves,
        l_status(insect_persent) in leaves &
```

```
r28([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_shape(honey_dew) in leaves,
        l_status(insect_persent) in leaves &
r29([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_color(black) in leaves,
        l_shape(honey_dew) in leaves,
        l_c_position('upper surface') in leaves &
r30([confirmed(scales)in disorder]) if
        suspected(scales) in disorder,
        existence(yes) in leaf_spots,
        (       l_s_color(black) in leaf_spots
        ;       l_s_color(yellow) in leaf_spots
        ), !,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        l_status(insect_persent) in leaves &
r31([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_color(black) in leaves,
        l_status(insect_persent) in leaves,
        l_type(old_leaves) in leaves,
        b_status('insect present') in branches &
r32([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_status(insect_persent) in leaves,
        l_type(old_leaves) in leaves,
        b_status('insect present') in branches,
        l_shape(honey_dew) in leaves &
r33([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_color(black) in leaves,
        l_shape(honey_dew) in leaves &
r34([confirmed(iron_def)in disorder]) if
        suspected(iron_def) in disorder,
        (       l_color(green_network) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        (       l_c_position(entire_leaf) in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves &
r35([confirmed(manganese_def)in disorder]) if
        suspected(manganese_def) in disorder,
        l_color(yellow) in leaves,
        l_c_position('between veins') in leaves,
        l_type(new_leaves) in leaves &
r36([confirmed(manganese_def)in disorder]) if
        suspected(manganese_def) in disorder,
        existence(yes) in leaf_spots,
        l_s_position('between veins') in leaf_spots &
r37([confirmed(nitrogen_def)in disorder]) if
```

```
        suspected(nitrogen_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position(entire_leaf) in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(old_leaves) in leaves &
r38([confirmed(potassium_def)in disorder]) if
        suspected(potassium_def) in disorder,
        (       l_color(bronze) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position('leaf tip') in leaves
        ), !,
        l_type(old_leaves) in leaves &
r39([confirmed(zinc_def)in disorder]) if
        suspected(zinc_def) in disorder,
        l_color(yellow) in leaves,
        l_c_position('between veins') in leaves,
        l_type(new_leaves) in leaves,
        b_status(stunted) in branches,
        l_shape(unsimilar_blade_halves) in leaves &
r40([confirmed(alternaria_leaves_spot)in disorder,confirmed(zinc_def)in disorder]) if
        (       suspected(alternaria_leaves_spot) in disorder
        ;       suspected(zinc_def) in disorder
        ), !,
        t_shape(dwarfing) in trunk,
        b_status(stunted) in branches,
        l_shape(unsimilar_blade_halves) in leaves &
r41([confirmed(alternaria_leaves_spot)in disorder,confirmed(phosphorus_def)in disorder]) if
        (       suspected(alternaria_leaves_spot) in disorder
        ;       suspected(phosphorus_def) in disorder
        ), !,
        suspected(phosphorus_def) in disorder,
        (       l_color(dark_green) in leaves
        ;       l_color(purple) in leaves
        ), !,
        (       l_c_position('leaf tip') in leaves
        ;       l_c_position('lower surface') in leaves
        ;       l_c_position('outer edge') in leaves
        ), !,
        (       l_type(new_leaves) in leaves
        ;       l_type(old_leaves) in leaves
        ), ! &
r42([confirmed(calcium_def)in disorder]) if
        suspected(calcium_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves &
r43([confirmed(calcium_def)in disorder]) if
        suspected(calcium_def) in disorder,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves,
        l_shape(cup_shape) in leaves &
r44([confirmed(magnesium_def)in disorder]) if
        suspected(magnesium_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position('between veins') in leaves
        ;       l_c_position(inverted_v) in leaves
        ;       l_c_position('leaf base') in leaves
        ;       l_c_position('outer edge') in leaves
        ), !,
        l_type(old_leaves) in leaves &
r45([confirmed(bud_mite)in disorder]) if
        suspected(bud_mite) in disorder,
        (       value(lime) in variety
        ;       value(navel) in variety
        ), !,
        u_color(brown) in buds,
        u_status(abnormal) in buds &
r46([confirmed(bud_mite)in disorder]) if
        suspected(bud_mite) in disorder,
        (       value(lime) in variety
        ;       value(navel) in variety
        ), !,
        u_color(brown) in buds,
        (       u_shape(deformed) in buds
        ;       u_shape(rosette) in buds
        ), ! &
r47([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        value(lime) in variety,
        existence(yes) in fruit_spots,
        f_s_color(sliver) in fruit_spots,
        f_s_shape(coarse) in fruit_spots &
r48([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        value(lime) in variety,
        l_color(brown) in leaves,
        f_shape(coarse) in fruits &
r49([confirmed(alternaria_rot)in disorder]) if
        suspected(alternaria_rot) in disorder,
        value(navel) in variety,
        f_color('yellow styler end') in fruits &
r50([confirmed(sun_burn)in disorder]) if
        suspected(sun_burn) in disorder,
        existence(yes) in fruit_spots,
        f_s_color(brown) in fruit_spots,
        f_s_position('fruits facing the sun') in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r51([confirmed(wilt_root_rot)in disorder]) if
```

```
        suspected(wilt_root_rot) in disorder,
        l_color(yellow) in leaves,
        l_status(wilted) in leaves,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r52([confirmed(alternaria_leaves_spot)in disorder]) if
        suspected(alternaria_leaves_spot) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(yellow) in leaf_spots,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position(scattered) in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r53([confirmed(anthracnose)in disorder]) if
        suspected(anthracnose) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(yellow) in leaf_spots,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position(scattered) in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r54([confirmed(gum_spots)in disorder]) if
        suspected(gum_spots) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(brown) in leaf_spots,
        l_s_position('lower surface') in leaf_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r55([confirmed(sooty_mold)in disorder]) if
        suspected(sooty_mold) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_119609) in plant,   :(_119609>=5),
        f_color(black) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r56([confirmed(armillaria_root_rot)in disorder]) if
        suspected(armillaria_root_rot) in disorder,
        r_status('fungal growth') in roots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r57([confirmed(sooty_mold)in disorder]) if
        suspected(sooty_mold) in disorder,
        l_color(black) in leaves,
        l_status(insect_persent) in leaves,
        b_status('insect present') in branches,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r58([confirmed(sooty_mold)in disorder]) if
        suspected(sooty_mold) in disorder,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        b_color(black) in branches &
r59([confirmed(fruit_cracking)in disorder]) if
        suspected(fruit_cracking) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_115973) in plant,   :(_115973>=5),
        f_shape(cracks) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r60([confirmed(fruit_creasing)in disorder]) if
        suspected(fruit_creasing) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_117791) in plant,   :(_117791>=5),
        f_r_status(creasing) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r61([confirmed(gummosis)in disorder]) if
        suspected(gummosis) in disorder,
        age(_121035) in plant,   :(_121035>=5),
        t_shape('gum spots') in trunk,
        (       t_position('basal part') in trunk
        ;       t_position('feeder roots') in trunk
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r62([confirmed(psorosis)in disorder]) if
```

```
        suspected(psorosis) in disorder,
        age(_122829) in plant,   :(_122829>=5),
        t_shape('bark scaling') in trunk,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r63([confirmed(ganoderma_rot)in disorder]) if
        suspected(ganoderma_rot) in disorder,
        age(_114707) in plant,   :(_114707>=5),
        t_shape('fungal growths') in trunk &

super(rules)
}.
verify_disorders :: {
r1([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_94978) in plant,    :(_94978>=5),
        (       f_color(purple) in fruits
        ;       f_color(rust) in fruits
        ), !,
        f_r_status(rough) in fruits &
r2([highly_confirmed(flat_mite)in disorder]) if
        confirmed(flat_mite) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_96682) in plant,    :(_96682>=5),
        existence(yes) in fruit_spots,
        (       f_s_color(bronze) in fruit_spots
        ;       f_s_color(brown) in fruit_spots
        ;       f_s_color('scabby patches') in fruit_spots
        ;       f_s_color(sliver) in fruit_spots
        ), !,
        f_s_shape(irregular) in fruit_spots,
        (       f_s_position(rind) in fruit_spots
        ;       f_s_position(scattered) in fruit_spots
        ), ! &
r3([highly_confirmed(flat_mite)in disorder]) if
        confirmed(flat_mite) in disorder,
        season(spring) in plant,
        age(_98942) in plant,    :(_98942>=5),
        fl_status(drop) in flowers &
r4([highly_confirmed(stubborn)in disorder]) if
        confirmed(stubborn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_100202) in plant,   :(_100202>=5),
        f_shape(asymatric) in fruits,
        f_r_status(irregular) in fruits &
```

```
r5([highly_confirmed(alternaria_rot)in disorder]) if
        confirmed(alternaria_rot) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_101598) in plant,  :(_101598>=5),
        f_r_status(drop) in fruits &
r6([highly_confirmed(sun_burn)in disorder]) if
        confirmed(sun_burn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_102838) in plant,  :(_102838>=5),
        f_r_status(leathery) in fruits &
r7([highly_confirmed(mediterranean_fruit_fly)in disorder]) if
        confirmed(mediterranean_fruit_fly) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_104082) in plant,  :(_104082>=5),
        (       f_s_shape(circular) in fruit_spots
        ;       f_s_shape('large and circular') in fruit_spots
        ), ! &
r8([highly_confirmed(green_stink_bug)in disorder]) if
        confirmed(green_stink_bug) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_105558) in plant,  :(_105558>=5),
        f_s_shape(irregular) in fruit_spots &
r9([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_106798) in plant,  :(_106798>=5),
        f_s_shape('zigzag tunnels') in fruit_spots &
r10([highly_confirmed(impieetratura)in disorder]) if
        confirmed(impieetratura) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_108038) in plant,  :(_108038>=5),
        (f_s_shape('gum pocket') in fruit_spots
        ;f_s_shape(raised) in fruit_spots )
        &
r11([highly_confirmed(citrus_flower_moth)in disorder]) if
        confirmed(citrus_flower_moth) in disorder,
        age(_108866) in plant,  :(_108866>=5),
        (       fl_color(brown) in flowers
        ;       fl_color(yellow) in flowers
        ), ! &
r12([highly_confirmed(citrus_flower_moth)in disorder]) if
        confirmed(citrus_flower_moth) in disorder,
        age(_109926) in plant,   :(_109926>=5),
        tw_shape(eaten) in twigs &
r13([highly_confirmed(phosphorus_def)in disorder]) if
        confirmed(phosphorus_def) in disorder,
        season(spring) in plant,
        tw_status(dieback) in twigs &
r14([highly_confirmed(rose_scarab)in disorder]) if
        confirmed(rose_scarab) in disorder,
        season(spring) in plant,
        age(_111643) in plant,   :(_111643>=5),
        fl_status(drop) in flowers &
r15([highly_confirmed(sun_burn)in disorder]) if
        confirmed(sun_burn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_112903) in plant,   :(_112903>=5),
        f_s_shape(circular) in fruit_spots,
        f_r_status(leathery) in fruits &
r16([highly_confirmed(lichens)in disorder]) if
        confirmed(lichens) in disorder,
        age(_113883) in plant,   :(_113883>=5),
        t_shape('lichen growths') in trunk &

r23([highly_confirmed(wilt_root_rot)in disorder]) if
        confirmed(wilt_root_rot) in disorder,
        (       r_color(black) in roots
        ;       r_color(brown) in roots
        ), ! &
r24([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        b_color(rust) in branches,
        b_type(flushes) in branches &
r25([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        b_status(decline) in branches &
r26([highly_confirmed(brown_mite)in disorder]) if
        confirmed(brown_mite) in disorder,
        l_shape(webbed) in leaves &
r27([highly_confirmed(brown_mite)in disorder]) if
        confirmed(brown_mite) in disorder,
        b_color(pale) in branches &
r28([highly_confirmed(bud_mite)in disorder]) if
        confirmed(bud_mite) in disorder,
        (       b_status(flattened) in branches
        ;       b_status(stunted) in branches
        ;       b_status(thickened) in branches
        ), !,
        b_type(flushes) in branches &
r29([highly_confirmed(bud_mite)in disorder]) if
        confirmed(bud_mite) in disorder,
        f_shape(malformed) in fruits &
r30([highly_confirmed(citrus_nematude)in disorder]) if
        confirmed(citrus_nematude) in disorder,
        r_color(brown) in roots,
        (       r_status(adhesive) in roots
        ;       r_status(sloughing) in roots
        ), !,
        r_type('feeder roots') in roots &
r31([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        l_s_shap('zigzag tunnels') in leaf_spots &
r32([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        l_shape(zigzag_tunnels) in leaves &
r33([highly_confirmed(aphids)in disorder]) if
        confirmed(aphids) in disorder,
        (       i_color(black) in insects
        ;       i_color(green) in insects
        ), !,
        (       i_status(aggregated) in insects
        ;       i_status(stationary) in insects
        ), ! &
r34([highly_confirmed(gummosis)in disorder]) if
        confirmed(gummosis) in disorder,
        b_status('die back') in branches &
r35([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        tw_status(dieback) in twigs &
r36([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        f_shape(small) in fruits &
r37([highly_confirmed(manganese_def)in disorder]) if
        confirmed(manganese_def) in disorder,
        tw_status(dieback) in twigs &
r38([highly_confirmed(potassium_def)in disorder]) if
        confirmed(potassium_def) in disorder,
        tw_status(dieback) in twigs &
r39([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        l_status(small) in leaves &
r40([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        tw_status(dieback) in twigs &
r41([highly_confirmed(iron_def)in disorder]) if
        confirmed(iron_def) in disorder,
        f_r_status(reduced) in fruits,
        f_shape(small) in fruits &
r42([highly_confirmed(citrus_white_fly)in disorder]) if
        confirmed(citrus_white_fly) in disorder,
        i_color(white) in insects,
        i_status(flying) in insects &
r43([highly_confirmed(mealy_bug)in disorder]) if
        confirmed(mealy_bug) in disorder,
        i_color(white) in insects,
        i_status(stationary) in insects &
r44([highly_confirmed(anthracnose)in disorder]) if
        confirmed(anthracnose) in disorder,
        b_color(brown) in branches,
        b_status(dry) in branches &
r45([highly_confirmed(anthracnose)in disorder]) if
        confirmed(anthracnose) in disorder,
        l_s_shap(necrotic) in leaf_spots &
r46([highly_confirmed(alternaria_leaves_spot)in disorder]) if
        confirmed(alternaria_leaves_spot) in disorder,
        l_s_shap('concentric zones') in leaf_spots &
r47([highly_confirmed(scales)in disorder]) if
        confirmed(scales) in disorder,
        i_status(stucked) in insects,
        (         i_color(black) in insects
        ;         i_color(purple) in insects
        ;         i_color(red) in insects
        ;         i_color(white) in insects
        ), ! &
r48([highly_confirmed(scales)in disorder]) if
        confirmed(scales) in disorder,
        (         f_shape(malformed) in fruits
        ;         f_shape(small) in fruits
        ), ! &
r52([highly_confirmed(iron_def)in disorder]) if
        confirmed(iron_def) in disorder,
        ph(_16825) in soil,        :(_16825<8.5),
        ca_carbonate(_17096) in soil,    :(_17096<10),
        iron_def_sp('most trees') in disorder &
r53([highly_confirmed(manganese_def)in disorder]) if
        confirmed(manganese_def) in disorder,
        ph(_17944) in soil,        :(_17944<8.5),
        ca_carbonate(_18215) in soil,    :(_18215<10),
        manganese_def_sp('most trees') in disorder &
r54([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        ph(_19063) in soil,        :(_19063<8.5),
        ca_carbonate(_19334) in soil,    :(_19334<10),
        zinc_def_sp('most trees') in disorder &
r55([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        water_table_level(_20162) in soil,        :(_20162<1.5),
        nitrogen_def_sp('most trees') in disorder &
r56([highly_confirmed(salt_injury)in disorder]) if
        confirmed(salt_injury) in disorder,
        ec(_20994) in soil,        :(_20994>=2),
        salt_injury_sp('most trees') in disorder &
r57([highly_confirmed(salt_injury)in disorder]) if
        confirmed(salt_injury) in disorder,
        eciw(_21818) in water,  :(_21818>=1),
        salt_injury_sp('most trees') in disorder &
r58([highly_confirmed(aphids)in disorder]) if
        confirmed(aphids) in disorder,
        season(spring) in plant &
r59([highly_confirmed(citrus_nematude)in disorder]) if
        confirmed(citrus_nematude) in disorder,
        season(spring) in plant &
```

```
r60([highly_confirmed(magnesium_def)in disorder]) if
        confirmed(magnesium_def) in disorder,
        eciw(_23744) in water,  :(_23744<1),
        ec(_24007) in soil,     :(_24007<2),
        magnesium_def_sp('most trees') in disorder &
r61([highly_confirmed(calcium_def)in disorder]) if
        confirmed(calcium_def) in disorder,
        eciw(_24851) in water,  :(_24851<1),
        ec(_25114) in soil,     :(_25114<2),
        calcium_def_sp('most trees') in disorder &
r62([highly_confirmed(potassium_def)in disorder]) if
        confirmed(potassium_def) in disorder,
        eciw(_25958) in water,  :(_25958<1),
        ec(_26221) in soil,     :(_26221<2),
        potassium_def_sp('most trees') in disorder &
super(rules)
}.
```

File name: diag_table.pl

```
:-ensure_loaded('$KROL/lib/tab').
plant_determine_plant :: {
        p([plant-current_month] , [plant-season]) &
         t(1,winter) &
         t(2,winter) &
         t(3,spring) &
         t(4,spring) &
         t(5,spring) &
         t(6,summer) &
         t(7,summer) &
         t(8,summer) &
         t(9,autumn) &
         t(10,autumn) &
         t(11,winter) &
         t(12,winter) &
        super(table)
}.
```

5.2 Inference layer

File name : diag_inference.pl

```
:- ensure_loaded('$KROL/lib/krol_init').
diag_inference :: {
        input(determine,[] )&
        output(determine,[] )&
        input(predict,[branches-b_color,buds-u_color,flowers-f_l_shape,fruit_spots-
existence,fruits-f_color,fruits-f_r_status,fruits-f_shape,leaf_spots-existence,leaves-
l_color,leaves-l_shape,plant-age,plant-season,roots-r_status,trunk-t_shape,variety-value] )&
        output(predict,[disorder-subbected] )&
        input(confirm,[branches-b_color,branches-b_ststus,branches-b_type,buds-
u_color,buds-u_shape,buds-u_status,disorder-subbected,flowers-f_l_shape,fruit_spots-
existence,fruit_spots-f_s_color,fruit_spots-f_s_position,fruit_spots-f_s_shape,fruits-
f_color,fruits-f_r_status,fruits-f_shape,leaf_spots-existence,leaf_spots-l_s_color,leaf_spots-
l_s_position,leaf_spots-l_s_shap,leaves-l_c_position,leaves-l_color,leaves-l_shape,leaves-
l_status,leaves-l_type,plant-age,plant-season,trunk-t_position,trunk-t_shape,twigs-
tw_color,variety-value] )&
        output(confirm,[disorder-confirmed] )&
        input(verify,[branches-b_color,branches-b_ststus,branches-b_type,disorder-
calcium_def_sp,disorder-confirmed,disorder-iron_def_sp,disorder-
magnesium_def_sp,disorder-manganese_def_sp,disorder-nitrogen_def_sp,disorder-
potassium_def_sp,disorder-salt_injury_sp,disorder-zinc_def_sp,flowers-fl_color,flowers-
fl_status,fruit_spots-existence,fruit_spots-f_s_color,fruit_spots-f_s_position,fruit_spots-
f_s_shape,fruits-f_color,fruits-f_r_status,fruits-f_shape,insects-i_color,insects-
i_status,leaf_spots-l_s_shap,leaves-l_color,leaves-l_shape,leaves-l_status,plant-age,plant-
season,roots-r_color,roots-r_status,roots-r_type,soil-ca_carbonate,soil-ec,soil-ph,soil-
water_table_level,trunk-t_position,trunk-t_shape,twigs-tw_shape,twigs-tw_status,variety-
value,water-eciw] )&
        output(verify,[disorder-highly_confirmed] )&
        description(determine, '') &
        determine :-
                plant_determine_plant :: table &
        description(predict, '') &
        predict :-
                caused_by_disoders :: conclude_all &

        description(confirm, '') &
        confirm :-
                confirm_disorders :: conclude_all &
        description(verify, '') &
        verify :-
                verify_disorders :: conclude_all &
super(krol_init)
}.
```

5.3 Task layer

File name : diag_task.pl

```
task([diag_task]).      % This is to mark that is file is generated by task editor. Please do not
delete
diag_task :: {
super(krol_init)
}.
diag_task_transfer :: {
super(diag_task)
}.
diag_task_uncondional :: {
start_inference :-
        diag_task_user :: init_inf,
        diag_task_user :: determine_exist,
        diag_task_user :: determine_age,
        diag_task_condional :: plantation_not_exist &

super(diag_task)
}.
```

```
diag_task_condional :: {
plantation_not_exist :-
        (       plantation :: get_value(existence(_9824)),
                :(_9824=no) ->
                        diag_task_user :: no_need_for_diag  ;
                        (diag_inference :: determine,
                        :citex_diag_dlg)
        ) &
super(diag_task)
}.
diag_task_repetitive :: {
super(diag_task)
}.
diag_task_user :: {
determine_exist :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        :(Date = [D, M, Y]),
        :current_week(Date, W),
        plantation :: set(current_date(Date)),
        plant :: set(current_date(Date)),
        plant :: set(current_month(M)),
        plant :: set(current_week(W)),
        (:compare_date(=<, [PD,PM,PY],[D,M,Y]) ->
                plantation :: set(existence(yes))
        ;       plantation :: set(existence(no))
        ) &
no_need_for_diag :-
krol_msgs :: show("There is no plantation exists to be diagnose ....",[])&
init_inf :-
        krol_init :: init,
        utility :: restart &
determine_age :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        :dif([PD,PM,PY],[D,M,Y],[_,_,Age]),
        plant :: set(age(Age)) &
super(diag_task)
}.
```

The following is the main file for both the diagnosis and treatment subsystems

File name: diag_system.pl

```
:-use_module(library(system)).
:-ensure_loaded('$KROL/lib/messages').
:-ensure_loaded('$KROL/lib/database').
:-ensure_loaded('$KROL/lib/tk_user').
```

```
:-ensure_loaded('$KROL/lib/date').
:-ensure_loaded(c_concept).
:-ensure_loaded('citex4.pl').
:-ensure_loaded(diag_rules).
:-ensure_loaded(diag_table).
:-ensure_loaded(season_dis).
:-ensure_loaded(citex_diag_dlg).
:-ensure_loaded(diag_task).
:-ensure_loaded(treat_task).
:-ensure_loaded(treat_rules).
:-ensure_loaded(treat_dlg).
:-ensure_loaded(order).
:-ensure_loaded(treat_inference).
:-ensure_loaded(diag_inference).
diag_start :-
        krol_init :: init,
        citex4ds :: open,
        select_table :: fetch([[SN,GN,DN,FN]]),
        farm_data :: set(sid(SN)),
        farm_data :: set(gid(GN)),
        farm_data :: set(did(DN)),
        farm_data :: set(fid(FN)),
        diag_task_uncondional :: start_inference,
        citex4ds :: close.
diag_main :-
        diag_start.
treat_main:-
        diag_start.
```

5.4 User Interface

File name : citex_diag_dlg.pl

```
:- ensure_loaded('$KROL/lib/buttonbox').
:- ensure_loaded('$KROL/lib/ComboBox').
:- ensure_loaded('$KROL/lib/frame').
:- ensure_loaded('$KROL/lib/HList').
:- ensure_loaded('$KROL/lib/labelframe').
:- dynamic prop/2.
:- dynamic val/3.
:- dynamic finding/4.
:- dynamic prop_type/4.
citex_diag_dlg :-
        krol_init :: set(mode(un)),
        tcl :: eval(['proc get_disorders {args}',br([prolog,dq(get_disorders)])]),
        tcl :: eval(['proc show_properties {args}',br([prolog,dq(show_properties)])]),
        tcl :: eval(['proc show_values {args}',br([prolog,dq(show_values)])]),
        citex_diag_dlg :: run,
        init_disorders,
        citex_diag_dlg :: tkwait.
citex_diag_dlg :: {
widget(citex_diag_dlg, []) &
window_title(Title):-
        (appl_pdw :: get(sys(diag)) ->          (!,Title = 'Citex Diagnosis');
                Title = 'Citex Diagnosis & Treatment'
        ) &
components(Xs) :- self(D), :findall(X, D :: cs(_, X), Xs) &
pack(citex_diag_frm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_frm, citex_diag_dlg) &
pack(citex_diag_ses_all_lft_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_ses_all_lft_frm, citex_diag_frm) &
pack(citex_diag_ses_all_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_ses_all_frm, citex_diag_ses_all_lft_frm) &
pack(citex_diag_all_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_all_lblfrm, citex_diag_ses_all_frm) &
pack(citex_diag_all_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_all_hlst, citex_diag_all_lblfrm) &
pack(citex_diag_down_lft_btn, ['-side',bottom,'-fill',both,'-anchor',s]) &
c(citex_diag_down_lft_btn, citex_diag_all_lblfrm) &
pack(citex_diag_dwn_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_dwn_lblfrm, citex_diag_ses_all_frm) &
pack(citex_diag_dwn_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_dwn_hlst, citex_diag_dwn_lblfrm) &
pack(citex_diag_del_btn, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_del_btn, citex_diag_dwn_lblfrm) &
pack(citex_diag_left_btncitex_diag_dlg, ['-side',right,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_left_btncitex_diag_dlg, citex_diag_ses_all_lft_frm) &
pack(citex_diag_conc_prop_val_fin_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_conc_prop_val_fin_frm, citex_diag_frm) &
pack(citex_diag_conc_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_conc_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_concept_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_concept_hlst, citex_diag_conc_lblfrm) &
pack(citex_diag_Property_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_Property_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_property_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_property_hlst, citex_diag_Property_lblfrm) &
pack(citex_diag_value_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_value_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_value_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_value_hlst, citex_diag_value_lblfrm) &
pack(citex_diag_down_btncitex_diag_dlg, ['-side',bottom,'-fill',both,'-anchor',s]) &
c(citex_diag_down_btncitex_diag_dlg, citex_diag_value_lblfrm) &
pack(citex_diag_finding_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_finding_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_finding_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_finding_hlst, citex_diag_finding_lblfrm) &
pack(citex_diag_why_what_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-
anchor',s]) &
```

```
c(citex_diag_why_what_btncitex_diag_dlg, citex_diag_finding_lblfrm) &
pack(citex_diag_rgt_sus_conf_hi_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_rgt_sus_conf_hi_frm, citex_diag_frm) &
pack(citex_diag_3_rgt_btncitex_diag_dlg, ['-side',left,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_3_rgt_btncitex_diag_dlg, citex_diag_rgt_sus_conf_hi_frm) &
pack(citex_diag_sus_conf_hi_frm, ['-side',right,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_sus_conf_hi_frm, citex_diag_rgt_sus_conf_hi_frm) &
pack(citex_diag_sus_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_sus_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_sus_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_sus_hlst, citex_diag_sus_lblfrm) &
pack(citex_diag_how_sus_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-
anchor',s]) &
c(citex_diag_how_sus_btncitex_diag_dlg, citex_diag_sus_lblfrm) &
pack(citex_diag_confirm_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_confirm_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_conf_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_conf_hlst, citex_diag_confirm_lblfrm) &
pack(citex_diag_how_conf_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-
anchor',s]) &
c(citex_diag_how_conf_btncitex_diag_dlg, citex_diag_confirm_lblfrm) &
pack(citex_diag_hi_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_hi_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_hi_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_hi_hlst, citex_diag_hi_lblfrm) &
pack(citex_diag_how_hi_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-
anchor',s]) &
c(citex_diag_how_hi_btncitex_diag_dlg, citex_diag_hi_lblfrm) &
pack(T, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) :-
        (appl_pdw :: get(sys(diag)) ->          (!,T =
        citex_diag_ok_cancel_btncitex_diag_dlg);
                T = citex_diagTreat_ok_cancel_btncitex_diag_dlg
        ) &
c(T, citex_diag_dlg) :-
        (appl_pdw :: get(sys(diag)) -> (!,T = citex_diag_ok_cancel_btncitex_diag_dlg);
                T = citex_diagTreat_ok_cancel_btncitex_diag_dlg
        ) &
super(dialog)
}.
citex_diag_3_rgt_btncitex_diag_dlg :: {
widget(citex_diag_3_rgt_btncitex_diag_dlg, ['-orient',vertical], ['-padx','','-pady','']) &
default(move_to_sus) &
button(move_to_sus, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_3_rgt_btncitex_diag_dlg :: move_sus'], '') &
% display the suspected disorder
move_sus :-
        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
        ;       :set_susbs(Fin,s),
                disorder :: reset_att(suspected/1),
                diag_inference :: predict,
                disorder :: get(suspected(Dsu)),
                (       Dsu = [] ->
                        krol_msgs :: show("There is no sufficient finidings to suspect 
disorders  ....",[]);
                        :sort(Dsu, Dsu1),
                        citex_diag_sus_hlst :: clean(citex_diag_dlg),
                        :insert_in_hlist(Dsu1,citex_diag_sus_hlst),
                        :in_process(Dsu1, confirmed, IPs),
                        confirm_disorders :: abduct_all(IPs, OPs),
                        :retractall(prop(_,_)),
                        :retractall(val(_,_,_)),
                        :out_process(OPs,Cps),
                        :sort(Cps,Cps1),
                        citex_diag_property_hlst :: clean(citex_diag_dlg),
                        citex_diag_value_hlst :: clean(citex_diag_dlg),
                        citex_diag_conf_hlst :: clean(citex_diag_dlg),
                        citex_diag_hi_hlst :: clean(citex_diag_dlg),
                        :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                )
        ) &
% display the confirm disorder
button(move_to_confirm, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_3_rgt_btncitex_diag_dlg :: move_to_confirm'], '') &
move_to_confirm :-
        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
        ;       disorder :: get(suspected(L)),
                (       L = [] ->
                        krol_msgs :: show("Please, get suspected disoredrs.. first  ....",[])
                ;       :set_susbs(Fin,c),
                        disorder :: reset_att(suspected/1),
                        :set_sus_dis(L,suspected),
                        disorder :: reset_att(confirmed/1),
                        diag_inference :: confirm,
                        disorder :: get(confirmed(Dsu)),
                        (Dsu = [] ->
                        krol_msgs :: show("There is no sufficient finidings to confirm
                        disorders  ....",[])
                        ;       :sort(Dsu, Dsu1),
                                citex_diag_conf_hlst :: clean(citex_diag_dlg),
                                :insert_in_hlist(Dsu1,citex_diag_conf_hlst),
                                :in_process(Dsu1, highly_confirmed, IPs),
                                verify_disorders :: abduct_all(IPs, OPs),
                                :retractall(prop(_,_)),
                                :retractall(val(_,_,_)),
                                :out_process(OPs,Cps),
                                :sort(Cps,Cps1),
                                        citex_diag_property_hlst :: clean(citex_diag_dlg),
                                        citex_diag_value_hlst :: clean(citex_diag_dlg),
                                        citex_diag_hi_hlst :: clean(citex_diag_dlg),
                                        :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                                )
                        )
                )&
% display the highy confirm disorder
button(move_to_hiconfirm, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_3_rgt_btncitex_diag_dlg :: move_to_hiconfirm'], '') &
move_to_hiconfirm :-
        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
        ;       disorder :: get(confirmed(L)),
                (       L = [] ->
                        krol_msgs :: show("Please, get confirmed disoredrs.. first  ....",[])
                ;       :set_susbs(Fin,h),
                        disorder :: reset_att(confirmed/1),
                        :set_sus_dis(L,confirmed),
                        disorder :: reset_att(highly_confirmed/1),
                        diag_inference :: verify,
                        disorder :: get(highly_confirmed(Dcu)),
                        (       Dcu = [] ->
                                krol_msgs :: show("There is no sufficient finidings to
get on highly confirmed disorders  ....",[])
                        ;       :sort(Dcu, Dcu1),
                                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                                :insert_in_hlist(Dcu1,citex_diag_hi_hlst),
                                citex_diag_property_hlst :: clean(citex_diag_dlg),
                                citex_diag_value_hlst :: clean(citex_diag_dlg),
                                citex_diag_concept_hlst :: clean(citex_diag_dlg)
                        )
                )
        )&
super(buttonbox)
}.
citex_diag_Property_lblfrm :: {
widget(citex_diag_Property_lblfrm, ['-label','Properties','-labelside',top], []) &
super(labelframe)
}.

citex_diag_all_lblfrm :: {
widget(citex_diag_all_lblfrm, ['-label','All Disorder','-labelside',top], []) &
super(labelframe)
}.
citex_diag_dwn_lblfrm :: {
widget(citex_diag_dwn_lblfrm, ['-label','User Suspected Disorder','-labelside',top], []) &
super(labelframe)
}.
```

```
citex_diag_conc_lblfrm :: {
widget(citex_diag_conc_lblfrm, ['-label','Concepts','-labelside',top], []) &
super(labelframe)
}.
citex_diag_conc_prop_val_fin_frm :: {
widget(citex_diag_conc_prop_val_fin_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.
citex_diag_confirm_lblfrm :: {
widget(citex_diag_confirm_lblfrm, ['-label','Confirmed Disorders','-labelside',top], []) &
super(labelframe)
}.
citex_diag_down_btncitex_diag_dlg :: {
widget(citex_diag_down_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
default(move_down) &
button(move_down, ['-bg', gray,'-image','Arrowdwn.gif','-
command','citex_diag_down_btncitex_diag_dlg :: move_down'], '') &
% move the legal value to the button list
move_down :-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        citex_diag_property_hlst :: fetch(citex_diag_dlg,P),
        citex_diag_value_hlst :: fetch(citex_diag_dlg,V),
        (       (C = '/' ; P = '/' ; V = '/') ->
                :true
        ;       :format_to_chars('~q of ~q = ~q', [P,C,V], Str),
                :name(I, Str),
                (       citex_diag_finding_hlst :: item(I) ->
                        :true
                ;       citex_diag_finding_hlst :: insert_item(citex_diag_dlg,I),
                        :assert(finding(I,C,P,V))
                )
        ) &
super(buttonbox)
}.
citex_diag_down_lft_btn :: {
widget(citex_diag_down_lft_btn, ['-orient',horizontal], ['-padx','','-pady','']) &
default(move_down) &
button(move_down, ['-bg', gray,'-image','Arrowdwn.gif','-
command','citex_diag_down_lft_btn :: move_to_user_sus'], '') &
% move the disorder to the user suspected list
move_to_user_sus :-
        citex_diag_all_hlst :: fetch(citex_diag_dlg,C),
        (       (C = '/' ) ->
                :true
        ;       citex_diag_dwn_hlst :: is_item(citex_diag_dlg,C) ->
                :true
        ;       citex_diag_dwn_hlst :: insert_item(citex_diag_dlg,C)
        ) &
super(buttonbox)
}.
```

```
citex_diag_finding_lblfrm :: {
widget(citex_diag_finding_lblfrm, ['-label','Findings','-labelside',top], []) &
super(labelframe)
}.
citex_diag_frm :: {
widget(citex_diag_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.
citex_diag_hi_lblfrm :: {
widget(citex_diag_hi_lblfrm, ['-label','High Confirmed Disorders','-labelside',top], []) &
super(labelframe)
}.
citex_diag_how_conf_btncitex_diag_dlg :: {
widget(citex_diag_how_conf_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
        default(how_conf) &
button(how_conf, ['-text','How','-command','citex_diag_how_conf_btncitex_diag_dlg ::
how_conf'], '') &
how_conf :-
        (
        krol_msgs :: show("It will be implemented soon....",[])
        )&
super(buttonbox)
}.
citex_diag_how_hi_btncitex_diag_dlg :: {
widget(citex_diag_how_hi_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
default(how_hi) &
button(how_hi, ['-text','How','-command','citex_diag_how_hi_btncitex_diag_dlg :: how_hi'],
'') &
how_hi :-
        krol_msgs :: show("It will be implemented soon....",[])&
super(buttonbox)
}.
citex_diag_how_sus_btncitex_diag_dlg :: {
widget(citex_diag_how_sus_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
default(how_sus) &
button(how_sus, ['-text','How','-command','citex_diag_how_sus_btncitex_diag_dlg ::
how_sus'], '') &
how_sus :-
        krol_msgs :: show("It will be implemented soon....",[])&
super(buttonbox)
}.
citex_diag_left_btncitex_diag_dlg :: {
widget(citex_diag_left_btncitex_diag_dlg, ['-orient',vertical], ['-padx','','-pady','']) &
default(move_left) &
button(move_left, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_left_btncitex_diag_dlg :: move_left'], '') &
%maryam   minimize the list of observation
move_left :-
        citex_diag_dwn_hlst :: content(citex_diag_dlg, L1),
        (        L1 = [] ->
                    :true
          ;         :retractall(prop(_,_)),
                    :retractall(val(_,_,_)),
                    :retractall(finding(_,_,_,_)),
                    :retractall(prop_type(_,_,_,_)),
                    :in_process(L1, suspected, IPs),
                    caused_by_disoders :: abduct_all(IPs, OPs),
                    :out_process(OPs,Cps),
                    :sort(Cps,Cps1),
                    citex_diag_property_hlst :: clean(citex_diag_dlg),
                    citex_diag_value_hlst :: clean(citex_diag_dlg),
                    citex_diag_finding_hlst :: clean(citex_diag_dlg),
                    citex_diag_sus_hlst :: clean(citex_diag_dlg),
                    citex_diag_conf_hlst :: clean(citex_diag_dlg),
                    citex_diag_hi_hlst :: clean(citex_diag_dlg),
                    :insert_in_hlist(Cps1,citex_diag_concept_hlst)
          )&

super(buttonbox)
}.
citex_diag_ok_cancel_btncitex_diag_dlg :: {
widget(citex_diag_ok_cancel_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
default(ok) &
button(ok, ['-text','Ok','-command','citex_diag_ok_cancel_btncitex_diag_dlg :: ok'], '') &
ok :-
          citex_diag_dlg :: destroy &
super(buttonbox)
}.
citex_diagTreat_ok_cancel_btncitex_diag_dlg :: {
widget(citex_diagTreat_ok_cancel_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-
pady','']) &
default(ok) &
button(ok, ['-text','Ok','-command','citex_diagTreat_ok_cancel_btncitex_diag_dlg :: ok'], '') &
ok :-
          citex_diag_dlg :: destroy &
button(treat, ['-text','Treatment','-command','citex_diagTreat_ok_cancel_btncitex_diag_dlg ::
treat'], '') &
treat :-
% Maryam confirm, high confirm lists
          disorder :: get(confirmed(L1)),
          disorder :: get(highly_confirmed(L2)),
          :append(L1,L2,Dis),
          (         Dis = [] ->
                    krol_msgs :: show("There are no disorders confirmed",[])
          ;
                    disorder :: reset_att(confirmed/1),
                    disorder :: reset_att(highly_confirmed/1),
                    krol_init :: init,
                    :set_sus_dis(L1,confirmed),
                    :set_sus_dis(L2,highly_confirmed),
                    krol_init :: set(mode(cm)),
                    treat_task_uncondional :: start_inference
          ) &
super(buttonbox)}.
citex_diag_rgt_sus_conf_hi_frm :: {
widget(citex_diag_rgt_sus_conf_hi_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)}.
citex_diag_ses_all_frm :: {
widget(citex_diag_ses_all_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)}.
citex_diag_ses_all_lft_frm :: {
widget(citex_diag_ses_all_lft_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)}.
citex_diag_sus_conf_hi_frm :: {
widget(citex_diag_sus_conf_hi_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)}.
citex_diag_sus_lblfrm :: {
widget(citex_diag_sus_lblfrm, ['-label','Suspected Disorders','-labelside',top], []) &
super(labelframe)}.
citex_diag_all_hlst :: {
widget(citex_diag_all_hlst, ['-scrollbar', auto], ['hlist.selectmode multiple','hlist.itemtype
imagetext hlist.drawBranch false hlist.indent 14 hlist.wideSelect false']) &
super(hlist)}.
citex_diag_dwn_hlst :: {
widget(citex_diag_dwn_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch
false hlist.indent 14 hlist.wideSelect false']) &
super(hlist)}.
citex_diag_concept_hlst :: {
widget(citex_diag_concept_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext
hlist.drawBranch false hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
configure(['-browsecmd show_properties']) &
super(hlist)}.
citex_diag_property_hlst :: {
widget(citex_diag_property_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext
hlist.drawBranch false hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
configure(['-browsecmd show_values']) &
super(hlist)}.
citex_diag_value_hlst :: {
widget(citex_diag_value_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch
false hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
super(hlist)}.
citex_diag_finding_hlst :: {
widget(citex_diag_finding_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext
hlist.drawBranch false hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
super(hlist)}.
citex_diag_sus_hlst :: {
widget(citex_diag_sus_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch
false hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)}.
citex_diag_conf_hlst :: {
```

```
widget(citex_diag_conf_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch
false hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)}.
citex_diag_hi_hlst :: {
widget(citex_diag_hi_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch
false hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)}.
citex_diag_value_lblfrm :: {
widget(citex_diag_value_lblfrm, ['-label','Values','-labelside',top], []) &
super(labelframe)}.
citex_diag_why_what_btncitex_diag_dlg :: {
widget(citex_diag_why_what_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady','']) &
default(why) &
button(why, ['-text','Why','-command','citex_diag_why_what_btncitex_diag_dlg :: why'], '')&
why :-
        krol_msgs :: show("It will be implemented soon....",[])&
button(delete, ['-text','Delete','-command','citex_diag_why_what_btncitex_diag_dlg ::
delete'], '') &
delete :-
        citex_diag_finding_hlst :: fetch(citex_diag_dlg,I),
        (       I = '/' ->
                :true
        ;       citex_diag_finding_hlst :: delete_item(citex_diag_dlg,I),
                :finding(I,C,P,V),
                :retractall(finding(I,_C,_P,_V)),
                :delete_subs(C,P,V)
        ) &
button(what, ['-text','What','-command','citex_diag_why_what_btncitex_diag_dlg :: what'], '')
&
what :-
        krol_msgs :: show("It will be implemented soon....",[])&

super(buttonbox)}.
citex_diag_del_btn :: {
widget(citex_diag_del_btn, ['-orient',horizontal], ['-padx','','-pady','']) &
default(del) &
button(delete, ['-text','Delete','-command','citex_diag_del_btn :: delete'], '') &
delete :-
        citex_diag_dwn_hlst :: fetch(citex_diag_dlg,I),
        (       I = '/' ->
                :true
        ;       citex_diag_dwn_hlst :: delete_item(citex_diag_dlg,I),
                :retractall(prop(_,_)),
                :retractall(val(_,_,_)),
                :retractall(finding(_,_,_,_)),
                :retractall(prop_type(_,_,_,_)),
                citex_diag_concept_hlst :: clean(citex_diag_dlg),
                citex_diag_property_hlst :: clean(citex_diag_dlg),
                citex_diag_value_hlst :: clean(citex_diag_dlg),
                citex_diag_finding_hlst :: clean(citex_diag_dlg),
                citex_diag_sus_hlst :: clean(citex_diag_dlg),
                citex_diag_conf_hlst :: clean(citex_diag_dlg),
                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                :findall(X,citex_diag_dwn_hlst :: is_item(citex_diag_dlg,X),L1),
                (       L1 = [] ->
                        :true
                ;       :in_process(L1, suspected, IPs),
                        caused_by_disoders :: abduct_all(IPs, OPs),
                        :out_process(OPs,Cps),
                        :sort(Cps,Cps1),
                        :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                )
        )&
super(buttonbox)}.
get_disorders :-
        krol_init :: init,
        plant :: get_value(season(Season)),
        season(Season,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        retractall(finding(_,_,_,_)),
        retractall(prop_type(_,_,_,_)),
        sort(L,L1),
        insert_in_hlist(L1,citex_diag_all_hlst),
        in_process(L1, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_finding_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_dwn_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
init_disorders :-
        krol_init :: init,
        plant :: get_value(season(Seson)),
        season(Seson,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        retractall(finding(_,_,_,_)),
        retractall(prop_type(_,_,_,_)),
        sort(L,L1),
        insert_in_hlist(L1,citex_diag_all_hlst),
        in_process(L1, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_finding_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_dwn_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
show_properties:-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        findall(P,prop(C,P),Lp),
        sort(Lp, Lp1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Lp1,citex_diag_property_hlst).
show_values:-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        citex_diag_property_hlst :: fetch(citex_diag_dlg,P),
        findall(V,val(C,P,V),Lv),
        sort(Lv, Lv1),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Lv1,citex_diag_value_hlst).
clean_hlist(H) :-
        H :: clean(citex_diag_dlg).
insert_in_hlist(L,H) :-
        H :: clean(citex_diag_dlg),
        H :: insert(citex_diag_dlg,L).
insert_in_hlist1([],_).
insert_in_hlist1([H|T],O) :-
        O :: insert_item(citex_diag_dlg,H),
        insert_in_hlist1(T,O).
in_process([], _, []).
in_process([D|Ds], F, [P in disorder|IPs]) :-
        P =.. [F,D],
        in_process(Ds, F, IPs).
out_process([], []).
out_process([C-P-V|OPs], [C|Cs]) :-
        (       prop(C,P) ->
                true
        ;       assert(prop(C,P))
        ),
        assert(val(C,P,V)),
        out_process(OPs, Cs).
set_susbs([],_).
set_susbs([C-P-V|Vals],Type) :-
        (       prop_type(C,P,V,_) ->
                true
        ;       F =.. [P,V],
                C :: set_value(F),
                assert(prop_type(C,P,V,Type))
        ),
        set_susbs(Vals,Type).
set_sus_dis([],_).
set_sus_dis([H|L],V) :-
        F =.. [V,H],
        disorder :: set_value(F),
        set_sus_dis(L,V).
del_sus_dis([]).
del_sus_dis([H|L]) :-
        H :: reset,
        del_sus_dis(L).
delete_subs(C,A,V) :-
        (       C :: is_single(A/1) ->
                C :: reset_att(A/1)
        ;       P =.. [A,Vs],
                C :: get(P),
                delete(Vs, V, Vs1),
                P1 =.. [A, Vs1],
                C :: set(P1)
        ),
        retract(prop_type(C,A,V,Type)),
        handle_p_type(Type).
p_type(h) :-
        disorder :: get(suspected(L)),
        disorder :: reset_att(suspected/1),
        set_sus_dis(L,suspected),
        disorder :: reset_att(confirmed/1),
        diag_inference :: confirm ,
        disorder :: get(confirmed(Dsu)),
        sort(Dsu, Dsu1),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Dsu1,citex_diag_conf_hlst),
        in_process(Dsu1, highly_confirmed, IPs),
        verify_disorders :: abduct_all(IPs, OPs),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
handle_p_type(c) :-
        handle_type(h),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        disorder :: reset_att(suspected/1),
        diag_inference :: predict,
        disorder :: get(suspected(Dsu)),
        sort(Dsu, Dsu1),
        insert_in_hlist(Dsu1,citex_diag_sus_hlst),
        in_process(Dsu1, confirmed, IPs),
        confirm_disorders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
handle_p_type(s) :-
        handle_type(h),
        handle_type(c),
        plant :: get(season(Seson)),
        season(Seson,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        in_process(L, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
handle_type(X) :-
        forall(retract(prop_type(C,A,V,X)),
                (         retract(finding(I,C,A,V)),
                          citex_diag_finding_hlst :: delete_item(citex_diag_dlg,I),
                          (         C :: is_single(A/1) ->
                                    C :: reset_att(A/1)
                          ;         P =.. [A,Vs],
                                    C :: get(P),
                                    delete(Vs, V, Vs1),
                                    P1 =.. [A, Vs1],
                                    C :: set(P1)
                          )
                )
        ).
get_treat(Ds) :-
        sort(Ds,Ds1),
        get_treat(Ds1, Ts1),
        sort(date_sort) :: qsort(Ts1, Ts),
        treat_dialog :: display,
        show_treat(Ts),
        treat_dialog :: tkwait.
```

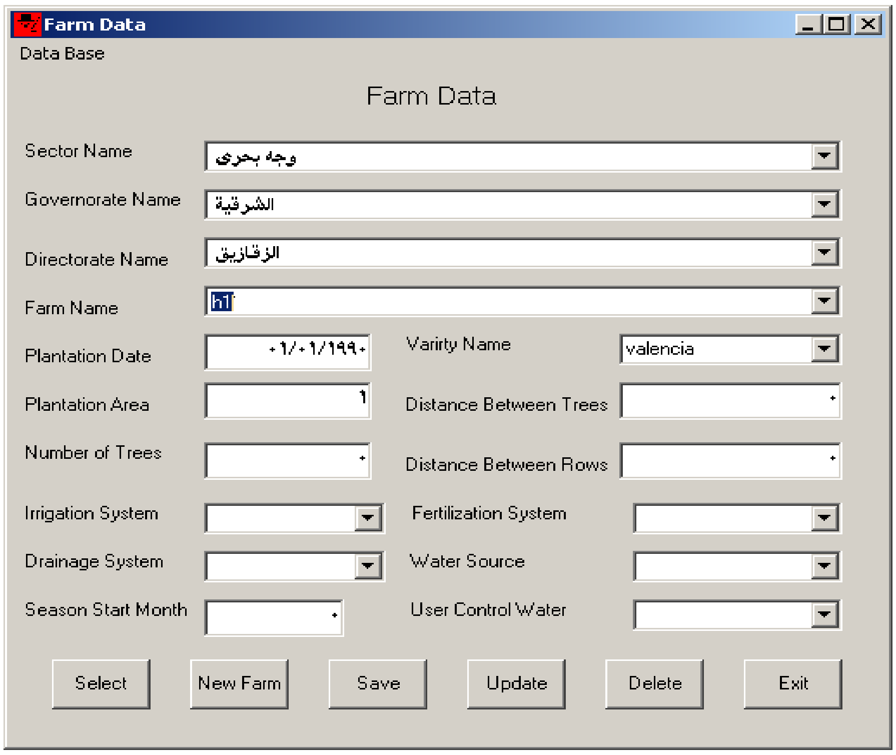
```
get_treat([], []).
get_treat([D|Ds], Ts) :-
        D :: get(method(Method)),
        (       Method = [] ->
                D :: leaves(Ls),
                (       (Ls = []; Ls = [D]) ->
                        format_to_chars("~tThere is no Treatment for the disease
~w~n~n",[D],Treat1),
                        name(Treat, Treat1),
                        T = [Treat]
                ;       append(Ls, Ds, Ds1),
                        get_treat(Ds1, Ts)
                )
        ;       my_get(D, material_name(Matx), operation),
                sort(Matx, Mat),
                my_get(D, number(Number), treat_op),
                my_get(D, date(Datex), treat_op),
                (       Datex = [] ->
                        my_get(D, special_date(Date), treat_op)
                ;       Date = Datex
                ),
                my_get(D, material_qty(Qty), operation),
                my_get(D, unit(Unit), operation),
                (       (Method = painting ; Method = disinfection ; Method = 'soil
treatment') ->
                        AT = 'any suitable time'
                ;       (Method = 'chemical spray' ; Method = 'foliage nutrition') ->
                        AT = 'early morning or afternoon'
                ;       AT = ''
                ),
                my_get(D, advice(Advx), treat_op),
                sort(Advx, Adv),
                (       (Mat = [], Method = [], Number = [], Date = [], Qty = [], Unit = [], AT
= [], Adv = []) ->
                        format_to_chars("~tThere is no Treatment for the disease
~w~n~n",[D],Treat1),
                        name(Treat, Treat1),
                        T = [Treat]
                ;       format_to_chars("Treatment of disorder ~w is :~n",[D],D1),
                        format_to_chars("      Material : ~w~n",[Mat],Mat1),
                        format_to_chars("      Method : ~w~n",[Method], Method1),
                        format_to_chars("      Number : ~w~n",[Number], Number1),
                        format_to_chars("      Qty : ~w~n",[Qty], Qty1),
                        format_to_chars("      Unit : ~w~n",[Unit], Unit1),
                        format_to_chars("      Application Time : ~w~n",[AT], AT1),
                        format_to_chars("      Advice : ~w~n~n",[Adv], Adv1),
                        name(D11, D1),
                        name(Mat11, Mat1),
                        name(Method11, Method1),
                        name(Number11, Number1),
                        name(Qty11, Qty1),
                        name(Unit11, Unit1),
                        name(AT11, AT1),
                        name(Adv11, Adv1),
                        T =
[D11,Mat11,Method11,Number11,Date,Qty11,Unit11,AT11,Adv11]
                ),
                Ts = [T|Ts1]
        ),
        get_treat(Ds, Ts1).
my_get(D, P, Super) :-
        D = Super, !,
        D :: get(P).

my_get(D, P, Super) :-
        copy_term(P, P1),
        D :: get(P1),
        (       arg(1, P1, []) ->
                D :: super(S),
                my_get(S, P, Super)
        ;       P = P1
        ).
```

File name : season_dis.pl

```
season(spring,[rose_scarab,citrus_flower_moth,psorosis,anthracnose,gummosis,wilt_root_rot
,ganoderma_rot,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_white_fl
y, scales,aphids,mealy_bug,leafminer,rust_mite,bud_mite,
brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,potassium_def,magnesi
um_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]).

season(summer,[psorosis,anthracnose,gummosis,wilt_root_rot,ganoderma_rot,armillaria_roo
t_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_white_fly,scales,aphids,mealy_bug,leaf
miner,rust_mite,bud_mite,brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_d
ef,potassium_def,magnesium_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]
).
season(autumn,[impieetratura,stubborn,sooty_mold,alternaria_rot,sun_burn,fruit_cracking,fr
uit_creasing,mediterranean_fruit_fly,green_stink_bug,psorosis,anthracnose,gummosis,wilt_r
oot_rot,ganoderma_rot,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_
white_fly,scales,aphids,mealy_bug,leafminer,rust_mite,bud_mite,brown_mite,flat_mite,citru
s_nematude,nitrogen_def,phosphorus_def, potassium_def,
magnesium_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]).

season(winter,[impieetratura,stubborn,sooty_mold,alternaria_rot,sun_burn,fruit_cracking,frui
t_creasing,mediterranean_fruit_fly,green_stink_bug,psorosis,anthracnose,gummosis,wilt_roo
t_rot,ganoderma_rot,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_whi
te_fly,scales,aphids,mealy_bug,leafminer,rust_mite,bud_mite,brown_mite,flat_mite,citrus_n
ematude,nitrogen_def,phosphorus_def,potassium_def,magnesium_def,manganese_def,iron_d
ef,calcium_def,zinc_def,salt_injury]).
```

5.5 Diagnosis Test Case

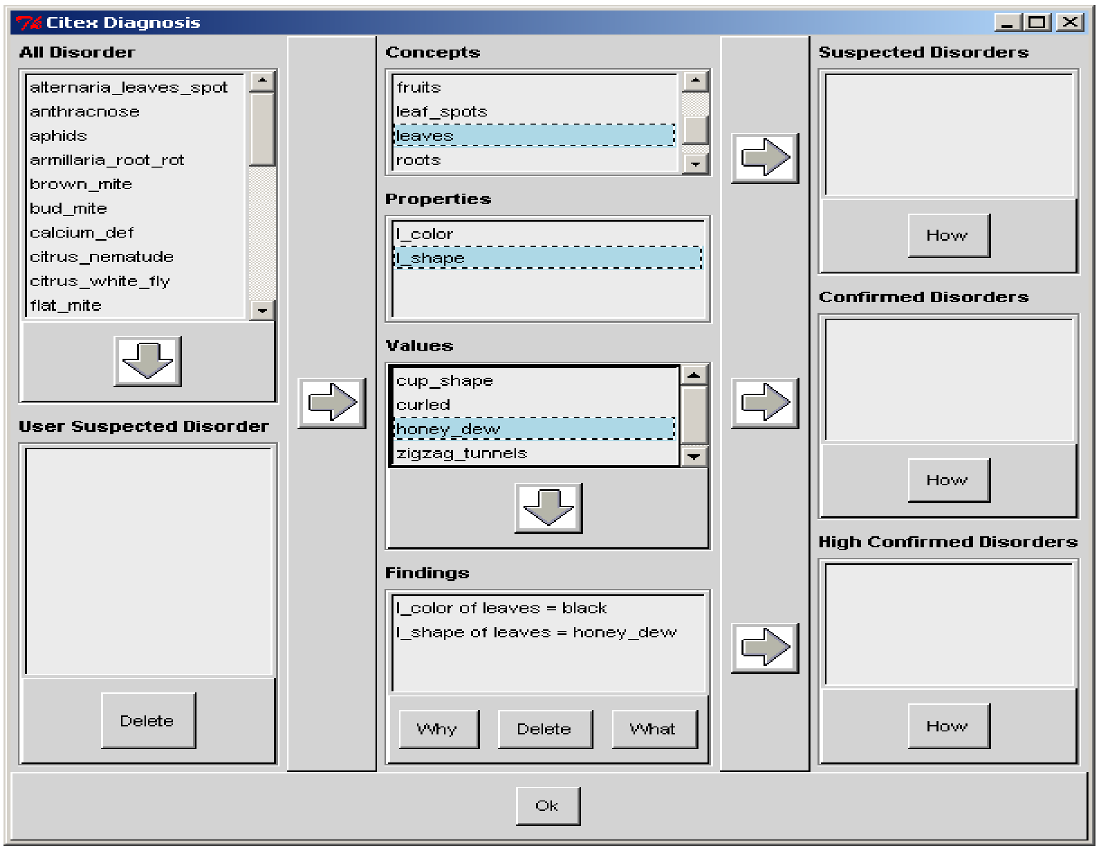
Case 1


Farm Data

Data Base

Farm Data

Sector Name	وجه بحرى		
Governorate Name	الشرقية		
Directorate Name	الزقازيق		
Farm Name	h1		
Plantation Date	٠١/٠١/١٩٩٠	Varirty Name	valencia
Plantation Area	١	Distance Between Trees	٠
Number of Trees	٠	Distance Between Rows	٠
Irrigation System		Fertilization System	
Drainage System		Water Source	
Season Start Month	٠	User Control Water	


Citex Diagnosis

All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite

User Suspected Disorder

Concepts
fruits
leaf_spots
leaves
roots

Properties
l_color
l_shape

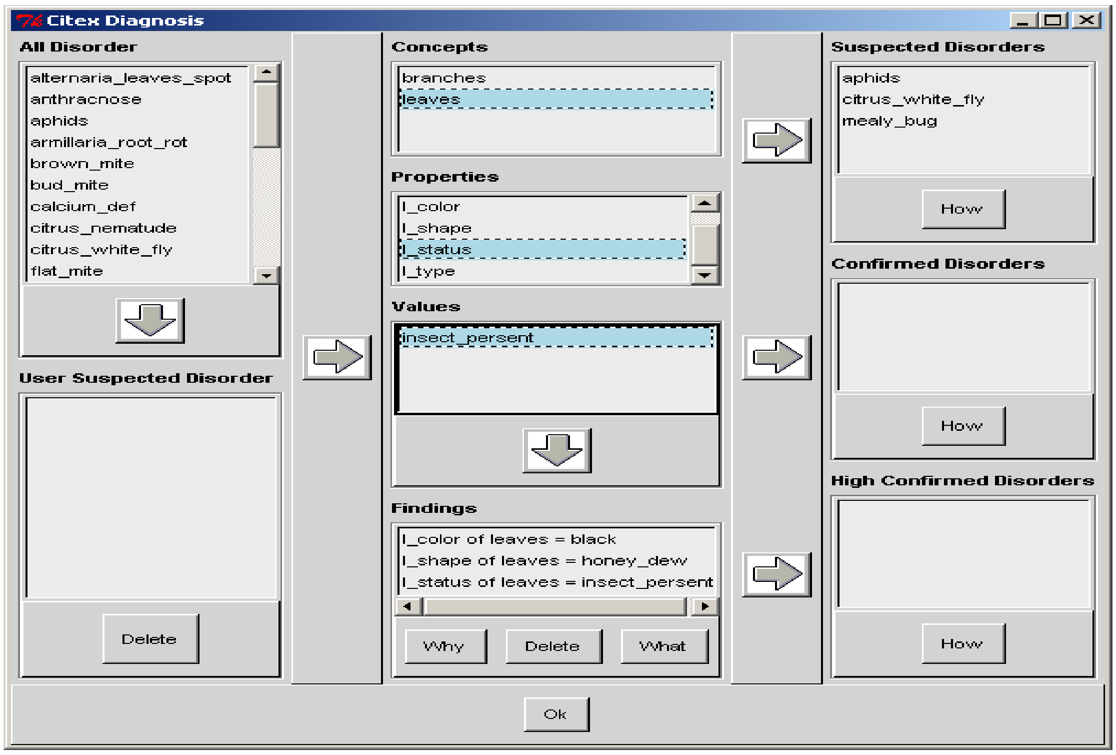
Values
cup_shape
curled
honey_dew
zigzag_tunnels

Findings
l_color of leaves = black
l_shape of leaves = honey_dew

Suspected Disorders

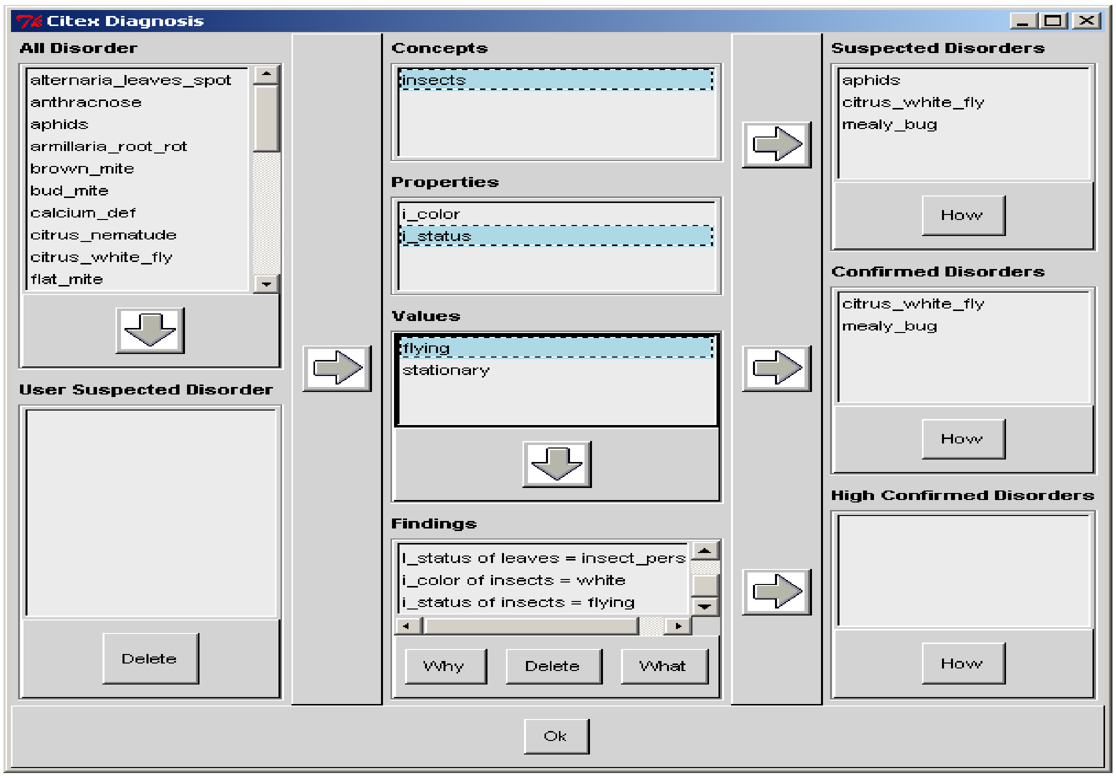
Confirmed Disorders

High Confirmed Disorders



l_color of leaves = black
l_shape of leaves = honey_dew
l_status of leaves = insect_persent

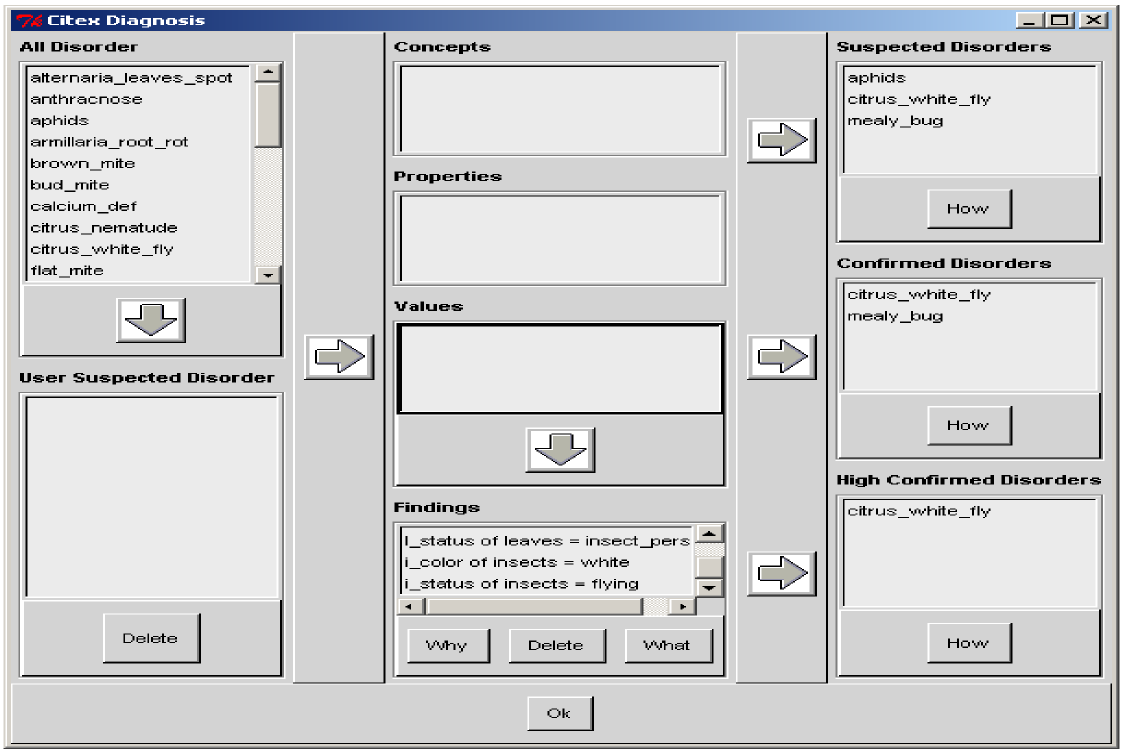
Suspected Disorders: aphids, citrus_white_fly, mealy_bug



l_status of leaves = insect_pers
i_color of insects = white
i_status of insects = flying

Suspected Disorders: aphids, citrus_white_fly, mealy_bug

Confirmed Disorders: citrus_white_fly, mealy_bug



Citex Diagnosis

All Disorder

alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite

User Suspected Disorder

Concepts

Properties

Values

Findings

l_status of leaves = insect_pers
i_color of insects = white
i_status of insects = flying

Suspected Disorders

aphids
citrus_white_fly
mealy_bug

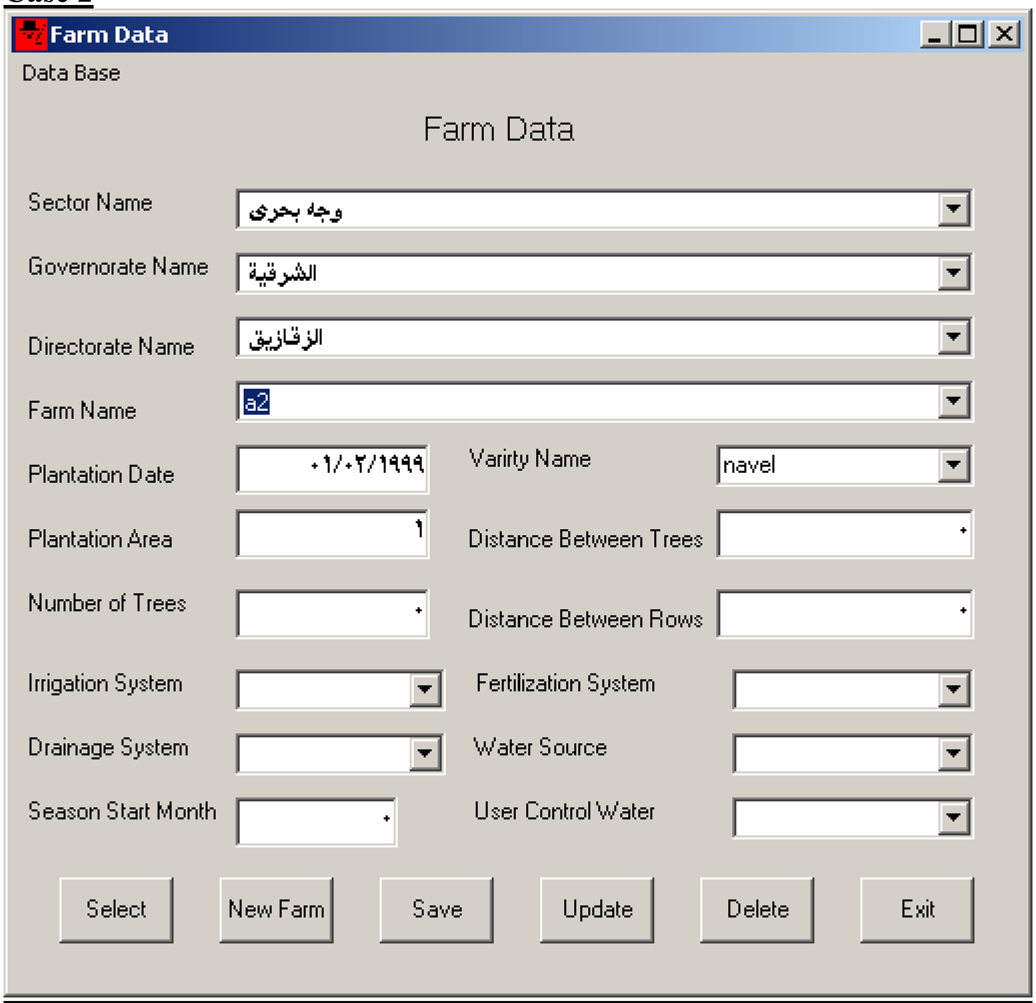
Confirmed Disorders

citrus_white_fly
mealy_bug

High Confirmed Disorders

citrus_white_fly

Case 2

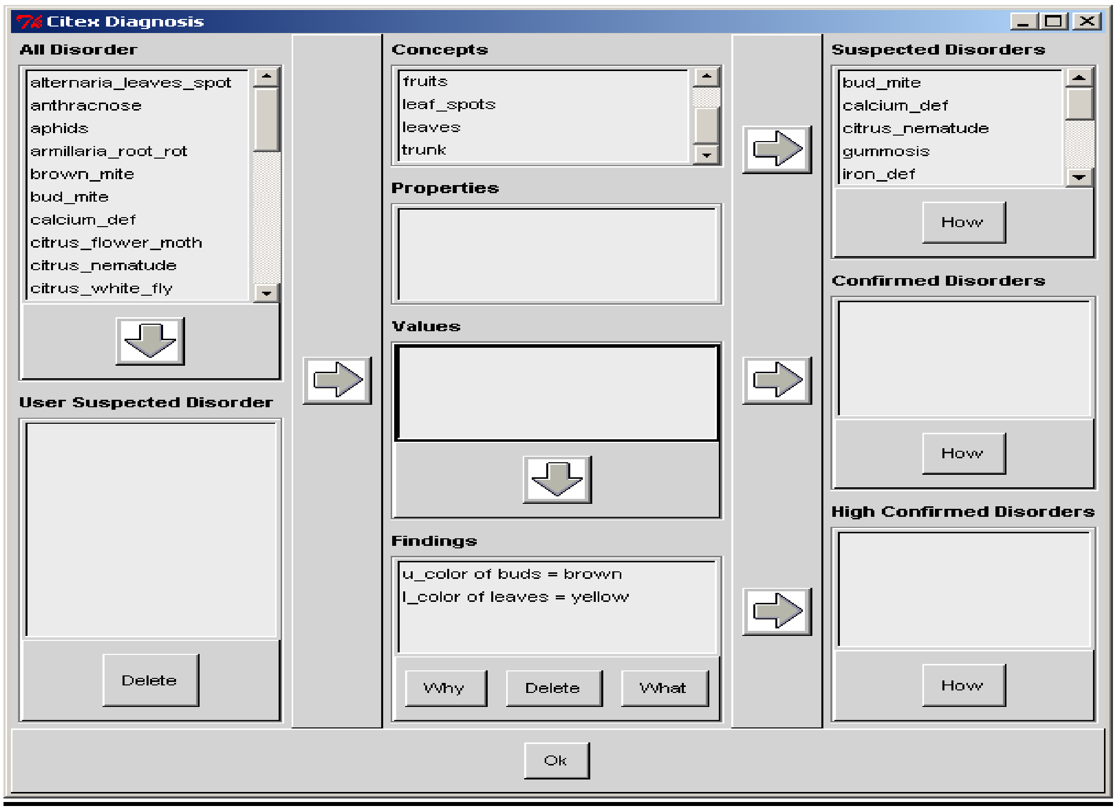


Farm Data

Data Base

Farm Data

Sector Name	وجه بحرى		
Governorate Name	الشرقية		
Directorate Name	الزقازيق		
Farm Name	a2		
Plantation Date	٠١/٠٢/١٩٩٩	Varirty Name	navel
Plantation Area	١	Distance Between Trees	٠
Number of Trees	٠	Distance Between Rows	٠
Irrigation System		Fertilization System	
Drainage System		Water Source	
Season Start Month	٠	User Control Water	


Citex Diagnosis

All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly

User Suspected Disorder

Concepts
fruits
leaf_spots
leaves
trunk

Properties

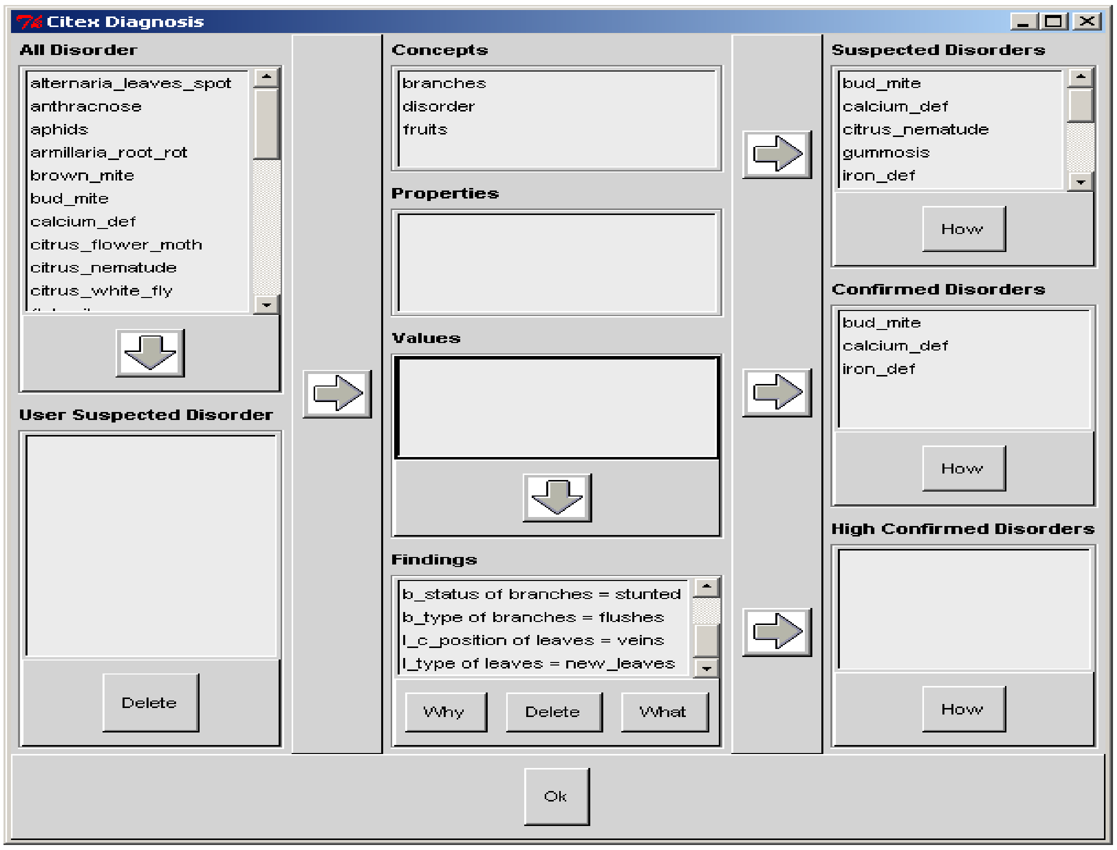
Values

Findings
u_color of buds = brown
l_color of leaves = yellow

Suspected Disorders
bud_mite
calcium_def
citrus_nematude
gummosis
iron_def

Confirmed Disorders

High Confirmed Disorders


Citex Diagnosis

All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly

User Suspected Disorder

Concepts
branches
disorder
fruits

Properties

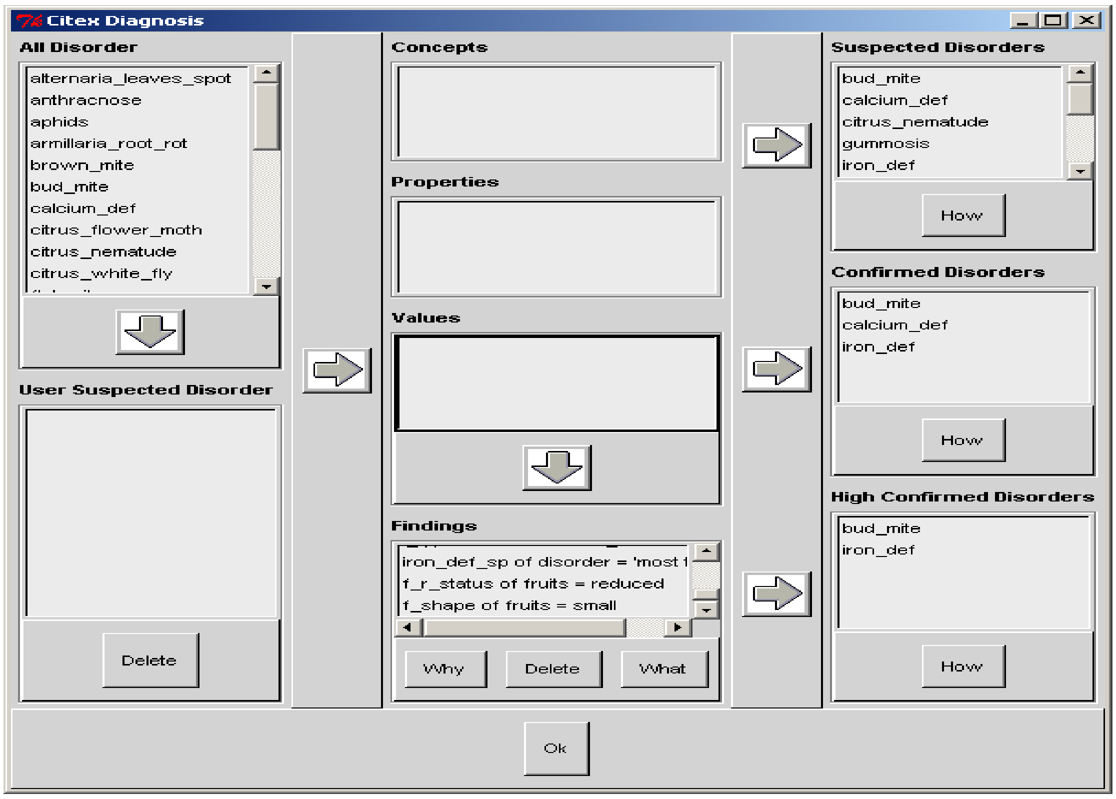
Values

Findings
b_status of branches = stunted
b_type of branches = flushes
l_c_position of leaves = veins
l_type of leaves = new_leaves

Suspected Disorders
bud_mite
calcium_def
citrus_nematude
gummosis
iron_def

Confirmed Disorders
bud_mite
calcium_def
iron_def

High Confirmed Disorders



Citex Diagnosis

All Disorder
- alternaria_leaves_spot
- anthracnose
- aphids
- armillaria_root_rot
- brown_mite
- bud_mite
- calcium_def
- citrus_flower_moth
- citrus_nematude
- citrus_white_fly

User Suspected Disorder

Concepts

Properties

Values

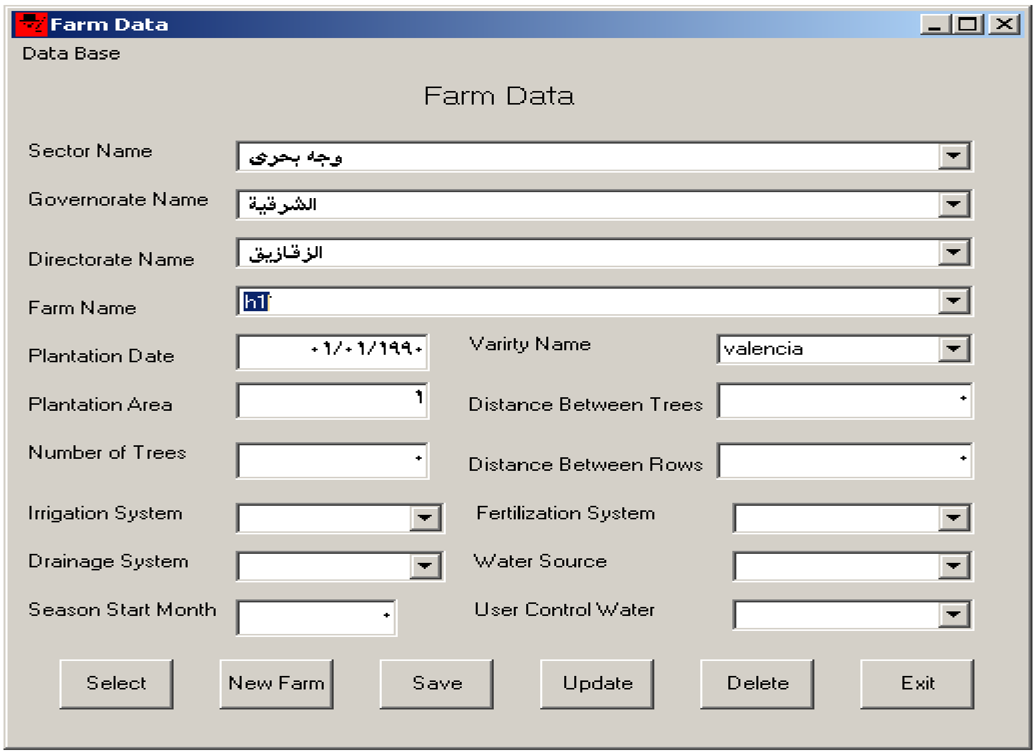
Findings
- iron_def_sp of disorder = 'most t
- f_r_status of fruits = reduced
- f_shape of fruits = small

Suspected Disorders
- bud_mite
- calcium_def
- citrus_nematude
- gummosis
- iron_def

Confirmed Disorders
- bud_mite
- calcium_def
- iron_def

High Confirmed Disorders
- bud_mite
- iron_def

Case 3

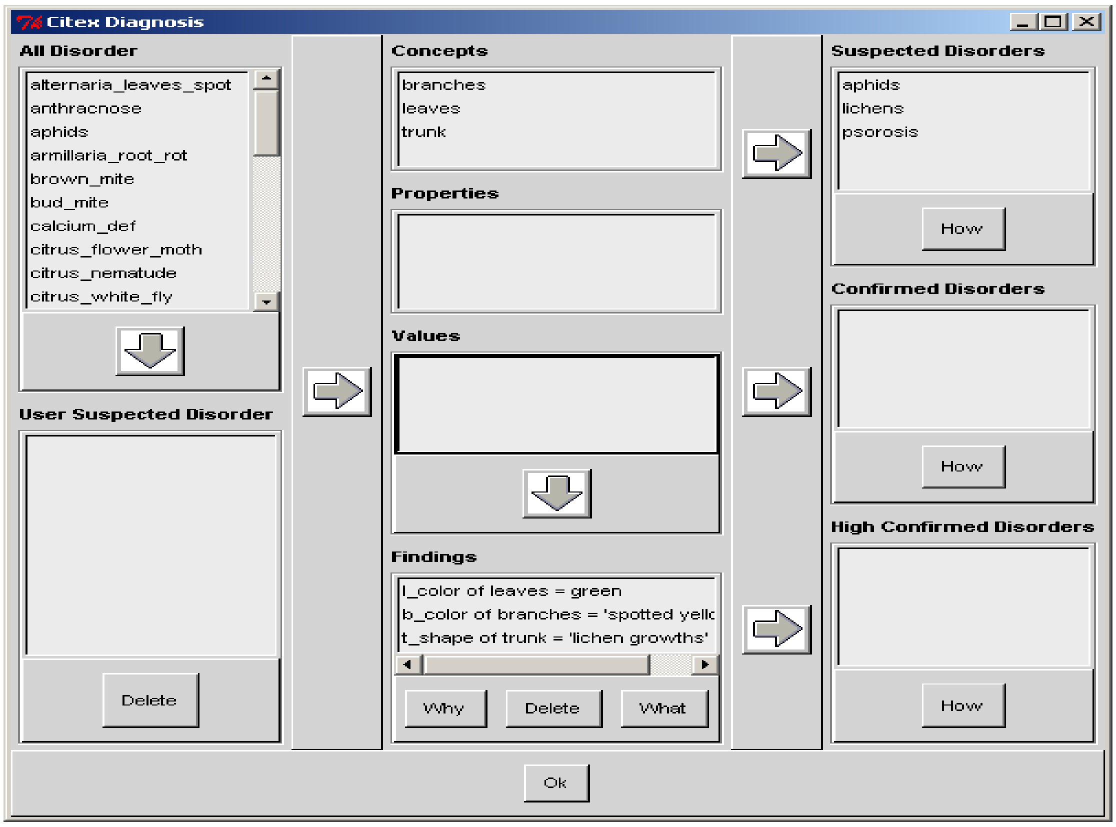


Farm Data

Data Base

Farm Data

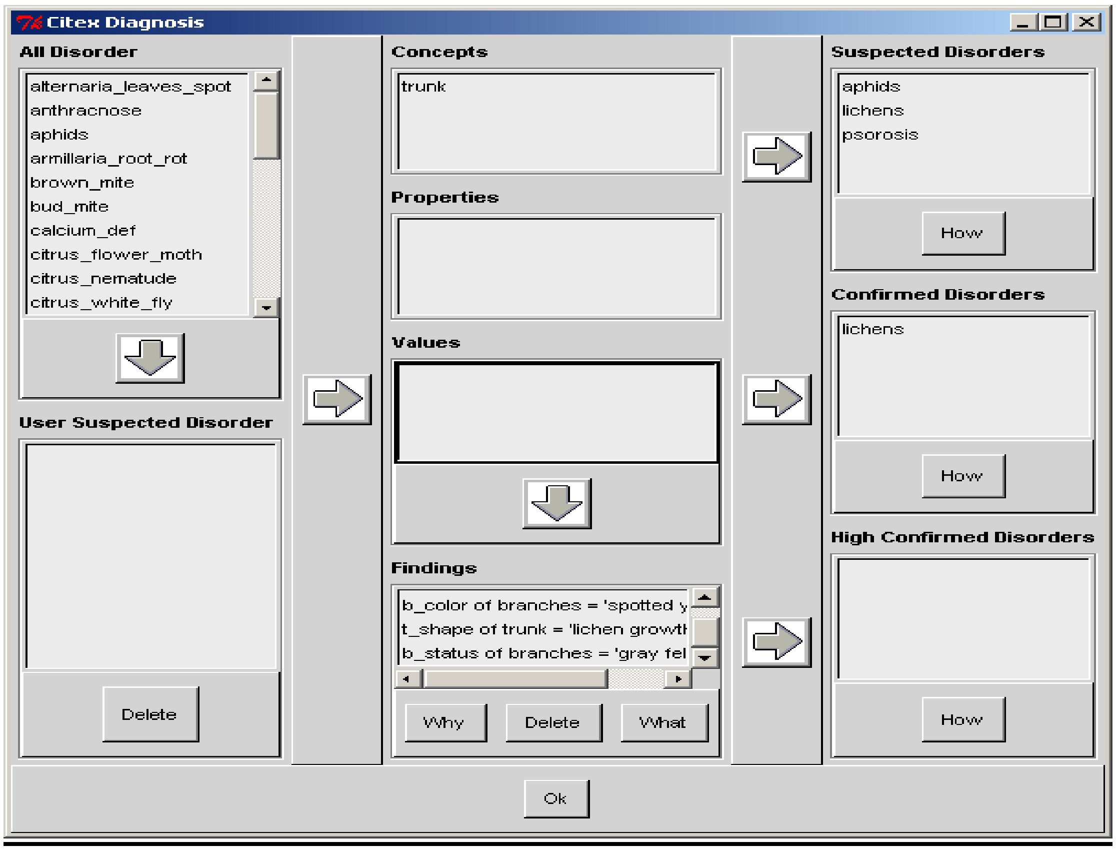
Sector Name	وجه بحرى
Governorate Name	الشرقية
Directorate Name	الزقازيق
Farm Name	h1
Plantation Date	٠١/٠١/١٩٩٠
Varirty Name	valencia
Plantation Area	١
Distance Between Trees	٠
Number of Trees	٠
Distance Between Rows	٠
Irrigation System	
Fertilization System	
Drainage System	
Water Source	
Season Start Month	٠
User Control Water	



Concepts: branches, leaves, trunk

Findings:
l_color of leaves = green
b_color of branches = 'spotted yello
t_shape of trunk = 'lichen growths'

Suspected Disorders: aphids, lichens, psorosis



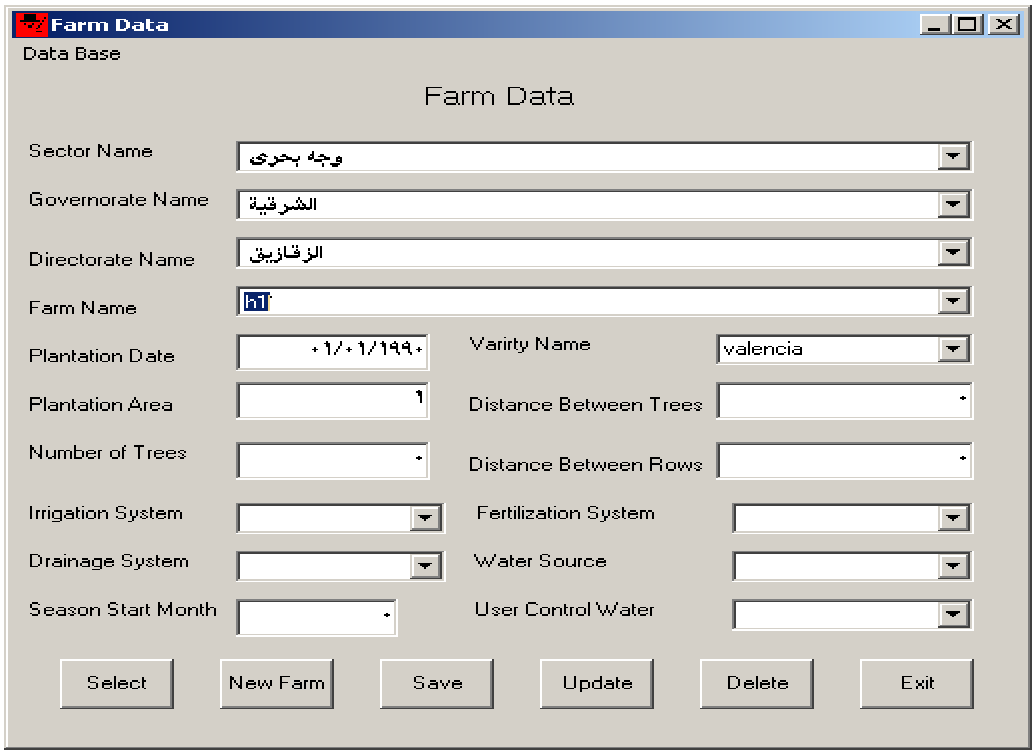
Concepts: trunk

Findings:
b_color of branches = 'spotted y
t_shape of trunk = 'lichen growth
b_status of branches = 'gray fel

Suspected Disorders: aphids, lichens, psorosis

Confirmed Disorders: lichens

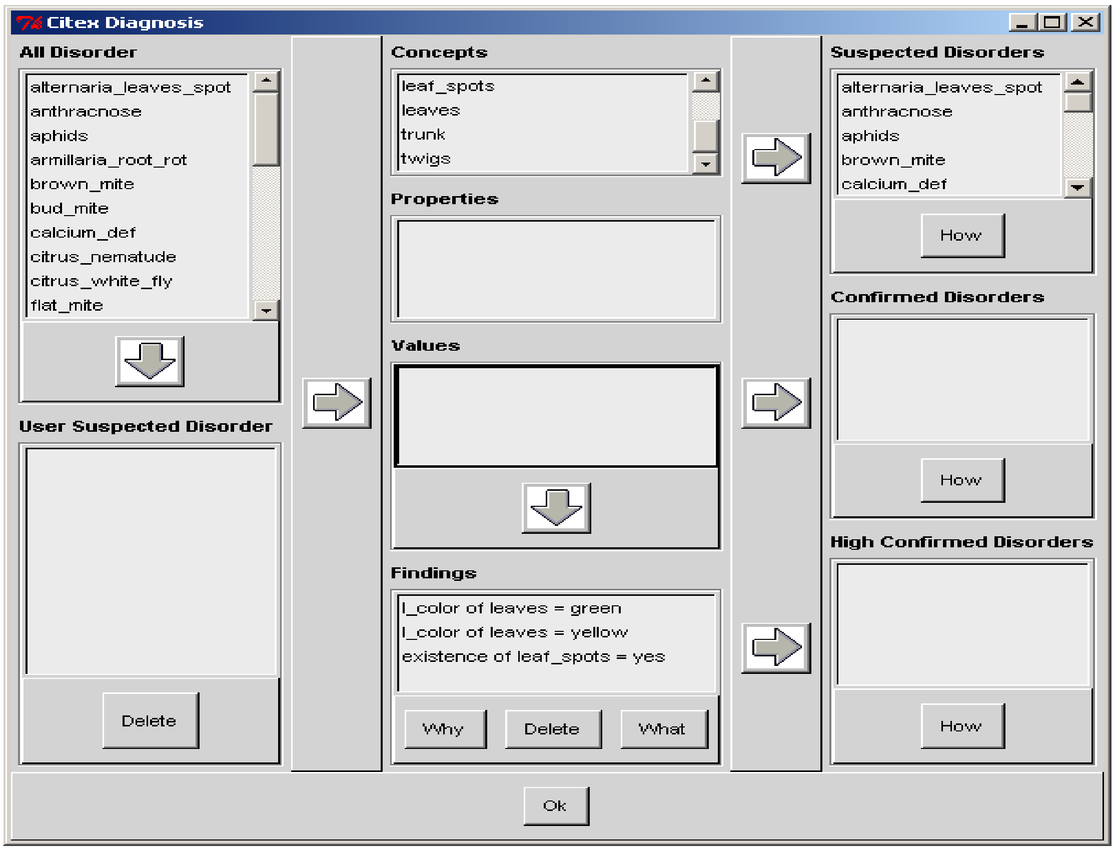
Case 4


Farm Data

Data Base

Farm Data

Sector Name	وجه بحرى		
Governorate Name	الشرقية		
Directorate Name	الزقازيق		
Farm Name	h1		
Plantation Date	٠١/٠١/١٩٩٠	Varirty Name	valencia
Plantation Area	١	Distance Between Trees	٠
Number of Trees	٠	Distance Between Rows	٠
Irrigation System		Fertilization System	
Drainage System		Water Source	
Season Start Month	٠	User Control Water	


Citex Diagnosis

All Disorder
- alternaria_leaves_spot
- anthracnose
- aphids
- armillaria_root_rot
- brown_mite
- bud_mite
- calcium_def
- citrus_nematude
- citrus_white_fly
- flat_mite

User Suspected Disorder

Concepts
- leaf_spots
- leaves
- trunk
- twigs

Properties

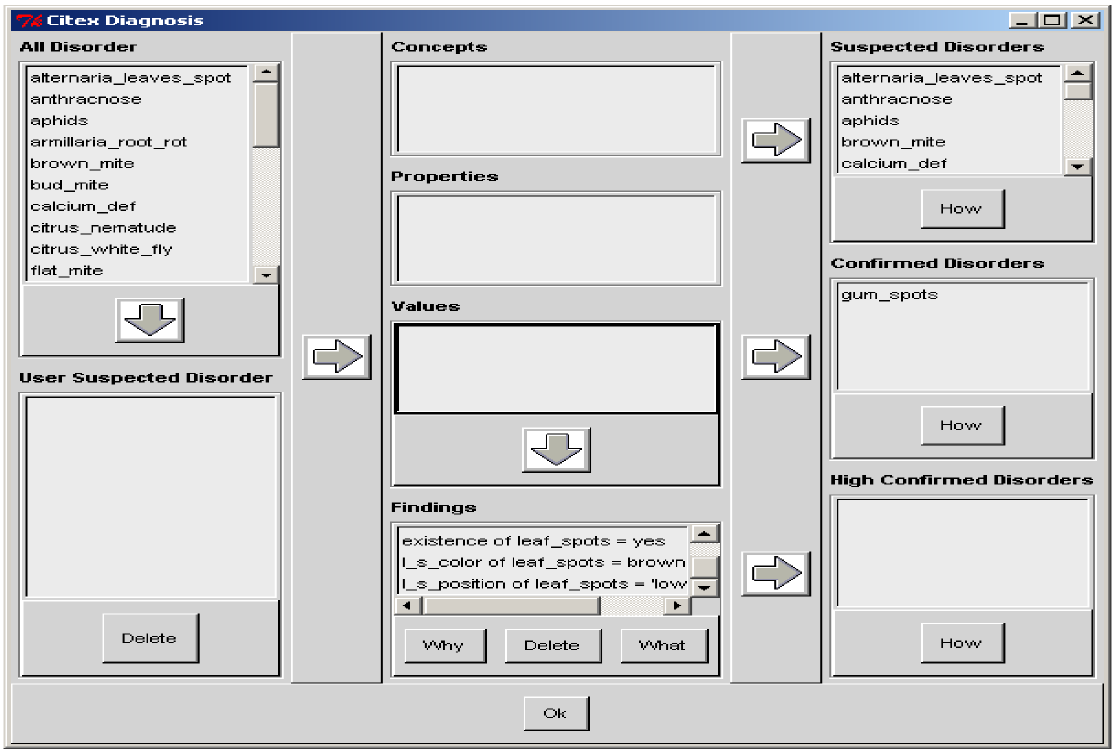
Values

Findings
- l_color of leaves = green
- l_color of leaves = yellow
- existence of leaf_spots = yes

Suspected Disorders
- alternaria_leaves_spot
- anthracnose
- aphids
- brown_mite
- calcium_def

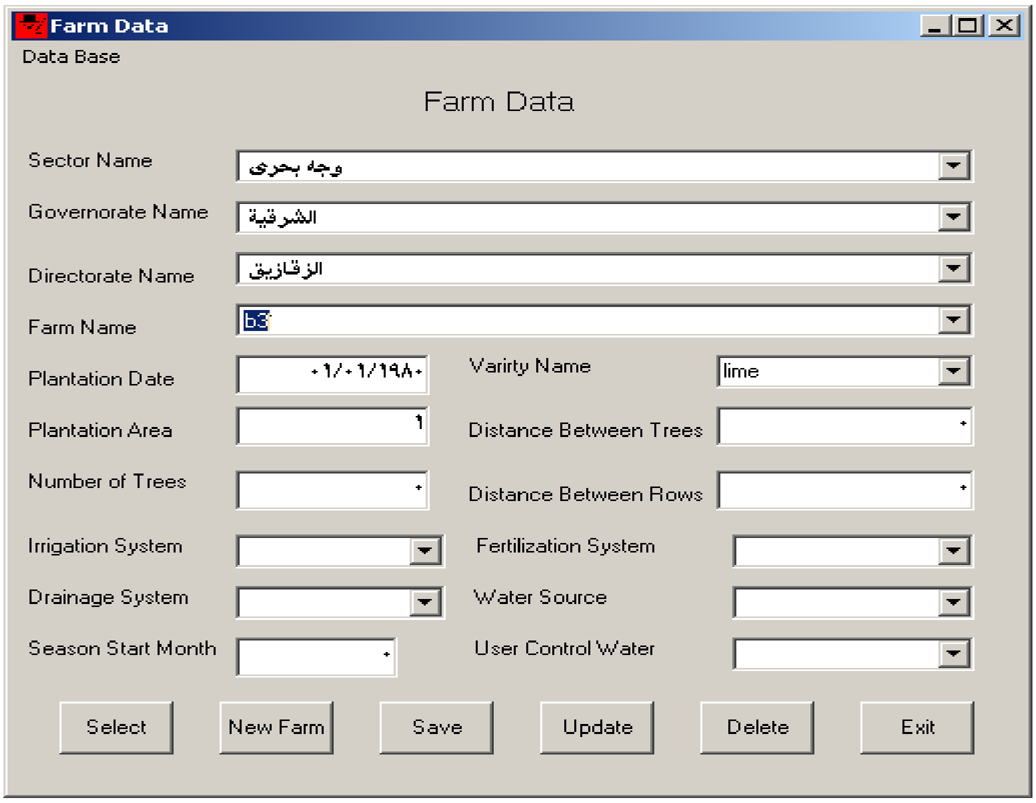
Confirmed Disorders

High Confirmed Disorders

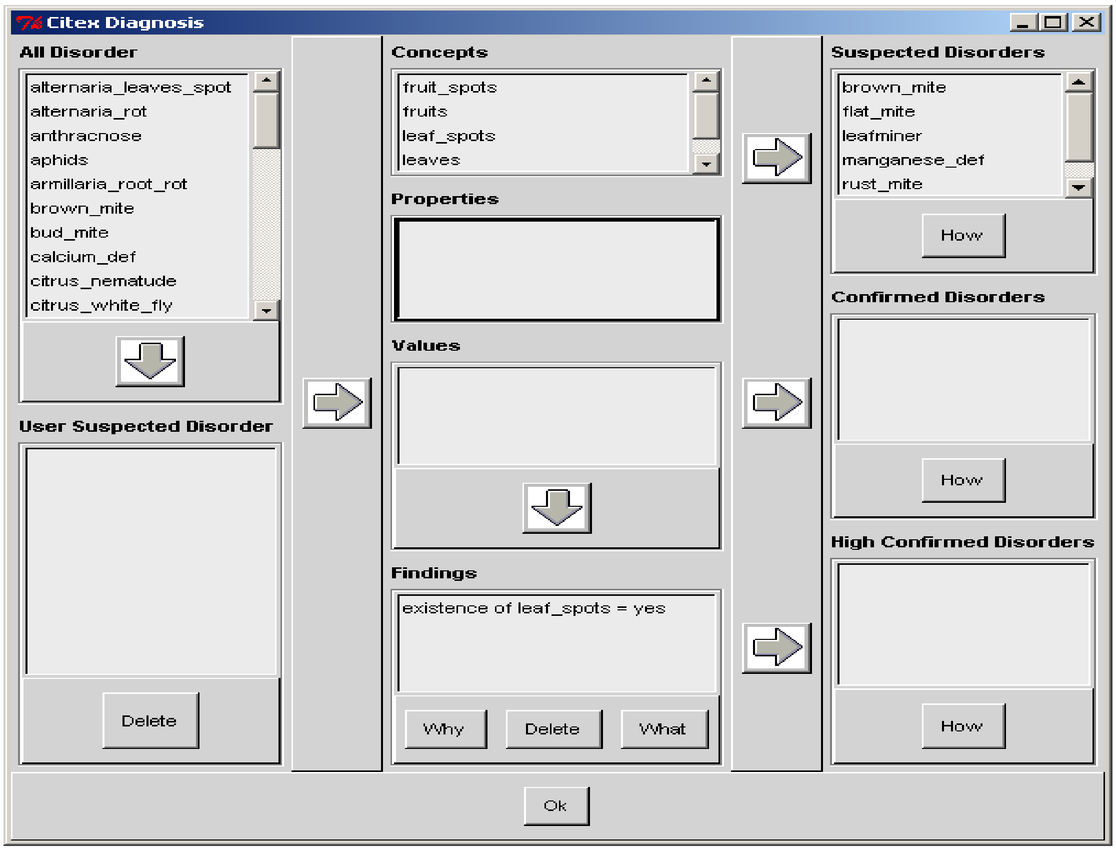


existence of leaf_spots = yes
l_s_color of leaf_spots = brown
l_s_position of leaf_spots = 'low

Case 5



Sector Name	وجه بحرى		
Governorate Name	الشرقية		
Directorate Name	الزقازيق		
Farm Name	b3		
Plantation Date	٠١/٠١/١٩٨٠	Varirty Name	lime
Plantation Area	١	Distance Between Trees	٠
Number of Trees	٠	Distance Between Rows	٠
Irrigation System		Fertilization System	
Drainage System		Water Source	
Season Start Month	٠	User Control Water	


Citex Diagnosis

All Disorder

alternaria_leaves_spot
alternaria_rot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly

User Suspected Disorder

Delete

Concepts

fruit_spots
fruits
leaf_spots
leaves

Properties

Values

Findings

existence of leaf_spots = yes

Why Delete What

Suspected Disorders

brown_mite
flat_mite
leafminer
manganese_def
rust_mite

How

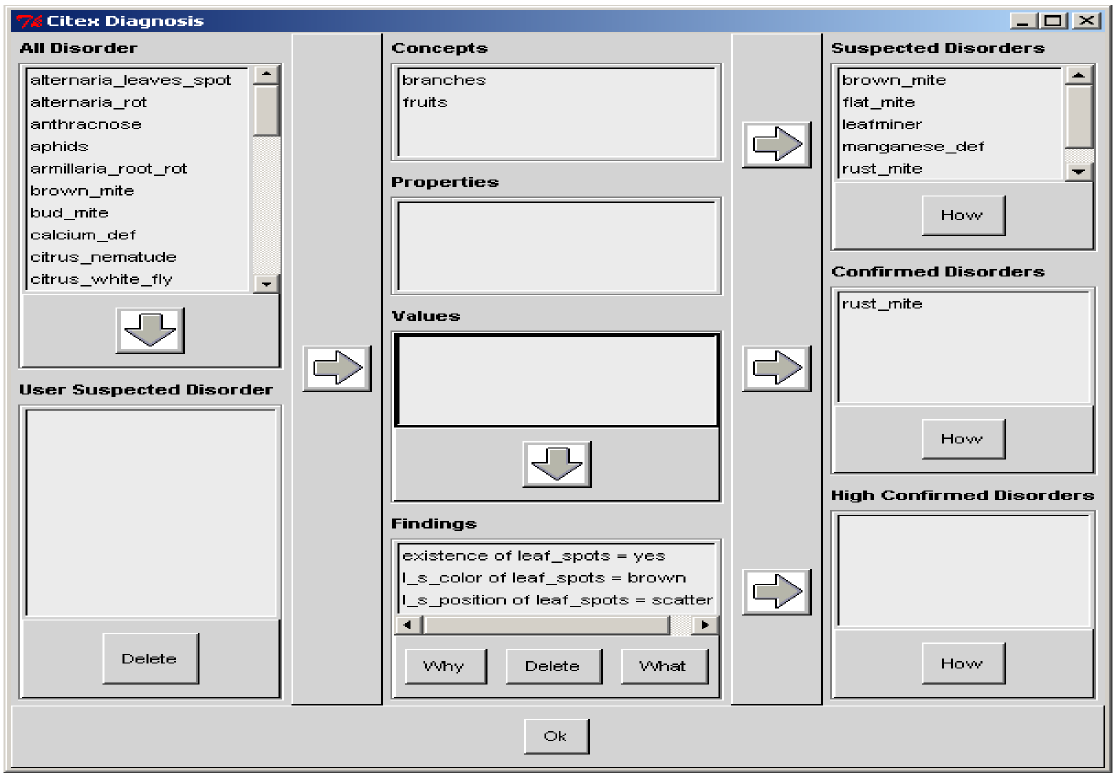
Confirmed Disorders

How

High Confirmed Disorders

How

Ok


Citex Diagnosis

All Disorder

alternaria_leaves_spot
alternaria_rot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly

User Suspected Disorder

Delete

Concepts

branches
fruits

Properties

Values

Findings

existence of leaf_spots = yes
l_s_color of leaf_spots = brown
l_s_position of leaf_spots = scatter

Why Delete What

Suspected Disorders

brown_mite
flat_mite
leafminer
manganese_def
rust_mite

How

Confirmed Disorders

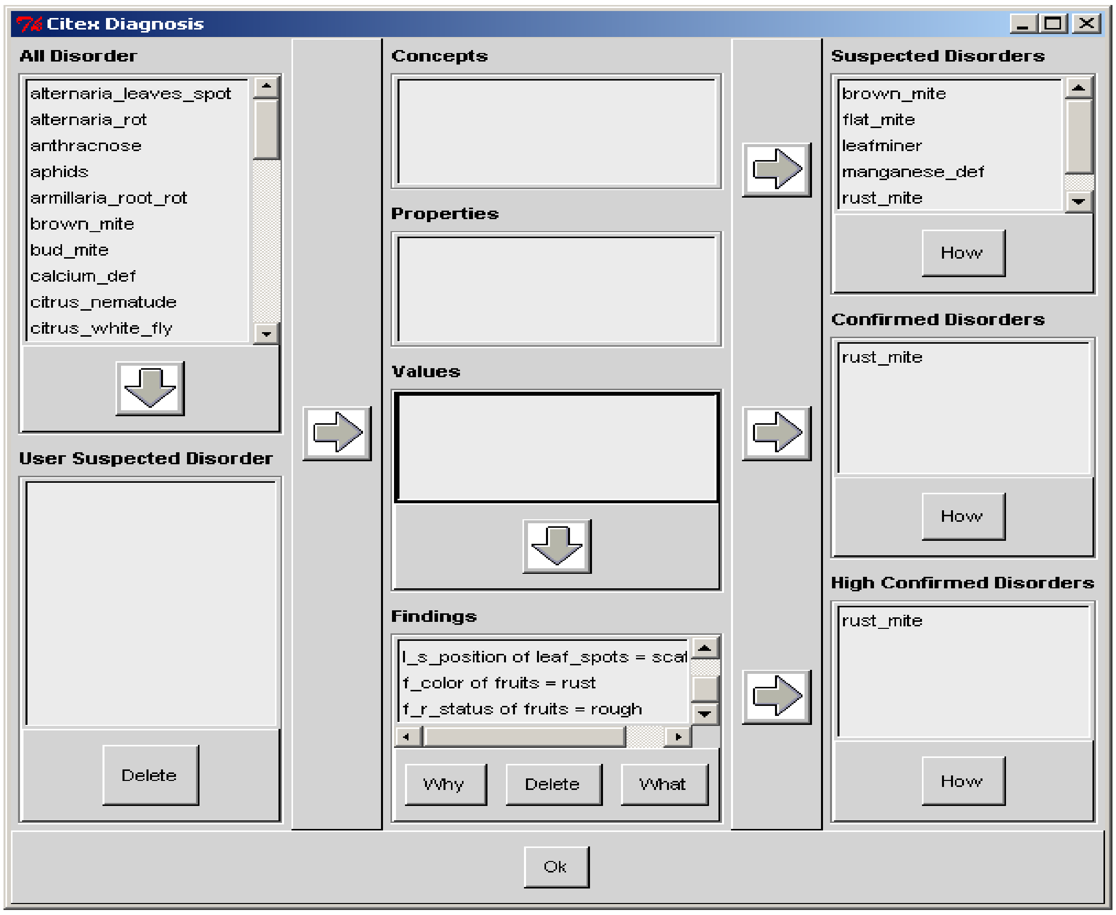
rust_mite

How

High Confirmed Disorders

How

Ok



```
l_s_position of leaf_spots = scat
f_color of fruits = rust
f_r_status of fruits = rough
```

6. Treatment subsystem

6.1. Relations between expressions

File name : treat_rules.pl

```
:- use_module(library(lists), [memberchk/2]).
:- ensure_loaded('$KROL/lib/rule_exp').
treated_by :: {
r1([     material_name(none)in stubborn,
         method(advice)in stubborn,
         number(1)in stubborn,
         date(Vv1) in stubborn]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(stubborn) in disorder &
r2([     material_name(none)in stubborn,
         method(advice)in stubborn,
         number(1)in stubborn,
         date(Vv1) in stubborn]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(stubborn) in disorder &
r3([     material_name(none)in impieetratura,
         method(advice)in impieetratura,
         number(1)in impieetratura,
         date(Vv1) in impieetratura]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(impieetratura) in disorder &
r4([     material_name(none)in impieetratura,
         method(advice)in impieetratura,
         number(1)in impieetratura,
         date(Vv1) in impieetratura]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(impieetratura) in disorder &
r5([     material_name(none)in anthracnose,
         method(advice)in anthracnose,
         number(1)in anthracnose,
         date(Vv1) in anthracnose]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(anthracnose) in disorder &
r6([     material_name(none)in anthracnose,
         method(advice)in anthracnose,
         number(1)in anthracnose,
         date(Vv1) in anthracnose]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(anthracnose) in disorder &
r7([     material_name(none)in alternaria_leaves_spot,
         method(advice)in alternaria_leaves_spot,
         number(1)in alternaria_leaves_spot,
         date(Vv1) in alternaria_leaves_spot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(alternaria_leaves_spot) in disorder &
r8([    material_name(none)in alternaria_leaves_spot,
         method(advice)in alternaria_leaves_spot,
         number(1)in alternaria_leaves_spot,
         date(Vv1) in alternaria_leaves_spot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(alternaria_leaves_spot) in disorder &
r9([    material_name(none)in alternaria_rot,
         method(advice)in alternaria_rot,
         number(1)in alternaria_rot,
         date(Vv1) in alternaria_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(alternaria_rot) in disorder &
r10([   material_name(none)in alternaria_rot,
         method(advice)in alternaria_rot,
         number(1)in alternaria_rot,
         date(Vv1) in alternaria_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(alternaria_rot) in disorder &
r11([   material_name(none)in gum_spots,
         method(advice)in gum_spots,
         number(1)in gum_spots,
         date(Vv1) in gum_spots]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(gum_spots) in disorder &
r12([   material_name(none)in gum_spots,
         method(advice)in gum_spots,
         number(1)in gum_spots,
         date(Vv1) in gum_spots]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(gum_spots) in disorder &
r13([   material_name(none)in sun_burn,
         method(advice)in sun_burn,
         number(1)in sun_burn,
         date(Vv1) in sun_burn]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(sun_burn) in disorder &
r14([   material_name(none)in sun_burn,
         method(advice)in sun_burn,
         number(1)in sun_burn,
         date(Vv1) in sun_burn]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(sun_burn) in disorder &
r15([   material_name(none)in salt_injury,
         method(advice)in salt_injury,
         number(1)in salt_injury,
         date(Vv1) in salt_injury]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(salt_injury) in disorder &
r16([   material_name(none)in salt_injury,
         method(advice)in salt_injury,
         number(1)in salt_injury,
         date(Vv1) in salt_injury]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(salt_injury) in disorder &
r17([   material_name(none)in rose_scarab,
         method(advice)in rose_scarab,
         number(1)in rose_scarab,
         date(Vv1) in rose_scarab]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(rose_scarab) in disorder &
r18([   material_name(none)in rose_scarab,
         method(advice)in rose_scarab,
         number(1)in rose_scarab,
         date(Vv1) in rose_scarab]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(rose_scarab) in disorder &
r19([   material_name(none)in green_stink_bug,
         method(advice)in green_stink_bug,
         number(1)in green_stink_bug,
         date(Vv1) in green_stink_bug]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(green_stink_bug) in disorder &
r20([   material_name(none)in green_stink_bug,
         method(advice)in green_stink_bug,
         number(1)in green_stink_bug,
         date(Vv1) in green_stink_bug]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(green_stink_bug) in disorder &
r21([   material_name(none)in psorosis,
         method(advice)in psorosis,
         number(1)in psorosis,
         date(Vv1) in psorosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(psorosis) in disorder &
r22([   material_name(none)in psorosis,
         method(advice)in psorosis,
         number(1)in psorosis,
         date(Vv1) in psorosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(psorosis) in disorder &
r23([   material_name(none)in armillaria_root_rot,
         method(advice)in armillaria_root_rot,
         number(1)in armillaria_root_rot,
         date(Vv1) in armillaria_root_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(armillaria_root_rot) in disorder &
r24([   material_name(none)in armillaria_root_rot,
         method(advice)in armillaria_root_rot,
         number(1)in armillaria_root_rot,
         date(Vv1) in armillaria_root_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(armillaria_root_rot) in disorder &
r25([   material_name(none)in fruit_cracking,
         method(advice)in fruit_cracking,
         number(1)in fruit_cracking,
         date(Vv1) in fruit_cracking]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(fruit_cracking) in disorder &
r26([   material_name(none)in fruit_cracking,
         method(advice)in fruit_cracking,
         number(1)in fruit_cracking,
         date(Vv1) in fruit_cracking]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(fruit_cracking) in disorder &
r27([   material_name(none)in fruit_creasing,
         method(advice)in fruit_creasing,
         number(1)in fruit_creasing,
         date(Vv1) in fruit_creasing]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(fruit_creasing) in disorder &
r28([   material_name(none)in fruit_creasing,
         method(advice)in fruit_creasing,
         number(1)in fruit_creasing,
         date(Vv1) in fruit_creasing]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(fruit_creasing) in disorder &
r29([   material_name(none)in sooty_mold,
         method(advice)in sooty_mold,
         number(1)in sooty_mold,
         date(Vv1) in sooty_mold]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(sooty_mold) in disorder &
r30([   material_name(none)in sooty_mold,
         method(advice)in sooty_mold,
         number(1)in sooty_mold,
         date(Vv1) in sooty_mold]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(sooty_mold) in disorder &
r31([   material_name(potassium_permenganat)in gummosis,
         method(disinfection)in gummosis,
         number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(gummosis) in disorder &
r32([   material_name(potassium_permenganat)in gummosis,
         method(disinfection)in gummosis,
         number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(gummosis) in disorder &
r33a([  material_name(topsin)in ganoderma_rot_op1,
         method('chemical spray')in ganoderma_rot_op1,
         number(1)in ganoderma_rot_op1,
         date(Vv1) in ganoderma_rot_op1]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(ganoderma_rot) in disorder &
r33b([  method('chemical spray')in ganoderma_rot_op2,
         material_name('bordeaux past')in ganoderma_rot_op2,
         number(2)in ganoderma_rot_op2,
         date(Vv1) in ganoderma_rot_op2]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(ganoderma_rot) in disorder &
r34a([  material_name(topsin)in ganoderma_rot_op1,
         method('chemical spray')in ganoderma_rot_op1,
         number(1)in ganoderma_rot_op1,
         date(Vv1) in ganoderma_rot_op1]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(ganoderma_rot) in disorder &
r34b([  method('chemical spray')in ganoderma_rot_op2,
         material_name('bordeaux past')in ganoderma_rot_op2,
         number(2)in ganoderma_rot_op2,
         date(Vv1) in ganoderma_rot_op2]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(ganoderma_rot) in disorder &
r35a([  material_name(topsin)in wilt_root_rot_op1,
         method('soil treatment')in wilt_root_rot_op1,
         number(1)in wilt_root_rot_op1,
         date(Vv1) in wilt_root_rot_op1]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(wilt_root_rot) in disorder &
r35b([  material_name(topsin)in wilt_root_rot_op2,
         method('soil treatment')in wilt_root_rot_op2,
         number(2)in wilt_root_rot_op2,
         date(Vv1) in wilt_root_rot_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv1),
         confirmed(wilt_root_rot) in disorder &
r36a([  material_name(topsin)in wilt_root_rot_op1,
         method('soil treatment')in wilt_root_rot_op1,
         number(1)in wilt_root_rot_op1,
         date(Vv1) in wilt_root_rot_op1]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(wilt_root_rot) in disorder &
r36b([  material_name(topsin)in wilt_root_rot_op2,
         method('soil treatment')in wilt_root_rot_op2,
         number(2)in wilt_root_rot_op2,
         date(Vv1) in wilt_root_rot_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv1),
         highly_confirmed(wilt_root_rot) in disorder &
r37([   material_name('vertimec 1.8%')in citrus_white_fly,
         method('chemical spray')in citrus_white_fly,
         number(1)in citrus_white_fly,
         date(Vv1) in citrus_white_fly]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(citrus_white_fly) in disorder,
         confirmed(_55189) in disorder,
         :( \+ memberchk(aphids, _55189)) &
r38([   material_name('vertimec 1.8%')in citrus_white_fly,
         method('chemical spray')in citrus_white_fly,
         number(1)in citrus_white_fly,
         date(Vv1) in citrus_white_fly]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(citrus_white_fly) in disorder,
         highly_confirmed(_56825) in disorder,
         :( \+ memberchk(aphids, _56825)) &
r39([   method('chemical spray')in aphids,
         method('chemical spray')in citrus_white_fly,
         number(1)in aphids,
         number(1)in citrus_white_fly,
         date(Vv1) in aphids,
         date(Vv1) in citrus_white_fly,
         material_name(Vv2) in aphids,
         material_name(Vv2) in citrus_white_fly]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(aphids) in disorder,
         confirmed(citrus_white_fly) in disorder,
         material_gr1(Vv2) in operation &
r40([   method('chemical spray')in aphids,
         method('chemical spray')in citrus_white_fly,
         number(1)in aphids,
         number(1)in citrus_white_fly,
         material_name(Vv1) in aphids,
         material_name(Vv1) in citrus_white_fly,
         date(Vv2) in aphids,
         date(Vv2) in citrus_white_fly]) if
         :eval_rule_exp(current_date of plant, Vv2),
         highly_confirmed(aphids) in disorder,
         highly_confirmed(citrus_white_fly) in disorder,
         material_gr1(Vv1) in operation &
r41([   material_name('malathion 57%')in aphids,
         method('chemical spray')in aphids,
         number(1)in aphids,
         date(Vv1) in aphids]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(aphids) in disorder,
         confirmed(_62849) in disorder,
         :( \+ memberchk(citrus_white_fly, _62849)) &
r42([   material_name('malathion 57%')in aphids,
         method('chemical spray')in aphids,
         number(1)in aphids,
         date(Vv1) in aphids]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(aphids) in disorder,
         highly_confirmed(_64485) in disorder,
         :( \+ memberchk(citrus_white_fly, _64485)) &
r43([   method('chemical spray')in citrus_flower_moth,
         number(1)in citrus_flower_moth,
         material_name(Vv1) in citrus_flower_moth,
         date(Vv2) in citrus_flower_moth]) if
         :eval_rule_exp(current_date of plant, Vv2),
         confirmed(citrus_flower_moth) in disorder,
         material_gr2(Vv1) in operation &
r44([   method('chemical spray')in citrus_flower_moth,
         number(1)in citrus_flower_moth,
         material_name(Vv1) in citrus_flower_moth,
         date(Vv2) in citrus_flower_moth]) if
         :eval_rule_exp(current_date of plant, Vv2),
         highly_confirmed(citrus_flower_moth) in disorder,
         material_gr2(Vv1) in operation &
r45([   method('chemical spray')in lichens,
         number(1)in lichens,
         material_name(Vv1) in lichens,
         date(Vv2) in lichens]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         confirmed(lichens) in disorder,
         material_gr3(Vv1) in operation &
r46([   method('chemical spray')in lichens,
         number(1)in lichens,
         material_name(Vv1) in lichens,
         date(Vv2) in lichens]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         highly_confirmed(lichens) in disorder,
         material_gr3(Vv1) in operation &
r47([   method('chemical spray')in lichens,
         number(1)in lichens,
         special_date('next 1/12')in lichens,
         material_name(Vv1) in lichens]) if
         confirmed(lichens) in disorder,
         season(_72183) in plant,        :(_72183\==winter),
         material_gr3(Vv1) in operation &
r48([   method('chemical spray')in lichens,
         number(1)in lichens,
         special_date('next 1/12')in lichens,
         material_name(Vv1) in lichens]) if
         season(_73566) in plant,        :(_73566\==winter),
         highly_confirmed(lichens) in disorder,
         material_gr3(Vv1) in operation &
r49([   material_name('bordeaux past')in gummosis,
         method(painting)in gummosis,
         number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(winter) in plant,
         confirmed(gummosis) in disorder &
r50([   material_name('bordeaux past')in gummosis,
         method(painting)in gummosis,
         number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(winter) in plant,
         highly_confirmed(gummosis) in disorder &
r51([   method('chemical spray')in scales,
         number(1)in scales,
         material_name(Vv1) in scales,
         date(Vv2) in scales]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(scales) in disorder,
         material_gr1(Vv1) in operation &
r52([   method('chemical spray')in scales,
         number(1)in scales,
         material_name(Vv1) in scales,
         date(Vv2) in scales]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(scales) in disorder,
         material_gr1(Vv1) in operation &
r53([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         material_name(Vv1) in mealy_bug,
         date(Vv2) in mealy_bug]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(mealy_bug) in disorder,
         material_gr1(Vv1) in operation &
r54([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         material_name(Vv1) in mealy_bug,
         date(Vv2) in mealy_bug]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(mealy_bug) in disorder,
         material_gr1(Vv1) in operation &
r55([   method('chemical spray')in scales,
         number(1)in scales,
         special_date('next summer')in scales,
         material_name(Vv1) in scales]) if
         season(spring) in plant,
         confirmed(scales) in disorder,
         material_gr1(Vv1) in operation &
r56([   method('chemical spray')in scales,
         number(1)in scales,
         special_date('next summer')in scales,
         material_name(Vv1) in scales]) if
         season(spring) in plant,
         highly_confirmed(scales) in disorder,
         material_gr1(Vv1) in operation &
r57([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         special_date('next summer')in mealy_bug,
         material_name(Vv1) in mealy_bug]) if
         season(spring) in plant,
         confirmed(mealy_bug) in disorder,
         material_gr1(Vv1) in operation &
r58([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         special_date('next summer')in mealy_bug,
         material_name(Vv1) in mealy_bug]) if
         season(spring) in plant,
         highly_confirmed(mealy_bug) in disorder,
         material_gr1(Vv1) in operation &
r59([   method('chemical spray')in scales,
         number(1)in scales,
         material_name(Vv1) in scales,
         date(Vv2) in scales]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         confirmed(scales) in disorder,
         material_gr4(Vv1) in operation &
r60([   method('chemical spray')in scales,
         number(1)in scales,
         material_name(Vv1) in scales,
         date(Vv2) in scales]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         highly_confirmed(scales) in disorder,
         material_gr4(Vv1) in operation &
r61([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         material_name(Vv1) in mealy_bug,
         date(Vv2) in mealy_bug]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         confirmed(mealy_bug) in disorder,
         material_gr4(Vv1) in operation &
r62([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         material_name(Vv1) in mealy_bug,
         date(Vv2) in mealy_bug]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(winter) in plant,
         highly_confirmed(mealy_bug) in disorder,
         material_gr4(Vv1) in operation &
r63([   method('chemical spray')in scales,
         number(1)in scales,
         special_date('next winter')in scales,
         material_name(Vv1) in scales]) if
         season(autumn) in plant,
         confirmed(scales) in disorder,
         material_gr4(Vv1) in operation &
r64([   method('chemical spray')in scales,
         number(1)in scales,
         special_date('next winter')in scales,
         material_name(Vv1) in scales]) if
         season(autumn) in plant,
         highly_confirmed(scales) in disorder,
         material_gr4(Vv1) in operation &
r65([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         special_date('next winter')in mealy_bug,
         material_name(Vv1) in mealy_bug]) if
         season(autumn) in plant,
         confirmed(mealy_bug) in disorder,
         material_gr4(Vv1) in operation &
r66([   method('chemical spray')in mealy_bug,
         number(1)in mealy_bug,
         special_date('next winter')in mealy_bug,
         material_name(Vv1) in mealy_bug]) if
         season(autumn) in plant,
         highly_confirmed(mealy_bug) in disorder,
         material_gr4(Vv1) in operation &
r67a([  method('chemical spray')in leafminer_op1,
         number(1)in leafminer_op1,
         material_name(Vv1) in leafminer_op1,
         date(Vv2) in leafminer_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r67b([  method('chemical spray')in leafminer_op2,
         number(2)in leafminer_op2,
         material_name(Vv1) in leafminer_op2,
         date(Vv2) in leafminer_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         season(summer) in plant,
         confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r67c([  method('chemical spray')in leafminer_op3,
         number(3)in leafminer_op3,
         material_name(Vv1) in leafminer_op3,
         date(Vv2) in leafminer_op3]) if
         :eval_rule_exp(current_date of plant+42, Vv2),
         season(summer) in plant,
         confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r68a([  method('chemical spray')in leafminer_op1,
         number(1)in leafminer_op1,
         material_name(Vv1) in leafminer_op1,
         date(Vv2) in leafminer_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r68b([  method('chemical spray')in leafminer_op2,
         number(2)in leafminer_op2,
         material_name(Vv1) in leafminer_op2,
         date(Vv2) in leafminer_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         season(summer) in plant,
         highly_confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r68c([  method('chemical spray')in leafminer_op3,
         number(3)in leafminer_op3,
         material_name(Vv1) in leafminer_op3,
         date(Vv2) in leafminer_op3]) if
         :eval_rule_exp(current_date of plant+42, Vv2),
         season(summer) in plant,
         highly_confirmed(leafminer) in disorder,
         material_gr5(Vv1) in operation &
r69a([  method('chemical spray')in leafminer_op1,
         number(1)in leafminer_op1,
         special_date('next 1/6')in leafminer_op1,
         material_name(Vv1) in leafminer_op1]) if
         confirmed(leafminer) in disorder,
         season(_112909) in plant,      :(_112909\==summer),
         material_gr5(Vv1) in operation &
r69b([  method('chemical spray')in leafminer_op2,
         number(2)in leafminer_op2,
         special_date('next 22/6')in leafminer_op2,
         material_name(Vv1) in leafminer_op2]) if
         confirmed(leafminer) in disorder,
         season(_114448) in plant,      :(_114448\==summer),
         material_gr5(Vv1) in operation &
r69c([  method('chemical spray')in leafminer_op3,
         number(3)in leafminer_op3,
         special_date('next 13/7')in leafminer_op3,
         material_name(Vv1) in leafminer_op3]) if
         confirmed(leafminer) in disorder,
         season(_115987) in plant,      :(_115987\==summer),
         material_gr5(Vv1) in operation &
r70a([  method('chemical spray')in leafminer_op1,
         number(1)in leafminer_op1,
         special_date('next 1/6')in leafminer_op1,
         material_name(Vv1) in leafminer_op1]) if
         highly_confirmed(leafminer) in disorder,
         season(_119065) in plant,      :(_119065\==summer),
         material_gr5(Vv1) in operation &
r70b([  method('chemical spray')in leafminer_op2,
         number(1)in leafminer_op2,
         special_date('next 22/6')in leafminer_op2,
         material_name(Vv1) in leafminer_op2]) if
         highly_confirmed(leafminer) in disorder,
         season(_120604) in plant,      :(_120604\==summer),
         material_gr5(Vv1) in operation &
r70c([  method('chemical spray')in leafminer_op3,
         number(1)in leafminer_op3,
         special_date('next 13/7')in leafminer_op3,
         material_name(Vv1) in leafminer_op3]) if
         highly_confirmed(leafminer) in disorder,
         season(_122143) in plant,      :(_122143\==summer),
         material_gr5(Vv1) in operation &
r71a([  method('chemical spray')in rust_mite_op1,
         number(1)in rust_mite_op1,
         material_name(Vv1) in rust_mite_op1,
         date(Vv2) in rust_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(rust_mite) in disorder,
         material_gr6(Vv1) in operation &
r71b([  method('chemical spray')in rust_mite_op2,
         number(2)in rust_mite_op2,
         material_name(Vv1) in rust_mite_op2,
         date(Vv2) in rust_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         confirmed(rust_mite) in disorder,
         material_gr6(Vv1) in operation &
r72a([  method('chemical spray')in rust_mite_op1,
         number(1)in rust_mite_op1,
         material_name(Vv1) in rust_mite_op1,
         date(Vv2) in rust_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(rust_mite) in disorder,
         material_gr6(Vv1) in operation &
r72b([  method('chemical spray')in rust_mite_op2,
         number(2)in rust_mite_op2,
         material_name(Vv1) in rust_mite_op2,
         date(Vv2) in rust_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         highly_confirmed(rust_mite) in disorder,
         material_gr6(Vv1) in operation &
r73([   material_name(none)in rust_mite,
         method(advice)in rust_mite,
         number(1)in rust_mite,
         date(Vv1) in rust_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_130378) in plant,      :(_130378\==summer),
         confirmed(rust_mite) in disorder &
r74([   material_name(none)in rust_mite,
         method(advice)in rust_mite,
         number(1)in rust_mite,
         date(Vv1) in rust_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_131949) in plant,      :(_131949\==summer),
         highly_confirmed(rust_mite) in disorder &
r75a([  method('chemical spray')in bud_mite_op1,
         number(1)in bud_mite_op1,
         material_name(Vv1) in bud_mite_op1,
         date(Vv2) in bud_mite_op1]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         confirmed(bud_mite) in disorder,
         current_week(_133738) in plant,      :(_133738>=7),
         current_week(_134001) in plant,      :(_134001=<22),
         material_gr6(Vv1) in operation &
r75b([  method('chemical spray')in bud_mite_op2,
         number(2)in bud_mite_op2,
         material_name(Vv1) in bud_mite_op2,
         date(Vv2) in bud_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         confirmed(bud_mite) in disorder,
         current_week(_135790) in plant,      :(_135790>=7),
         current_week(_136053) in plant,      :(_136053=<22),
         material_gr6(Vv1) in operation &
r75c([  method('chemical spray')in bud_mite_op1,
         number(1)in bud_mite_op1,
         material_name(Vv1) in bud_mite_op1,
         date(Vv2) in bud_mite_op1]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         confirmed(bud_mite) in disorder,
         current_week(_137842) in plant,      :(_137842>=35),
         current_week(_138105) in plant,      :(_138105=<44),
         material_gr6(Vv1) in operation &
r75d([  method('chemical spray')in bud_mite_op2,
         number(2)in bud_mite_op2,
         material_name(Vv1) in bud_mite_op2,
         date(Vv2) in bud_mite_op2]) if
```

```
         :eval_rule_exp(current_date of plant+15, Vv2),
         confirmed(bud_mite) in disorder,
         current_week(_139894) in plant,      :(_139894>=35),
         current_week(_140157) in plant,      :(_140157=<44),
         material_gr6(Vv1) in operation &
r76a([    method('chemical spray')in bud_mite_op1,
         number(1)in bud_mite_op1,
         material_name(Vv1) in bud_mite_op1,
         date(Vv2) in bud_mite_op1]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         highly_confirmed(bud_mite) in disorder,
         current_week(_141946) in plant,      :(_141946>=7),
         current_week(_142209) in plant,      :(_142209=<22),
         material_gr6(Vv1) in operation &
r76b([    method('chemical spray')in bud_mite_op1,
         number(1)in bud_mite_op1,
         material_name(Vv1) in bud_mite_op1,
         date(Vv2) in bud_mite_op1]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         highly_confirmed(bud_mite) in disorder,
         current_week(_144258) in plant,      :(_144258>=35),
         current_week(_144521) in plant,      :(_144521=<44),
         material_gr6(Vv1) in operation &
r76c([    method('chemical spray')in bud_mite_op2,
         number(2)in bud_mite_op2,
         material_name(Vv1) in bud_mite_op2,
         date(Vv2) in bud_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         highly_confirmed(bud_mite) in disorder,
         current_week(_146310) in plant,      :(_146310>=7),
         current_week(_146573) in plant,      :(_146573=<22),
         material_gr6(Vv1) in operation &
r76d([    method('chemical spray')in bud_mite_op2,
         number(2)in bud_mite_op2,
         material_name(Vv1) in bud_mite_op2,
         date(Vv2) in bud_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         highly_confirmed(bud_mite) in disorder,
         current_week(_148362) in plant,      :(_148362>=35),
         current_week(_148625) in plant,      :(_148625=<44),
         material_gr6(Vv1) in operation &
r77a([    material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(bud_mite) in disorder,
         current_week(_150378) in plant,      :(_150378>0),
         current_week(_150641) in plant,      :(_150641<7) &
r77b([    material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(bud_mite) in disorder,
         current_week(_152238) in plant,     :(_152238>22),
         current_week(_152501) in plant,     :(_152501<35) &
r77c([  material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(bud_mite) in disorder,
         current_week(_154098) in plant,     :(_154098>44),
         current_week(_154361) in plant,     :(_154361=<52) &
r78a([  material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(bud_mite) in disorder,
         current_week(_155958) in plant,     :(_155958>0),
         current_week(_156221) in plant,     :(_156221<7) &
r78b([  material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(bud_mite) in disorder,
         current_week(_157818) in plant,     :(_157818>22),
         current_week(_158081) in plant,     :(_158081<35) &
r78c([  material_name(none)in bud_mite,
         method(advice)in bud_mite,
         number(1)in bud_mite,
         date(Vv1) in bud_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(bud_mite) in disorder,
         current_week(_159678) in plant,     :(_159678>44),
         current_week(_159941) in plant,     :(_159941=<52) &
r79a([  method('chemical spray')in brown_mite_op1,
         number(1)in brown_mite_op1,
         material_name(Vv1) in brown_mite_op1,
         date(Vv2) in brown_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(brown_mite) in disorder,
         material_gr7(Vv1) in operation &
r79b([  method('chemical spray')in brown_mite_op2,
         number(2)in brown_mite_op2,
         material_name(Vv1) in brown_mite_op2,
         date(Vv2) in brown_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         confirmed(brown_mite) in disorder,
         material_gr7(Vv1) in operation &
r80a([  method('chemical spray')in brown_mite_op1,
         number(1)in brown_mite_op1,
         material_name(Vv1) in brown_mite_op1,
         date(Vv2) in brown_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(brown_mite) in disorder,
         material_gr7(Vv1) in operation &
r80b([  method('chemical spray')in brown_mite_op2,
         number(2)in brown_mite_op2,
         material_name(Vv1) in brown_mite_op2,
         date(Vv2) in brown_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         highly_confirmed(brown_mite) in disorder,
         material_gr7(Vv1) in operation &
r81([   material_name(none)in brown_mite,
         method(advice)in brown_mite,
         number(1)in brown_mite,
         date(Vv1) in brown_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_167980) in plant,      :(_167980\==summer),
         confirmed(brown_mite) in disorder &
r82([   material_name(none)in brown_mite,
         method(advice)in brown_mite,
         number(1)in brown_mite,
         date(Vv1) in brown_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_169551) in plant,      :(_169551\==summer),
         highly_confirmed(brown_mite) in disorder &
r83a([  method('chemical spray')in flat_mite_op1,
         number(1)in flat_mite_op1,
         material_name(Vv1) in flat_mite_op1,
         date(Vv2) in flat_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         confirmed(flat_mite) in disorder,
         material_gr7(Vv1) in operation &
r83b([  method('chemical spray')in flat_mite_op2,
         number(2)in flat_mite_op2,
         material_name(Vv1) in flat_mite_op2,
         date(Vv2) in flat_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         confirmed(flat_mite) in disorder,
         material_gr7(Vv1) in operation &
r84a([    method('chemical spray')in flat_mite_op1,
         number(1)in flat_mite_op1,
         material_name(Vv1) in flat_mite_op1,
         date(Vv2) in flat_mite_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(summer) in plant,
         highly_confirmed(flat_mite) in disorder,
         material_gr7(Vv1) in operation &
r84b([    method('chemical spray')in flat_mite_op2,
         number(2)in flat_mite_op2,
         material_name(Vv1) in flat_mite_op2,
         date(Vv2) in flat_mite_op2]) if
         :eval_rule_exp(current_date of plant+15, Vv2),
         season(summer) in plant,
         highly_confirmed(flat_mite) in disorder,
         material_gr7(Vv1) in operation &
r85([     material_name(none)in flat_mite,
         method(advice)in flat_mite,
         number(1)in flat_mite,
         date(Vv1) in flat_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_177746) in plant,      :(_177746\==summer),
         confirmed(flat_mite) in disorder &
r86([     material_name(none)in flat_mite,
         method(advice)in flat_mite,
         number(1)in flat_mite,
         date(Vv1) in flat_mite]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_179317) in plant,      :(_179317\==summer),
         highly_confirmed(flat_mite) in disorder &
r87a1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1,
         date(Vv2) in citrus_nematude_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         confirmed(citrus_nematude) in disorder,
         current_month(2) in plant,
         material_gr8(Vv1) in operation &
r87a2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2,
         date(Vv2) in citrus_nematude_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         confirmed(citrus_nematude) in disorder,
         current_month(2) in plant,
         material_gr8(Vv1) in operation &
r87b1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1,
         date(Vv2) in citrus_nematude_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         confirmed(citrus_nematude) in disorder,
         current_month(3) in plant,
         material_gr8(Vv1) in operation &
r87b2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2,
         date(Vv2) in citrus_nematude_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         confirmed(citrus_nematude) in disorder,
         current_month(3) in plant,
         material_gr8(Vv1) in operation &
r88a1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1,
         date(Vv2) in citrus_nematude_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         highly_confirmed(citrus_nematude) in disorder,
         current_month(2) in plant,
         material_gr8(Vv1) in operation &
r88a2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2,
         date(Vv2) in citrus_nematude_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         highly_confirmed(citrus_nematude) in disorder,
         current_month(2) in plant,
         material_gr8(Vv1) in operation &
r88b1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1,
         date(Vv2) in citrus_nematude_op1]) if
         :eval_rule_exp(current_date of plant, Vv2),
         highly_confirmed(citrus_nematude) in disorder,
         current_month(3) in plant,
         material_gr8(Vv1) in operation &
r88b2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2,
         date(Vv2) in citrus_nematude_op2]) if
         :eval_rule_exp(current_date of plant+21, Vv2),
         highly_confirmed(citrus_nematude) in disorder,
         current_month(3) in plant,
         material_gr8(Vv1) in operation &
r89a1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         special_date('next 1/2')in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1]) if
         confirmed(citrus_nematude) in disorder,
         current_month(_187480) in plant,     :(_187480\==2),
         material_gr8(Vv1) in operation &
r89a2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         special_date('next 22/2')in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2]) if
         confirmed(citrus_nematude) in disorder,
         current_month(_187480) in plant,     :(_187480\==2),
         material_gr8(Vv1) in operation &
r89b1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         special_date('next 1/2')in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1]) if
         confirmed(citrus_nematude) in disorder,
         current_month(_189019) in plant,     :(_189019\==3),
         material_gr8(Vv1) in operation &
r89b2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         special_date('next 22/2')in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2]) if
         confirmed(citrus_nematude) in disorder,
         current_month(_189019) in plant,     :(_189019\==3),
         material_gr8(Vv1) in operation &
r90a1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         special_date('next 1/2')in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1]) if
         highly_confirmed(citrus_nematude) in disorder,
         current_month(_190558) in plant,     :(_190558\==2),
         material_gr8(Vv1) in operation &
r90a2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         special_date('next 22/2')in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2]) if
         highly_confirmed(citrus_nematude) in disorder,
         current_month(_190558) in plant,     :(_190558\==2),
         material_gr8(Vv1) in operation &
r90b1([method('soil treatment')in citrus_nematude_op1,
         number(1)in citrus_nematude_op1,
         special_date('next 1/2')in citrus_nematude_op1,
         material_name(Vv1) in citrus_nematude_op1]) if
         highly_confirmed(citrus_nematude) in disorder,
         current_month(_192097) in plant,     :(_192097\==3),
         material_gr8(Vv1) in operation &
r90b2([method('soil treatment')in citrus_nematude_op2,
         number(2)in citrus_nematude_op2,
         special_date('next 22/2')in citrus_nematude_op2,
         material_name(Vv1) in citrus_nematude_op2]) if
         highly_confirmed(citrus_nematude) in disorder,
         current_month(_192097) in plant,     :(_192097\==3),
         material_gr8(Vv1) in operation &
r91([   method('foliage nutrition')in nitrogen_def,
         number(1)in nitrogen_def,
         material_name(Vv1) in nitrogen_def,
         date(Vv2) in nitrogen_def]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(_193704) in plant,      :(_193704\==winter),
         confirmed(nitrogen_def) in disorder,
         material_gr9(Vv1) in operation &
r92([   method('foliage nutrition')in nitrogen_def,
         number(1)in nitrogen_def,
         material_name(Vv1) in nitrogen_def,
         date(Vv1) in nitrogen_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_195467) in plant,      :(_195467\==winter),
         highly_confirmed(nitrogen_def) in disorder,
         material_gr9(Vv1) in operation &
r93([   material_name('triple phosphate')in phosphorus_def,
         method('foliage nutrition')in phosphorus_def,
         number(1)in phosphorus_def,
         date(Vv1) in phosphorus_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_197194) in plant,      :(_197194\==winter),
         confirmed(phosphorus_def) in disorder &
r94([   material_name('triple phosphate')in phosphorus_def,
         method('foliage nutrition')in phosphorus_def,
         number(1)in phosphorus_def,
         date(Vv1) in phosphorus_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_198765) in plant,      :(_198765\==winter),
         highly_confirmed(phosphorus_def) in disorder &
r95([   method('foliage nutrition')in potassium_def,
         number(1)in potassium_def,
         material_name(Vv1) in potassium_def,
         date(Vv2) in potassium_def]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(_200372) in plant,      :(_200372\==winter),
         confirmed(potassium_def) in disorder,
         material_gr10(Vv1) in operation &
r96([   method('foliage nutrition')in potassium_def,
         number(1)in potassium_def,
         material_name(Vv1) in potassium_def,
         date(Vv2) in potassium_def]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(_202135) in plant,      :(_202135\==winter),
         highly_confirmed(potassium_def) in disorder,
         material_gr10(Vv1) in operation &
r97([   material_name(magnesium_sulfate)in magnesium_def,
         method('foliage nutrition')in magnesium_def,
         number(1)in magnesium_def,
         date(Vv1) in magnesium_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         confirmed(magnesium_def) in disorder &
r98([   material_name(magnesium_sulfate)in magnesium_def,
         method('foliage nutrition')in magnesium_def,
         number(1)in magnesium_def,
         date(Vv1) in magnesium_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         highly_confirmed(magnesium_def) in disorder &
r99([   material_name('micro element mixture')in manganese_def,
         method('foliage nutrition')in manganese_def,
         number(1)in manganese_def,
         date(Vv1) in manganese_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         confirmed(manganese_def) in disorder &
r100([  material_name('micro element mixture')in manganese_def,
         method('foliage nutrition')in manganese_def,
         number(1)in manganese_def,
         date(Vv1) in manganese_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         highly_confirmed(manganese_def) in disorder &
r101([  material_name('micro element mixture')in iron_def,
         method('foliage nutrition')in iron_def,
         number(1)in iron_def,
         date(Vv1) in iron_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         confirmed(iron_def) in disorder &
r102([  material_name('micro element mixture')in iron_def,
         method('foliage nutrition')in iron_def,
         number(1)in iron_def,
         date(Vv1) in iron_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         highly_confirmed(iron_def) in disorder &
r103([  method('foliage nutrition')in calcium_def,
         number(1)in calcium_def,
         material_name(Vv1) in calcium_def,
         date(Vv2) in calcium_def]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(_212682) in plant,     :(_212682\==winter),
         confirmed(calcium_def) in disorder,
         material_gr11(Vv1) in operation &
r104([  method('foliage nutrition')in calcium_def,
         number(1)in calcium_def,
         material_name(Vv1) in calcium_def,
         date(Vv2) in calcium_def]) if
         :eval_rule_exp(current_date of plant, Vv2),
         season(_214445) in plant,      :(_214445\==winter),
         highly_confirmed(calcium_def) in disorder,
         material_gr11(Vv1) in operation &
r105([  method('foliage nutrition')in zinc_def,
         material_name('micro element mixture')in zinc_def,
         number(1)in zinc_def,
         date(Vv1) in zinc_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         confirmed(zinc_def) in disorder &
r106([  method('foliage nutrition')in zinc_def,
         material_name('micro element mixture')in zinc_def,
         number(1)in zinc_def,
         date(Vv1) in zinc_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(summer) in plant,
         highly_confirmed(zinc_def) in disorder &
r107([  material_name(none)in zinc_def,
         method(advice)in zinc_def,
         number(1)in zinc_def,
         date(Vv1) in zinc_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_219100) in plant,      :(_219100\==summer),
         confirmed(zinc_def) in disorder &
r108([  material_name(none)in zinc_def,
         method(advice)in zinc_def,
         number(1)in zinc_def,
         date(Vv1) in zinc_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_220671) in plant,      :(_220671\==summer),
         highly_confirmed(zinc_def) in disorder &
r109([  material_name(none)in iron_def,
         method(advice)in iron_def,
         number(1)in iron_def,
         date(Vv1) in iron_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_222242) in plant,      :(_222242\==summer),
         confirmed(iron_def) in disorder &
r110([  material_name(none)in iron_def,
         method(advice)in iron_def,
         number(1)in iron_def,
         date(Vv1) in iron_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_223813) in plant,      :(_223813\==summer),
         highly_confirmed(iron_def) in disorder &
r111([  material_name(none)in manganese_def,
         method(advice)in manganese_def,
         number(1)in manganese_def,
         date(Vv1) in manganese_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_225384) in plant,      :(_225384\==summer),
         confirmed(manganese_def) in disorder &
r112([   material_name(none)in manganese_def,
         method(advice)in manganese_def,
         number(1)in manganese_def,
         date(Vv1) in manganese_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_226955) in plant,      :(_226955\==summer),
         highly_confirmed(manganese_def) in disorder &
r113([   material_name(none)in magnesium_def,
         method(advice)in magnesium_def,
         number(1)in magnesium_def,
         date(Vv1) in magnesium_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_228526) in plant,      :(_228526\==summer),
         confirmed(magnesium_def) in disorder &
r114([   material_name(none)in magnesium_def,
         method(advice)in magnesium_def,
         number(1)in magnesium_def,
         date(Vv1) in magnesium_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(_230097) in plant,      :(_230097\==summer),
         highly_confirmed(magnesium_def) in disorder &
r115([   material_name(none)in nitrogen_def,
         method(advice)in nitrogen_def,
         number(1)in nitrogen_def,
         date(Vv1) in nitrogen_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(winter) in plant,
         confirmed(nitrogen_def) in disorder &
r116([   material_name(none)in nitrogen_def,
         method(advice)in nitrogen_def,
         number(1)in nitrogen_def,
         date(Vv1) in nitrogen_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(winter) in plant,
         highly_confirmed(nitrogen_def) in disorder &
r117([   material_name(none)in potassium_def,
         method(advice)in potassium_def,
         number(1)in potassium_def,
         date(Vv1) in potassium_def]) if
         :eval_rule_exp(current_date of plant, Vv1),
         season(winter) in plant,
         confirmed(potassium_def) in disorder &
r118([   material_name(none)in potassium_def,
         method(advice)in potassium_def,
         number(1)in potassium_def,
         date(Vv1) in potassium_def]) if
```

```
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(potassium_def) in disorder &
r119([  material_name(none)in phosphorus_def,
        method(advice)in phosphorus_def,
        number(1)in phosphorus_def,
        date(Vv1) in phosphorus_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        confirmed(phosphorus_def) in disorder &
r120([  material_name(none)in phosphorus_def,
        method(advice)in phosphorus_def,
        number(1)in phosphorus_def,
        date(Vv1) in phosphorus_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(phosphorus_def) in disorder &
r121([  material_name(none)in calcium_def,
        method(advice)in calcium_def,
        number(1)in calcium_def,
        date(Vv1) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        confirmed(calcium_def) in disorder &
r122([  material_name(none)in calcium_def,
        method(advice)in calcium_def,
        number(1)in calcium_def,
        date(Vv1) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(calcium_def) in disorder &
r127([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(4) in plant,
        material_gr12(Vv1) in operation &
r128([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(4) in plant,
        material_gr12(Vv1) in operation &
r129([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(9) in plant,
        material_gr12(Vv1) in operation &
r130([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(9) in plant,
        material_gr12(Vv1) in operation &
r131([  material_name(none)in mediterranean_fruit_fly,
        method(advice)in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        date(Vv1) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(_250160) in plant,     :(_250160\==4) &
r132([  material_name(none)in mediterranean_fruit_fly,
        method(advice)in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        date(Vv1) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(_251731) in plant,     :(_251731\==4) &
r133([  material_name(none)in mediterranean_fruit_fly,
        method(advice)in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        date(Vv1) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(_253302) in plant,     :(_253302\==9) &
r134([  material_name(none)in mediterranean_fruit_fly,
        method(advice)in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        date(Vv1) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(_254873) in plant,     :(_254873\==9) &
super(rules)}.
treat_op_determine_treat_op :: {
r1([    material_qty(200)in citrus_flower_moth,
        unit('gm/100 l water')in citrus_flower_moth]) if
        material_name('super aside') in citrus_flower_moth &
r2([    material_qty(200)in citrus_flower_moth,
        unit('gm/100 l water')in citrus_flower_moth]) if
        material_name(aikaten) in citrus_flower_moth &
r3([    material_qty(300)in citrus_flower_moth,
        unit('ml/100 l water')in citrus_flower_moth]) if
        material_name('anthio 33%') in citrus_flower_moth &
r4([    material_qty(300)in citrus_flower_moth,
        unit('ml/100 l water')in citrus_flower_moth]) if
        material_name('actellic 50%') in citrus_flower_moth &
r5([    material_qty(150)in aphids,
        unit('ml/100 l water')in aphids]) if
        material_name('malathion 57%') in aphids &
r6([    material_qty(30)in citrus_white_fly,
        unit('ml/100 l water')in citrus_white_fly]) if
        material_name('vertimec 1.8%') in citrus_white_fly &
r7([    material_qty(1.5)in aphids,
        unit('L/100 l water')in aphids]) if
        material_name('super masrona 94%') in aphids &
r8([    material_qty(1.5)in citrus_white_fly,
        unit('L/100 l water')in citrus_white_fly]) if
        material_name('super masrona 94%') in citrus_white_fly &
r9([    material_qty(1.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('super masrona 94%') in scales &
r10([   material_qty(1.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('super masrona 94%') in mealy_bug &
r11([   material_qty(1.5)in aphids,
        unit('L/100 l water')in aphids]) if
        material_name('super royal 95%') in aphids &
r12([   material_qty(1.5)in citrus_white_fly,
        unit('L/100 l water')in citrus_white_fly]) if
        material_name('super royal 95%') in citrus_white_fly &
r13([   material_qty(1.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('super royal 95%') in scales &
r14([   material_qty(1.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('super royal 95%') in mealy_bug &
r15([   material_qty(1.5)in aphids,
        unit('L/100 l water')in aphids]) if
        material_name('K.Z. 95%') in aphids &
r16([   material_qty(1.5)in citrus_white_fly,
        unit('L/100 l water')in citrus_white_fly]) if
        material_name('K.Z. 95%') in citrus_white_fly &
r17([   material_qty(1.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('K.Z. 95%') in scales &
r18([   material_qty(1.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('K.Z. 95%') in mealy_bug &
r19([   unit('L/100 l water')in aphids,
        material_qty(1.6)in aphids]) if
        material_name('Kimisol 95%') in aphids &
```

```
r20([   unit('L/100 l water')in citrus_white_fly,
        material_qty(1.6)in citrus_white_fly]) if
        material_name('Kimisol 95%') in citrus_white_fly &
r21([   unit('L/100 l water')in scales,
        material_qty(1.6)in scales]) if
        material_name('Kimisol 95%') in scales &
r22([   unit('L/100 l water')in mealy_bug,
        material_qty(1.6)in mealy_bug]) if
        material_name('Kimisol 95%') in mealy_bug &
r23a([  material_qty(25)in leafminer_op1,
        unit('ml + 25 ml/100 l water')in leafminer_op1]) if
        material_name('vertimec + super masrona 94%') in leafminer_op1 &
r23b([  material_qty(25)in leafminer_op2,
        unit('ml + 25 ml/100 l water')in leafminer_op2]) if
        material_name('vertimec + super masrona 94%') in leafminer_op2 &
r23c([  material_qty(25)in leafminer_op3,
        unit('ml + 25 ml/100 l water')in leafminer_op3]) if
        material_name('vertimec + super masrona 94%') in leafminer_op3 &
r24a([  material_qty(25)in leafminer_op1,
        unit('ml + 25 ml/100 l water')in leafminer_op1]) if
        material_name('vertimec + super royal oil 95%') in leafminer_op1 &
r24b([  material_qty(25)in leafminer_op2,
        unit('ml + 25 ml/100 l water')in leafminer_op2]) if
        material_name('vertimec + super royal oil 95%') in leafminer_op2 &
r24c([  material_qty(25)in leafminer_op3,
        unit('ml + 25 ml/100 l water')in leafminer_op3]) if
        material_name('vertimec + super royal oil 95%') in leafminer_op3 &
r25a([  material_qty(25)in leafminer_op1,
        unit('ml + 25 ml/100 l water')in leafminer_op1]) if
        material_name('vertimec + K.Z oil 95%') in leafminer_op1 &
r25b([  material_qty(25)in leafminer_op2,
        unit('ml + 25 ml/100 l water')in leafminer_op2]) if
        material_name('vertimec + K.Z oil 95%') in leafminer_op2 &
r25c([  material_qty(25)in leafminer_op3,
        unit('ml + 25 ml/100 l water')in leafminer_op3]) if
        material_name('vertimec + K.Z oil 95%') in leafminer_op3 &
r26a([  material_qty(25)in leafminer_op1,
        unit('ml + 25 ml/100 l water')in leafminer_op1]) if
        material_name('vertimec + Kimisol oil 95%') in leafminer_op1 &
r26b([  material_qty(25)in leafminer_op2,
        unit('ml + 25 ml/100 l water')in leafminer_op2]) if
        material_name('vertimec + Kimisol oil 95%') in leafminer_op2 &
r26c([  material_qty(25)in leafminer_op3,
        unit('ml + 25 ml/100 l water')in leafminer_op3]) if
        material_name('vertimec + Kimisol oil 95%') in leafminer_op3 &
r27([   material_qty(10)in gummosis,
        unit('gm/1 l water')in gummosis]) if
        material_name(potassium_permenganat) in gummosis &
r28([   material_qty(1)in gummosis,
        unit('kg CuSo4 + 2 Kg CaO + 10 L water')in gummosis]) if
        material_name('bordeaux past') in gummosis &
r29a([  material_qty(20)in wilt_root_rot_op1,
        unit('gm/tree')in wilt_root_rot_op1]) if
        material_name(topsin) in wilt_root_rot_op1 &
r29b([  material_qty(20)in wilt_root_rot_op2,
        unit('gm/tree')in wilt_root_rot_op2]) if
        material_name(topsin) in wilt_root_rot_op2 &
r30a([  material_qty(150)in ganoderma_rot_op1,
        unit('gm/100 l water')in ganoderma_rot_op1]) if
        material_name(topsin) in ganoderma_rot_op1 &
r30b([  material_qty(150)in ganoderma_rot_op2,
        unit('gm/100 l water')in ganoderma_rot_op2]) if
        material_name(topsin) in ganoderma_rot_op1 &
r31([   material_qty(500)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name(copper_oxychloride) in lichens &
r32([   material_qty(250)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name('cuprus K.Z 50%') in lichens &
r33([   material_qty(500)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name('pory coper 50%') in lichens &
r34([   material_qty(500)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name('pro coper 50%') in lichens &
r35([   material_qty(500)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name('copox 50%') in lichens &
r36([   material_qty(1)in lichens,
        unit('Kg Cu So4 + 1.5 CaO/100 l water')in lichens]) if
        material_name('caprimex 98%') in lichens &
r37([   material_qty(350)in lichens,
        unit('gm/100 l water')in lichens]) if
        material_name('halomac 65%') in lichens &
r38a([  material_qty(50)in flat_mite_op1,
        unit('ml + 150 ml/100 l water')in flat_mite_op1]) if
        material_name('ortis 5% sc + kz oil') in flat_mite_op1 &
r38b([  material_qty(50)in flat_mite_op2,
        unit('ml + 150 ml/100 l water')in flat_mite_op2]) if
        material_name('ortis 5% sc + kz oil') in flat_mite_op2 &
r39a([  material_qty(50)in brown_mite_op1,
        unit('ml + 150 ml/100 l water')in brown_mite_op1]) if
        material_name('ortis 5% sc + kz oil') in brown_mite_op1 &
r39b([  material_qty(50)in brown_mite_op2,
        unit('ml + 150 ml/100 l water')in brown_mite_op2]) if
        material_name('ortis 5% sc + kz oil') in brown_mite_op2 &
r40a([  material_qty(100)in rust_mite_op1,
        unit('ml + 150 ml/100 l water')in rust_mite_op1]) if
        material_name('ortis 5% sc + kz oil') in rust_mite_op1 &
r40b([  material_qty(100)in rust_mite_op2,
        unit('ml + 150 ml/100 l water')in rust_mite_op2]) if
        material_name('ortis 5% sc + kz oil') in rust_mite_op2 &
r41a([  material_qty(100)in bud_mite_op1,
        unit('ml + 150 ml/100 l water')in bud_mite_op1]) if
        material_name('ortis 5% sc + kz oil') in bud_mite_op1 &
r41b([  material_qty(100)in bud_mite_op2,
        unit('ml + 150 ml/100 l water')in bud_mite_op2]) if
        material_name('ortis 5% sc + kz oil') in bud_mite_op2 &
r42a([  material_qty(40)in rust_mite_op1,
        unit('ml/100 l water')in rust_mite_op1]) if
        material_name('neron 50%') in rust_mite_op1 &
r42b([  material_qty(40)in rust_mite_op2,
        unit('ml/100 l water')in rust_mite_op2]) if
        material_name('neron 50%') in rust_mite_op2 &
r43a([  material_qty(40)in bud_mite_op1,
        unit('ml/100 l water')in bud_mite_op1]) if
        material_name('neron 50%') in bud_mite_op1 &
r43b([  material_qty(40)in bud_mite_op2,
        unit('ml/100 l water')in bud_mite_op2]) if
        material_name('neron 50%') in bud_mite_op2 &
r44a([  material_qty(30)in rust_mite_op1,
        unit('ml + 250 ml/100 L water')in rust_mite_op1]) if
        material_name('vertimec 1.8% + kz oil') in rust_mite_op1 &
r44b([  material_qty(30)in rust_mite_op2,
        unit('ml + 250 ml/100 L water')in rust_mite_op2]) if
        material_name('vertimec 1.8% + kz oil') in rust_mite_op2 &
r45a([  material_qty(30)in bud_mite_op1,
        unit('ml + 250 ml/100 L water')in bud_mite_op1]) if
        material_name('vertimec 1.8% + kz oil') in bud_mite_op1 &
r45b([  material_qty(30)in bud_mite_op2,
        unit('ml + 250 ml/100 L water')in bud_mite_op2]) if
        material_name('vertimec 1.8% + kz oil') in bud_mite_op2 &
r46a([  material_qty(30)in flat_mite_op1,
        unit('ml + 250 ml/100 L water')in flat_mite_op1]) if
        material_name('vertimec 1.8% + kz oil') in flat_mite_op1 &
r46b([  material_qty(30)in flat_mite_op2,
        unit('ml + 250 ml/100 L water')in flat_mite_op2]) if
        material_name('vertimec 1.8% + kz oil') in flat_mite_op2 &
r47a([  material_qty(30)in brown_mite_op1,
        unit('ml + 250 ml/100 L water')in brown_mite_op1]) if
        material_name('vertimec 1.8% + kz oil') in brown_mite_op1 &
r47b([  material_qty(30)in brown_mite_op2,
        unit('ml + 250 ml/100 L water')in brown_mite_op2]) if
        material_name('vertimec 1.8% + kz oil') in brown_mite_op2 &
r48a([  material_qty(100)in flat_mite_op1,
        unit('ml/100 l water')in flat_mite_op1]) if
        material_name(pride) in flat_mite_op1 &
r48b([  material_qty(100)in flat_mite_op2,
        unit('ml/100 l water')in flat_mite_op2]) if
        material_name(pride) in flat_mite_op2 &
```

```
r49a([  material_qty(100)in brown_mite_op1,
        unit('ml/100 l water')in brown_mite_op1]) if
        material_name(pride) in brown_mite_op1 &
r49b([  material_qty(100)in brown_mite_op2,
        unit('ml/100 l water')in brown_mite_op2]) if
        material_name(pride) in brown_mite_op2 &
r50a([  material_qty(17)in citrus_nematude_op1,
        unit('kg/feddan')in citrus_nematude_op1]) if
        material_name('temic 15%') in citrus_nematude_op1 &
r50b([  material_qty(17)in citrus_nematude_op2,
        unit('kg/feddan')in citrus_nematude_op2]) if
        material_name('temic 15%') in citrus_nematude_op2 &
r51a([  material_qty(40)in citrus_nematude_op1,
        unit('kg/feddan')in citrus_nematude_op1]) if
        material_name('furidan 10%') in citrus_nematude_op1 &
r51b([  material_qty(40)in citrus_nematude_op2,
        unit('kg/feddan')in citrus_nematude_op2]) if
        material_name('furidan 10%') in citrus_nematude_op2 &
r52a([  material_qty(24)in citrus_nematude_op1,
        unit('kg/feddan')in citrus_nematude_op1]) if
        material_name('ragbi 10%') in citrus_nematude_op1 &
r52b([  material_qty(24)in citrus_nematude_op2,
        unit('kg/feddan')in citrus_nematude_op2]) if
        material_name('ragbi 10%') in citrus_nematude_op2 &
r53a([  material_qty(4)in citrus_nematude_op1,
        unit('L/feddan')in citrus_nematude_op1]) if
        material_name(vaydete) in citrus_nematude_op1 &
r53b([  material_qty(4)in citrus_nematude_op2,
        unit('L/feddan')in citrus_nematude_op2]) if
        material_name(vaydete) in citrus_nematude_op2 &
r54([   material_qty(2.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('bolum oil 80%') in mealy_bug &
r55([   material_qty(2.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('royal oil 80%') in mealy_bug &
r56([   material_qty(2.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('misrona oil 80%') in mealy_bug &
r57([   material_qty(2.5)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('agro oil 80%') in mealy_bug &
r58([   material_qty(2.0)in mealy_bug,
        unit('L/100 l water')in mealy_bug]) if
        material_name('focal oil 82%') in mealy_bug &
r59([   material_qty(2.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('bolum oil 80%') in scales &
r60([   material_qty(2.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('royal oil 80%') in scales &
r61([   material_qty(2.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('misrona oil 80%') in scales &
r62([   material_qty(2.5)in scales,
        unit('L/100 l water')in scales]) if
        material_name('agro oil 80%') in scales &
r63([   material_qty(2.0)in scales,
        unit('L/100 l water')in scales]) if
        material_name('focal oil 82%') in scales &
r64([   material_qty(100)in mediterranean_fruit_fly,
        unit('ml + 500 ml/100 l water')in mediterranean_fruit_fly]) if
        material_name('malthion 57% + policure') in mediterranean_fruit_fly &
r65([   material_qty(500)in mediterranean_fruit_fly,
        unit('ml + L/100 l water')in mediterranean_fruit_fly]) if
        material_name('libacid 50% + bominal') in mediterranean_fruit_fly &
r66([   material_qty(2)in calcium_def,
        unit('kg/100 l water')in calcium_def]) if
        material_name('calcium nitrate') in calcium_def &
r67([   material_qty(0.5)in calcium_def,
        unit('kg/100 l water')in calcium_def]) if
        material_name('calcium chloride') in calcium_def &
r68([   material_qty(1.5)in potassium_def,
        unit('kg/100 l water')in potassium_def]) if
        material_name(potassium_nitrate) in potassium_def &
r69([   material_qty(2)in potassium_def,
        unit('kg/100 l water')in potassium_def]) if
        material_name(potassium_sulfate) in potassium_def &
r70([   material_qty(1)in phosphorus_def,
        unit('kg/100 l water')in phosphorus_def]) if
        material_name('triple phosphate') in phosphorus_def &
r71([   material_qty(1)in nitrogen_def,
        unit('kg/100 l water')in nitrogen_def]) if
        material_name(urea) in nitrogen_def &
r72([   material_qty(1.5)in nitrogen_def,
        unit('kg/100 l water')in nitrogen_def]) if
        material_name('ammonium nitrate') in nitrogen_def &
r73([   material_qty(0.5)in magnesium_def,
        unit('kg/100 l water')in magnesium_def]) if
        material_name(magnesium_sulfate) in magnesium_def &
r74([   material_qty(0)in treat_op,
        unit('as below')in treat_op]) if
        material_name('micro element mixture') in treat_op &
super(rules)
}.
enhanced_by :: {
r1([    advice('Avoid excess of nitrogen fertilizers and organic manure near the trunk. Also,
avoid excess irrigation water near the trunk.')in gummosis]) if
        method(painting) in gummosis,
        season(winter) in plant &
```

```
r2a([	advice('Application of acaricides is recommended at 20 % infestation, in general.
Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application. Spraying should be as a mist, tacking
umbrella shape at lower pressure and as possible over the entire tree')in rust_mite_op1]) if
	method('chemical spray') in rust_mite_op1,
	season(summer) in plant &
r2b([	advice('Application of acaricides is recommended at 20 % infestation, in general.
Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application. Spraying should be as a mist, tacking
umbrella shape at lower pressure and as possible over the entire tree')in rust_mite_op2]) if
	method('chemical spray') in rust_mite_op2,
	season(summer) in plant &
r3a([	advice('The treatment at this time is not recommended. Time of chemical control in
late of April, in case of infestation')in rust_mite_op1]) if
	method('chemical spray') in rust_mite_op1,
	season(_60191) in plant,	:(_60191\==summer) &
r3b([	advice('The treatment at this time is not recommended. Time of chemical control in
late of April, in case of infestation')in rust_mite_op2]) if
	method('chemical spray') in rust_mite_op2,
	season(_60191) in plant,	:(_60191\==summer) &
r4a1([	advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in bud_mite_op1,
	advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible')in bud_mite_op1]) if
	method('chemical spray') in bud_mite_op1,
	current_week(_61304) in plant,	:(_61304>=7),
	current_week(_61567) in plant,	:(_61567=<22) &
r4a2([	advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in bud_mite_op2,
	advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible')in bud_mite_op2]) if
	method('chemical spray') in bud_mite_op2,
	current_week(_61304) in plant,	:(_61304>=7),
	current_week(_61567) in plant,	:(_61567=<22) &
r4b1([	advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in bud_mite_op1,
	advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible')in bud_mite_op1]) if
	method('chemical spray') in bud_mite_op1,
	current_week(_62680) in plant,	:(_62680>=35),
	current_week(_62943) in plant,	:(_62943=<44) &
r4b2([	advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in bud_mite_op2,
	advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible')in bud_mite_op2]) if
	method('chemical spray') in bud_mite_op2,
	current_week(_62680) in plant,	:(_62680>=35),
	current_week(_62943) in plant,	:(_62943=<44) &
r5a1([	advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op1]) if
```

```
        method('chemical spray') in bud_mite_op1,
        current_week(_63926) in plant,      :(_63926>0),
        current_week(_64189) in plant,      :(_64189<7) &
r5a2([  advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_63926) in plant,      :(_63926>0),
        current_week(_64189) in plant,      :(_64189<7) &
r5b1([  advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_65172) in plant,      :(_65172>22),
        current_week(_65435) in plant,      :(_65435<35) &
r5b2([  advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_65172) in plant,      :(_65172>22),
        current_week(_65435) in plant,      :(_65435<35) &
r5c1([  advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_66418) in plant,      :(_66418>44),
        current_week(_66681) in plant,      :(_66681<54) &
r5c2([  advice('The treatment at this time is not recommended. Time of chemical control in
late of February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_66418) in plant,      :(_66418>44),
        current_week(_66681) in plant,      :(_66681<54) &
r6a([   advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in brown_mite_op1,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from upwards to downwards')in brown_mite_op1]) if
        method('chemical spray') in brown_mite_op1,
        season(summer) in plant &
r6b([   advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in brown_mite_op2,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from upwards to downwards')in brown_mite_op2]) if
        method('chemical spray') in brown_mite_op2,
        season(summer) in plant &
r7a([   advice('The treatment at this time is not recommended. Time of chemical control in
late of May, in case of infestation')in brown_mite_op1]) if
        method('chemical spray') in brown_mite_op1,
        season(_68618) in plant,     :(_68618\==summer) &
r7b([   advice('The treatment at this time is not recommended. Time of chemical control in
late of May, in case of infestation')in brown_mite_op2]) if
        method('chemical spray') in brown_mite_op2,
        season(_68618) in plant,     :(_68618\==summer) &
r8a([   advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in flat_mite_op1,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from downwards to up words and pointing to the core of tree')in
flat_mite_op1]) if
        method('chemical spray') in flat_mite_op1,
        season(summer) in plant &
r8b([   advice('Spot spraying localized infestation is good practice and tractor drawn
equipment with agitator is often the ideal machine for application')in flat_mite_op2,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as
far as possible from downwards to up words and pointing to the core of tree')in
flat_mite_op2]) if
        method('chemical spray') in flat_mite_op2,
        season(summer) in plant &
r9a([   advice('The treatment at this time is not recommended. Time of chemical control in
late of April, in case of infestation')in flat_mite_op1]) if
        method('chemical spray') in flat_mite_op1,
        season(_70555) in plant,       :(_70555\==summer) &
r9b([   advice('The treatment at this time is not recommended. Time of chemical control in
late of April, in case of infestation')in flat_mite_op2]) if
        method('chemical spray') in flat_mite_op2,
        season(_70555) in plant,       :(_70555\==summer) &
r10([   advice('It is important to check the soil salinity, and in case of high salinity the
leaching is recommended')in magnesium_def]) if
        method('foliage nutrition') in magnesium_def,
        season(summer) in plant &
r11([   advice('The micro elements mixture is formulated, for every 100 lt water, as follow :
30 gm Iron Chelate (EDTA) + 30 gm Zinc Chelate + 75 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax')in manganese_def]) if
        method('foliage nutrition') in manganese_def,
        season(summer) in plant &
r12([   advice('The micro elements mixture is formulated, for every 100 lt water , as follow :
150 gm Iron Chelate (EDTA) + 30 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax')in iron_def]) if
        method('foliage nutrition') in iron_def,
        season(summer) in plant &
r13([   advice('The micro elements mixture is formulated, for every 100 lt water, as follow:
30 gm Iron Chelate (EDTA) + 150 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper
Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax')in zinc_def]) if
        method('foliage nutrition') in zinc_def,
        season(summer) in plant &
r14([   advice('No foliage application during the flowering stage and fruit setting')in
zinc_def]) if
        method(advice) in zinc_def,
        season(spring) in plant &
r15([   advice('No foliage application during the flowering stage and fruit setting')in
iron_def]) if
        method(advice) in iron_def,
        season(spring) in plant &
r16([   advice('No foliage application during the flowering stage and fruit setting')in
manganese_def]) if
        method(advice) in manganese_def,
        season(spring) in plant &
r17([   advice('No foliage application during the flowering stage and fruit setting')in
magnesium_def]) if
        method(advice) in magnesium_def,
        season(spring) in plant &
r18([   advice('No foliage application during the fruits collecting period')in zinc_def]) if
        method(advice) in zinc_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r19([   advice('No foliage application during the fruits collecting period')in iron_def]) if
        method(advice) in iron_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r20([   advice('No foliage application during the fruits collecting period')in manganese_def])
if
        method(advice) in manganese_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r21([   advice('No foliage application during the fruits collecting period')in magnesium_def])
if
        method(advice) in magnesium_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r22([   advice('Substitute the nitrogen quantity in the fertilization expert system
recommendation by its equivalence of calcium nitrate')in calcium_def]) if
        method('foliage nutrition') in calcium_def,
        season(_82672) in plant,        :(_82672\==winter) &
r23([   advice('No significant response of trees to foliar application during winter. Therefore
treat your trees in the beginning of spring')in nitrogen_def]) if
        method(advice) in nitrogen_def,
        season(winter) in plant &
r24([   advice('No significant response of trees to foliar application during winter. Therefore
treat your trees in the beginning of spring')in potassium_def]) if
        method(advice) in potassium_def,
        season(winter) in plant &
r25([   advice('No significant response of trees to foliar application during winter. Therefore
treat your trees in the beginning of spring')in phosphorus_def]) if
        method(advice) in phosphorus_def,
        season(winter) in plant &
r26([   advice('No significant response of trees to foliar application during winter. Therefore
treat your trees in the beginning of spring')in calcium_def]) if
        method(advice) in calcium_def,
        season(winter) in plant &
r27([   advice('Spraying two branches only in each tree and collects infested fruits and bury
it.')in mediterranean_fruit_fly]) if
        method('chemical spray') in mediterranean_fruit_fly,
        current_month(4) in plant &
r28([   advice('Spraying two branches only in each tree and collects infested fruits and bury
it.')in mediterranean_fruit_fly]) if
        method('chemical spray') in mediterranean_fruit_fly,
        current_month(9) in plant &
r29a([  advice('Collect infested fruits and bury it.')in mediterranean_fruit_fly,
        advice('The treatment at this time is not recommended.')in mediterranean_fruit_fly])
if
        method(advice) in mediterranean_fruit_fly,
        current_month(_88859) in plant,      :(_88859\==4) &
r29b([  advice('Collect infested fruits and bury it.')in mediterranean_fruit_fly,
        advice('The treatment at this time is not recommended.')in mediterranean_fruit_fly])
if
        method(advice) in mediterranean_fruit_fly,
        current_month(_89946) in plant,      :(_89946\==9) &
r30([   advice('Increase quantity of fertilizer application by 50% and incrementally increased
up to 100% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in nitrogen_def]) if
        method('foliage nutrition') in nitrogen_def,
        nitrogen_infestation('very low') in disorder,
        season(_91085) in plant,       :(_91085\==winter) &
r31([   advice('Increase quantity of fertilizer application by 25% and incrementally increased
up to 50% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in nitrogen_def]) if
        method('foliage nutrition') in nitrogen_def,
        nitrogen_infestation(low) in disorder,
        season(_92224) in plant,       :(_92224\==winter) &
r32([   advice('Increase quantity of fertilizer application by 50% and incrementally increased
up to 100% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in potassium_def]) if
        method('foliage nutrition') in potassium_def,
        nitrogen_infestation('very low') in disorder,
        season(_93363) in plant,       :(_93363\==winter) &
r33([   advice('Increase quantity of fertilizer application by 25% and incrementally increased
up to 50% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in potassium_def]) if
        method('foliage nutrition') in potassium_def,
        nitrogen_infestation(low) in disorder,
        season(_94502) in plant,       :(_94502\==winter) &
r34([   advice('Increase quantity of fertilizer application by 50% and incrementally increased
up to 100% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in phosphorus_def]) if
        method('foliage nutrition') in phosphorus_def,
        nitrogen_infestation('very low') in disorder,
        season(_94502) in plant,       :(_94502\==winter) &
r35([   advice('Increase quantity of fertilizer application by 25% and incrementally increased
up to 50% or until disappearance of nutrient deficiency observations, then apply the
recommendations given by the fertilization expert system')in phosphorus_def]) if
        method('foliage nutrition') in phosphorus_def,
        nitrogen_infestation(low) in disorder,
        season(_96673) in plant,        :(_96673\==winter) &
r36([   advice('Good caring the diseased trees; i.e. better agriculture practices and
fertilization to extend the productive life of tree when yield becomes not economic, the
diseased trees must be replaced. Use certified transplants')in psorosis]) if
        method(advice) in psorosis &
r37([   advice('Infected young trees should be replaced by other healthy plants. Use certified
transplants')in stubborn]) if
        method(advice) in stubborn &
r38([   advice('Infected young trees should be replaced by other healthy plants. Use certified
transplants')in impieetratura]) if
        method(advice) in impieetratura &
r39([   advice('Improve the agriculture practices')in anthracnose]) if
        method(advice) in anthracnose &
r40([   advice('Improve the agriculture practices')in alternaria_leaves_spot]) if
        method(advice) in alternaria_leaves_spot &
r41([   advice('Control the insects that produce the honey dew')in sooty_mold]) if
        method(advice) in sooty_mold &
r42([   advice('Collect infected fruits and bury it. Perform the suitable agriculture
practices')in alternaria_rot]) if
        method(advice) in alternaria_rot &
r43([   advice('The diseased trees must be replaced')in armillaria_root_rot]) if
        method(advice) in armillaria_root_rot &
r44([   advice('No treatment for this phenomena where its economic importance is limited')in
gum_spots]) if
        method(advice) in gum_spots &
r45([   advice('Improve the growth of trees to protect the fruits from direct sun light')in
sun_burn]) if
        method(advice) in sun_burn &
r46([   advice('Manage the irrigation and increase the fertilization quantity of Potassium')in
fruit_cracking]) if
        method(advice) in fruit_cracking &
r47([   advice('Manage the irrigation and increase the fertilization quantity of Potassium')in
fruit_creasing]) if
        method(advice) in fruit_creasing &
r48([   advice('Cultivate plant tarps for scarab like faba-beans, turnip and cauliflower')in
rose_scarab,
        advice('Picking up the insects twice a day')in rose_scarab,
        advice('Spread watercolor traps at the rate 35 to 40 traps per feddan')in rose_scarab,
        advice('Use compost organic manure')in rose_scarab]) if
        method(advice) in rose_scarab &
r49([   advice('Use irrigation program to add leaching requirements')in salt_injury]) if
        method(advice) in salt_injury &
r50([   advice('No treatment for this pest, such that it is not important economically')in
green_stink_bug]) if
        method(advice) in green_stink_bug &
r51([   advice('The gum pocked must be removed with sharp knife, the wound and exposed
tissues must be disinfected with solution')in gummosis]) if
        method(disinfection) in gummosis &
```

```
r52a([  advice('Remove fungal growths and painting the wound by Bordeaux past then spray
the green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water')in
ganoderma_rot_op1]) if
        method('chemical spray') in ganoderma_rot_op1 &
r52b([  advice('Remove fungal growths and painting the wound by Bordeaux past then spray
the green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water')in
ganoderma_rot_op2]) if
        method('chemical spray') in ganoderma_rot_op2 &
r53([   advice('Spray the infested trees only')in aphids,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch
without direct application')in aphids]) if
        method('chemical spray') in aphids,
        :( \+ method('chemical spray') in citrus_white_fly) &
r54([   advice('Spray the infested trees only')in citrus_white_fly,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch
without direct application')in citrus_white_fly]) if
        :( \+ method('chemical spray') in aphids),
        method('chemical spray') in citrus_white_fly &
r55([   advice('Spray the infested trees only')in aphids,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch
without direct application')in aphids,
        advice('This operation used as shared treatment for aphids and citrus white fly')in
aphids]) if
        method('chemical spray') in aphids,
        method('chemical spray') in citrus_white_fly &
r56([   advice('Spray the infested trees only')in citrus_white_fly,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch
without direct application')in citrus_white_fly,
        advice('This operation used as shared treatment for aphids and citrus white fly')in
citrus_white_fly]) if
        method('chemical spray') in aphids,
        method('chemical spray') in citrus_white_fly &
r57([   advice('Spray trees of entire farm')in citrus_flower_moth,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch
without direct application')in citrus_flower_moth]) if
        method('chemical spray') in citrus_flower_moth &
r58([   advice('Lichens control includes good agricultural practices; i.e. pruning and avoid
excess irrigation water')in lichens]) if
        method('chemical spray') in lichens &
r59([   advice('Use fit spraying motor with good mixing. The trees must be completely
washed.')in scales]) if
        method('chemical spray') in scales &
r60([   advice('Use fit spraying motor with good mixing. The trees must be completely
washed.')in mealy_bug]) if
        method('chemical spray') in mealy_bug &
r61a([  advice('You must follow this operation by light irrigation to avoid application of fruit
bearing trees')in citrus_nematude_op1]) if
        method('soil treatment') in citrus_nematude_op1 &
r61b([  advice('You must follow this operation by light irrigation to avoid application of fruit
bearing trees')in citrus_nematude_op2]) if
```

```
          method('soil treatment') in citrus_nematude_op2 &
super(rules)
}.
```

6.2. Inference layer

File name : treat_inference.pl

```
    :- ensure_loaded('$KROL/lib/krol_init').
    treat_inference :: {
        instantiate :-
        treated_by :: conclude_all &
        assign :-
        treat_op_determine_treat_op :: conclude_all,
        enhanced_by :: conclude_all &
        order :-
                :orderM&
    super(krol_init)
    }.
```

File name : order.pl

```
    date_sort :: {
        '<'(I, J) :-
                I = [_,_,_,_,X,_,_,_,_],
                J = [_,_,_,_,Y,_,_,_,_],
                X = [], Y = [_,_,_], ! &
        '<'([_,_,_,_,X,_,_,_,_], [_,_,_,_,Y,_,_,_,_]) :-
                :atom(X), Y = [_,_,_], ! &
        '<'([_,_,_,_,X,_,_,_,_], [_,_,_,_,Y,_,_,_,_]) :-
                :compare_date(<, X, Y)}.
    treated_before :: {
        '<'(X, Y) :-
                insects :: descendant(X),
                (       nematode :: descendant(Y)
                ;       nutrition_def :: descendant(Y)
                ;       Y = lichens
                ), !
    }.
    sort(Type) :: {
         :- :use_module(library(lists), [append/3]) &
         qsort([], []) &
         qsort([P|L], S) :-
              partition(L, P, Small, Large),
              qsort(Small, S0),
              qsort(Large, S1),
              :append(S0, [P|S1], S) &
         partition([], _P, [], []) &
         partition([X|L1], P, Small, Large) :-
              (  Type :: '<'(X, P) ->
                 Small = [X|Small1], Large = Large1
              ;  Small = Small1, Large = [X|Large1]
```

```
             ),
             partition(L1, P, Small1, Large1)}.
orderM :-
    findall(O,
        (       treat_op :: leaf(O),
                O :: get(number(X)),
                X \== []
        ), List),
    List = [_,_|_], !,
    sort(treated_before) :: qsort(List, List1),
    ordernumber(List1,1),
    satisfy_3days(List1).
orderM.
ordernumber([],_).
ordernumber([O|ListT],N):-
    O :: set(number(N)),
    N1 is N + 1,
    ordernumber(ListT,N1).
satisfy_3days([]).
satisfy_3days([_]) :- !.
satisfy_3days([gummosis_op1, gummosis_op2|List]) :- !,
    gummosis_op1 :: get(date(Date1)),
    gummosis_op2 :: get(date(Date1)),
    satisfy_3days([gummosis_op2|List]).
satisfy_3days([O1, O2|List]) :-
    O1 :: get(material_name([none|_]))->
    satisfy_3days([O2|List]);
(
    O1 :: get(date(Date1)),
    (       Date1 = [] ->
            satisfy_3days([O2|List])
    ;       O2 :: get(date(Date2)),
            (       Date2 = [] ->
                    satisfy_3days([O1|List])
            ;       diference(Date2, Date1, _, Days),
                    (       Date1 = Date2 ->
                            new_date(Date2,[3,0,0], Date2x),
                            O2 ::  set_value(date(Date2x)),
                            add_three_days(List, 3)
                    ;       compare_date(<, Date2, Date1) ->
                            Ds is Days + 3,
                            new_date(Date2,[Ds,0,0], Date2x),
                            O2 ::  set_value(date(Date2x)),
                            add_three_days(List, Ds)
                    ;       Days < 3 ->
                            Ds is 3 - Days,
                            new_date(Date2,[3,0,0], Date2x),
                            O2 ::  set_value(date(Date2x)),
                            add_three_days(List, 3)
                    ;       true
```

```
                          ),
                          satisfy_3days([O2|List])
                    )
          )
     ).
     add_three_days([], _).
     add_three_days([O|Os], Ds) :-
          O :: get(date(Date2)),
          (       Date2 = [] ->
                  true
          ;       new_date(Date2,[Ds,0,0], Date2x),
                  O ::  set_value(date(Date2x))
          ),
          add_three_days(Os, Ds).
```

6.3. Task layer

File name : treat_task.pl

```
treat_task :: {
super(krol_init)}.
treat_task_transfer :: {
super(treat_task)}.
treat_task_uncondional :: {
start_inference :-
                    treat_inference :: instantiate,
                    krol_init :: set(mode(un)),
                    treat_inference :: assign,
                    treat_inference :: order,
                    disorder :: get(confirmed(L1)),
                    disorder :: get(highly_confirmed(L2)),
                    :append(L1,L2,Dis),
                    :get_treat(Dis)&
super(treat_task)}.
treat_task_condional :: {
super(treat_task)}.
treat_task_repetitive :: {
super(treat_task)}.
treat_task_user :: {
super(treat_task)}.
```

6.4. User Interface

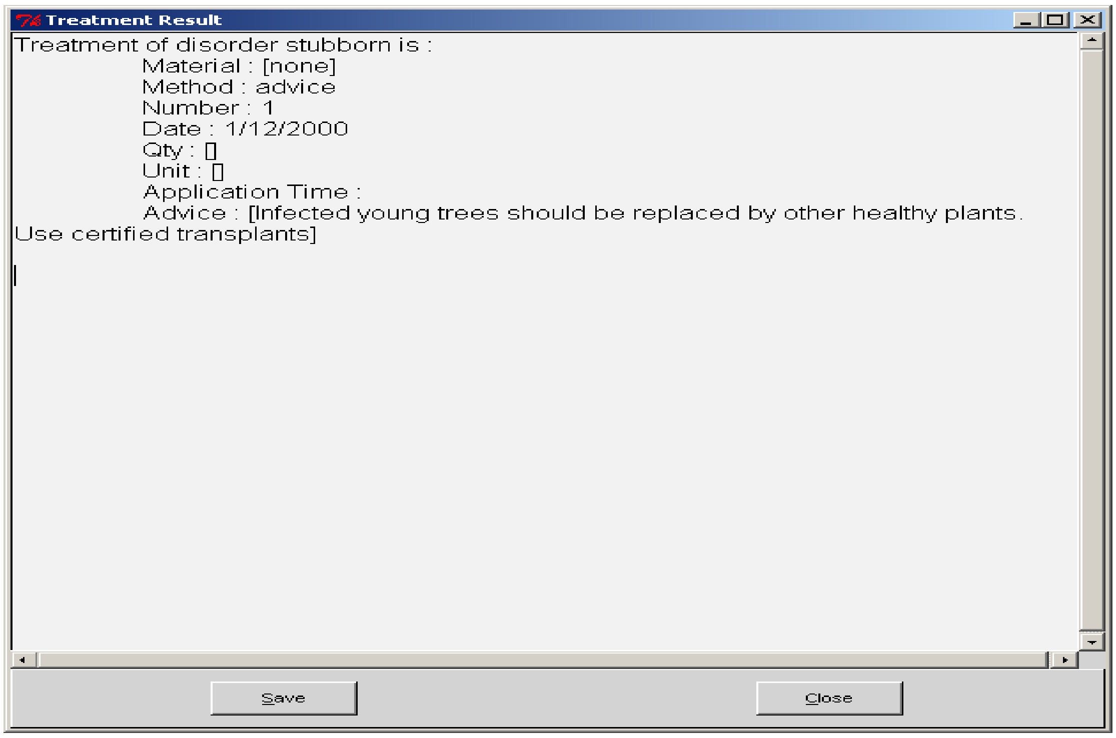
File name : treat_dlg.pl

```
:- ensure_loaded('$KROL/lib/flatten').
:- ensure_loaded('$KROL/lib/txtw').
:- ensure_loaded('$KROL/lib/buttonbox').
:- use_module(library(lists), [prefix/2]).
treat_dialog :: {
belong_to(citex_diag_dlg) &
window_title('Treatment Result') &
```

```
widget(treat_dialog, []) &
components([
            treat_txt,
            treat_txt_buttons]) &
handle_abnormal_exit :-
            treat_txt_buttons :: action(end) &
super(dialog)}.
treat_txt :: {
belong_to(treat_dialog) &
widget(treat_txt, ['-height', 480, '-width', 640], ['text.font 8x13']) &
pack(['-expand true -fill both']) &
super(textwindow)}.
treat_txt_buttons :: {
belong_to(treat_dialog) &
widget(treat_txt_buttons, Args, Options) :-
            Args = ['-orient horizontal'],
            Options = [] &
pack(['-fill x']) &
button(save, Args, Bind) :-
            Args = ['-text', 'Save', '-command', 'treat_txt_buttons :: action(save)',
                  '-underline 0', '-width 10'],
            Bind = '<Control-s>' &
button(close, Args, Bind) :-
            Args = ['-text', 'Close', '-command', 'treat_txt_buttons :: action(close)',
                  '-underline 0', '-width 10'],
            Bind = '<Control-e>' &
default(close) &
action(close) :-
            treat_dialog :: destroy &
action(save) :-
            tcl :: get_save_file('', File, 'Save Treatment Result File'),
            (    File = '' ->
                 :true
            ;    treat_txt :: fetch(T),
                 :open(File, write, Stream),
                 :format(Stream,'~w', [T]),
                 :close(Stream)
            ) &

super(buttonbox)
}.
show_treat([]).
show_treat([X|Xs]) :-
            show_treat1(X),
            show_treat(Xs).
show_treat1([]).
show_treat1([X|Xs]) :-
            (    (X = [D,M,Y], valid_date(M, D, Y)) ->
                 formate_date(X1, X),
                 format_to_chars("      Date : ~s~n",[X1], X11),
                 name(X2, X11)
            ;    name(X, Y),
                 (         prefix("next ", Y) ->
                           format_to_chars("      Date : ~s~n",[Y], X11),
                           name(X2, X11)
                 ;         X2 = X
                 )
            ),
            treat_txt :: insert(X2),
            show_treat1(Xs).
```

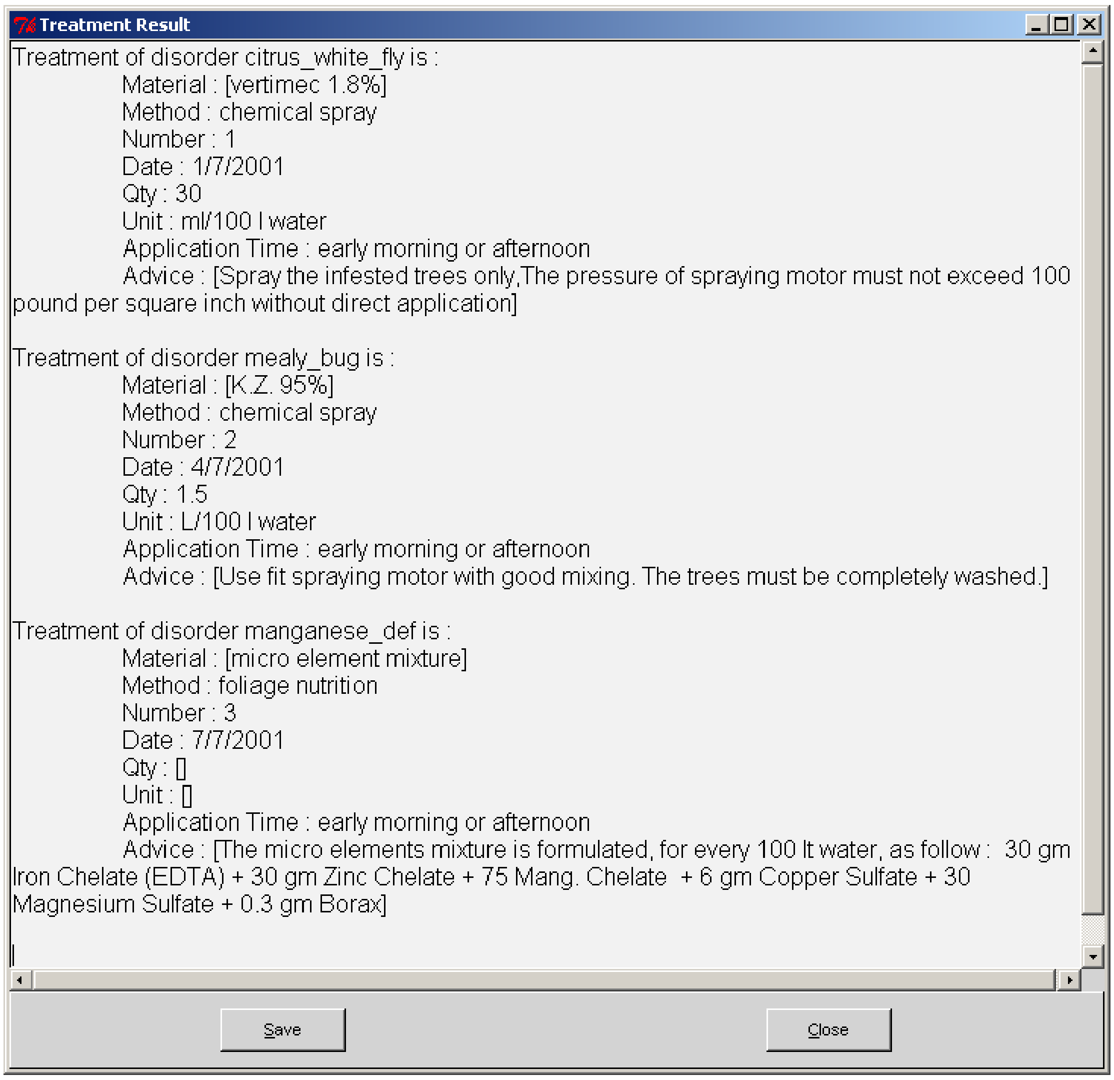
6.5. Treatment Test Case

Case 1



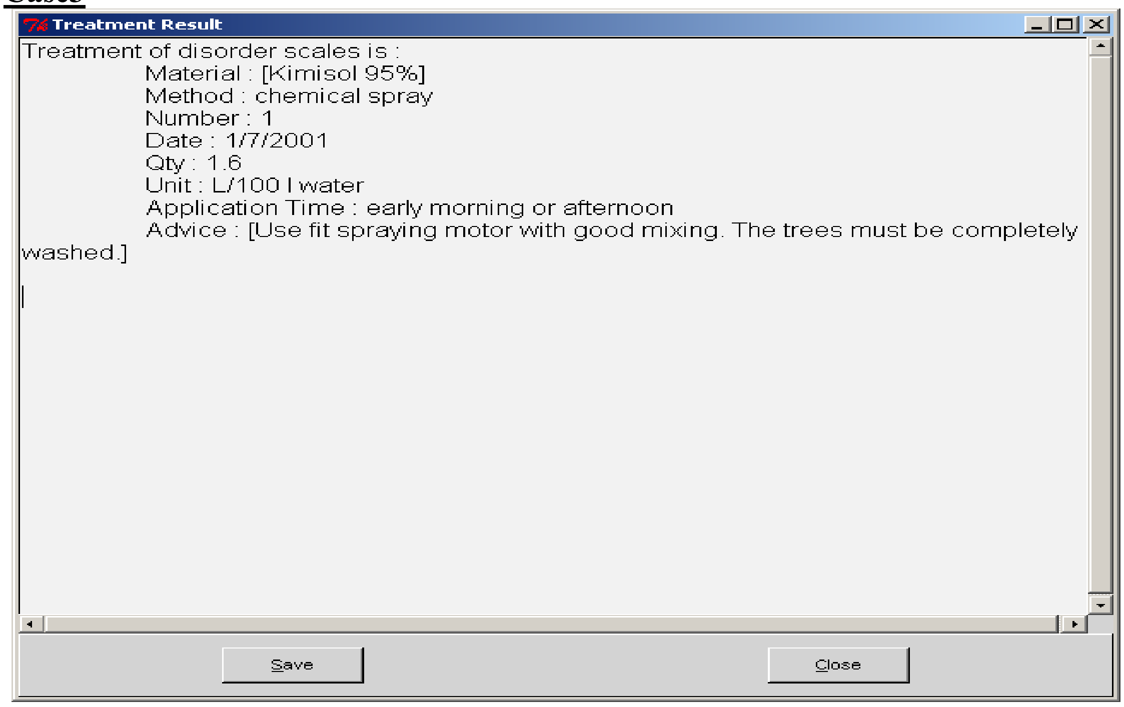
```
Treatment of disorder stubborn is :
                Material : [none]
                Method : advice
                Number : 1
                Date : 1/12/2000
                Qty : []
                Unit : []
                Application Time :
                Advice : [Infected young trees should be replaced by other healthy plants.
Use certified transplants]
```

Case 2



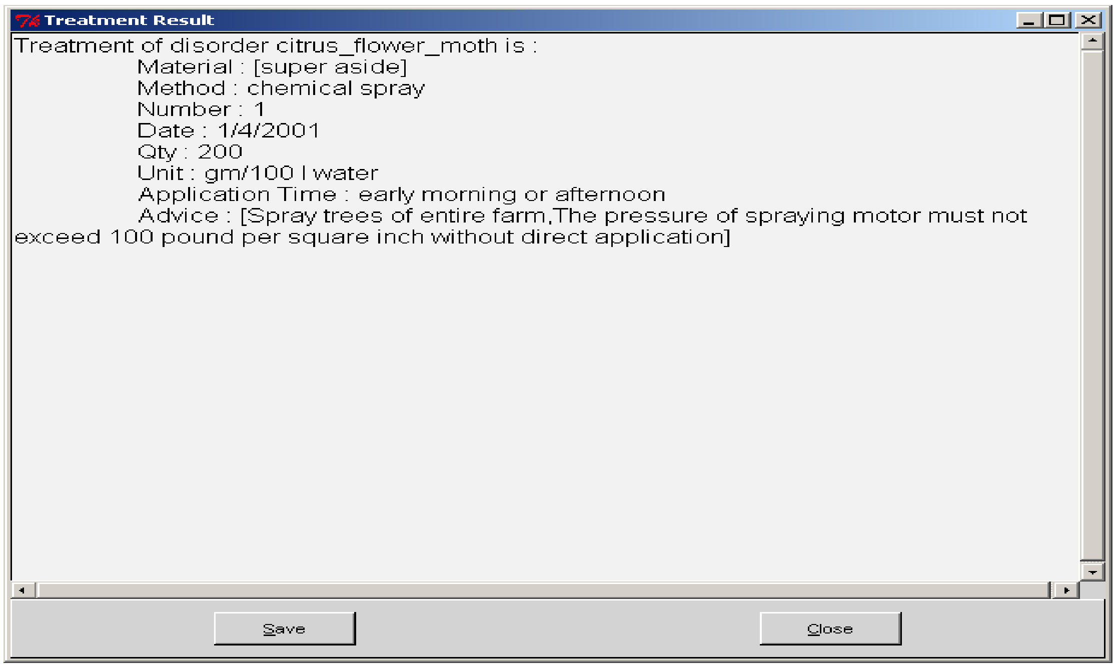
```
Treatment of disorder citrus_white_fly is :
		Material : [vertimec 1.8%]
		Method : chemical spray
		Number : 1
		Date : 1/7/2001
		Qty : 30
		Unit : ml/100 l water
		Application Time : early morning or afternoon
		Advice : [Spray the infested trees only,The pressure of spraying motor must not exceed 100
pound per square inch without direct application]

Treatment of disorder mealy_bug is :
		Material : [K.Z. 95%]
		Method : chemical spray
		Number : 2
		Date : 4/7/2001
		Qty : 1.5
		Unit : L/100 l water
		Application Time : early morning or afternoon
		Advice : [Use fit spraying motor with good mixing. The trees must be completely washed.]

Treatment of disorder manganese_def is :
		Material : [micro element mixture]
		Method : foliage nutrition
		Number : 3
		Date : 7/7/2001
		Qty : []
		Unit : []
		Application Time : early morning or afternoon
		Advice : [The micro elements mixture is formulated, for every 100 lt water, as follow :  30 gm
Iron Chelate (EDTA) + 30 gm Zinc Chelate + 75 Mang. Chelate  + 6 gm Copper Sulfate + 30
Magnesium Sulfate + 0.3 gm Borax]
```

Case3



```
Treatment of disorder scales is :
                Material : [Kimisol 95%]
                Method : chemical spray
                Number : 1
                Date : 1/7/2001
                Qty : 1.6
                Unit : L/100 l water
                Application Time : early morning or afternoon
                Advice : [Use fit spraying motor with good mixing. The trees must be completely
washed.]
```

Case 4



```
Treatment of disorder citrus_flower_moth is :
                Material : [super aside]
                Method : chemical spray
                Number : 1
                Date : 1/4/2001
                Qty : 200
                Unit : gm/100 l water
                Application Time : early morning or afternoon
                Advice : [Spray trees of entire farm,The pressure of spraying motor must not
exceed 100 pound per square inch without direct application]
```

7. Database

The integration is done with the end user in the database. Note that there

File name : citex4.pl

```
:- ensure_loaded('$KROL/lib/oodbc').
citex4ds :: {
      server(citex4ds) &
      uid('') &
      pwd('') &
      super(oodbc)}.
soil_ref_table :: {
      tab(soil_ref_table) &
      col(1, gid, 'SHORT', 2) &
      col(2, did, 'SHORT', 2) &
      col(3, texture, 'TEXT', 50) &
      col(4, water_table_level, 'SINGLE', 4) &
      col(5, ec, 'SINGLE', 4) &
      col(6, ph, 'SINGLE', 4) &
      col(8, fc, 'SINGLE', 4) &
      col(9, pwp, 'SINGLE', 4) &
      condition_item(citex4ds, soil_ref_table, select, gid, =, 'gid of farm_data ', 'SHORT')&
      condition_item(citex4ds, soil_ref_table, select, did, =, 'did of farm_data ','SHORT') &
      query_fs(citex4ds, soil_ref_table, ['All Fields']) &
      sql_select(soil_ref_table, ['SELECT', '*', 'FROM soil_ref_table WHERE','gid = ',
_66993,and,'did = ', _67490]) :-
            farm_data :: get_value(gid(_66993)),
            farm_data :: get_value(did(_67490)) &
      super(citex4ds)}.
water_ref_table :: {
      tab(water_ref_table) &
      col(1, gid, 'SHORT', 2) &
      col(2, did, 'SHORT', 2) &
      col(3, eciw, 'SINGLE', 4) &
      condition_item(citex4ds,water_ref_table,select,did, =, 'did of farm_data ', 'SHORT')&
      condition_item(citex4ds,water_ref_table, select,gid, =,'gid of farm_data ','SHORT') &
      query_fs(citex4ds, water_ref_table, ['All Fields']) &
      sql_select(water_ref_table, ['SELECT', '*', 'FROM water_ref_table WHERE','did = ',
_71188,and,'gid = ', _71685]) :-
            farm_data :: get_value(did(_71188)),
            farm_data :: get_value(gid(_71685)) &
      super(citex4ds)}.
climate_ref_table :: {
      tab(climate_ref_table) &
      col(1, gid, 'SHORT', 2) &
      col(2, did, 'SHORT', 2) &
      col(3, month, 'TEXT', 50) &
      col(4, avg_tc, 'SINGLE', 4) &
      col(5, avg_rh, 'SINGLE', 4) &
      col(6, ash, 'SINGLE', 4) &
      col(7, msh, 'SINGLE', 4) &
```

```
    col(8, ra, 'SINGLE', 4) &
    condition_item(citex4ds,climate_ref_table,select,gid,=,'gid of farm_data ','SHORT')&
    condition_item(citex4ds,climate_ref_table,select,did,=,'did of farm_data', 'SHORT')&
    condition_item(citex4ds,climate_ref_table,select,month,=,'month     of     farm_data',
'SHORT')&
    query_fs(citex4ds, climate_ref_table, ['All Fields']) &
    sql_select(climate_ref_table, ['SELECT', '*', 'FROM climate_ref_table WHERE','gid
= ', _78140,and,'did = ', _78661,and,'month = ', _79158]) :-
            farm_data :: get_value(gid(_78140)),
            farm_data :: get_value(did(_78661)),
            farm_data :: get_value(month(_79158)) &
    super(citex4ds)}.
sector_table :: {
    tab(sector_table) &
    col(1, sid, 'SHORT', 2) &
    col(2, sname, 'TEXT', 50) &
    condition_item(citex4ds, sector_table, select, sid, =, 'sid of farm_data ', 'SHORT') &
    query_fs(citex4ds, sector_table, ['All Fields']) &
    sql_select(sector_table,['SELECT','*','FROM sector_table WHERE','sid=', _8392]) :-
    farm_data :: get_value(sid(_8392)) &
    super(citex4ds)}.
governorate_table :: {
    tab(governorate_table) &
    col(1, sid, 'SHORT', 2) &
    col(2, gid, 'SHORT', 2) &
    col(3, gname, 'TEXT', 50) &
    condition_item(citex4ds,governorate_table,select,gid,=,'gid of farm_data','SHORT')&
    query_fs(citex4ds, governorate_table, ['All Fields']) &
    sql_select(governorate_table, ['SELECT', '*', 'FROM governorate_table WHERE','gid
= ', _11888]) :-
            farm_data :: get_value(gid(_11888)) &
    super(citex4ds)}.
directorate_table :: {
    tab(directorate_table) &
    col(1, sid, 'SHORT', 2) &
    col(2, gid, 'SHORT', 2) &
    col(3, did, 'SHORT', 2) &
    col(4, dname, 'TEXT', 50) &
    condition_item(citex4ds,  directorate_table,  select,  gid,  =,  'gid  of  farm_data  ',
'SHORT') &
    condition_item(citex4ds,  directorate_table,  select,  did,  =,  'did  of  farm_data  ',
'SHORT') &
    query_fs(citex4ds, directorate_table, ['All Fields']) &
    sql_select(directorate_table, ['SELECT', '*', 'FROM directorate_table WHERE','gid =
', _15961,and,'did = ', _16458]) :-
            farm_data :: get_value(gid(_15961)),
            farm_data :: get_value(did(_16458)) &
    super(citex4ds)}.
farm_table :: {
    tab(farm_table) &
    col(1, sid, 'SHORT', 2) &
    col(2, gid, 'SHORT', 2) &
    col(3, did, 'SHORT', 2) &
    col(4, fid, 'SHORT', 2) &
    col(5, fname, 'TEXT', 50) &
    col(6, area, 'SINGLE', 4) &
    col(7, plantation_date, 'DATE', 0) &
    col(8, irr_system, 'TEXT', 50) &
    col(9, fert_system, 'TEXT', 50) &
    col(10, drainage_system, 'TEXT', 50) &
    col(11, nt, 'SHORT', 2) &
    col(12, r_dist, 'SINGLE', 4) &
    col(13, t_dist, 'SINGLE', 4) &
    col(14, water_source, 'TEXT', 50) &
    col(15, user_cont_water, 'TEXT', 50) &
    col(16, variety_name, 'TEXT', 50) &
    col(17, s_s_month, 'SHORT', 2) &
    condition_item(citex4ds, farm_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
    condition_item(citex4ds, farm_table, select, did, =, 'did of farm_data ', 'SHORT') &
    condition_item(citex4ds, farm_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
    query_fs(citex4ds, farm_table, ['All Fields']) &
    sql_select(farm_table, ['SELECT', '*', 'FROM farm_table WHERE','gid = ',
_27634,and,'did = ', _28155,and,'fid = ', _28652]) :-
            farm_data :: get_value(gid(_27634)),
            farm_data :: get_value(did(_28155)),
            farm_data :: get_value(fid(_28652)) &
    super(citex4ds)}.
soil_table :: {
    tab(soil_table) &
    col(1, gid, 'SHORT', 2) &
    col(2, did, 'SHORT', 2) &
    col(3, fid, 'SHORT', 2) &
    col(4, texture, 'TEXT', 50) &
    col(5, water_table_level, 'SINGLE', 4) &
    col(6, ec, 'SINGLE', 4) &
    col(7, ph, 'SINGLE', 4) &
    col(8, fc, 'SINGLE', 4) &
    col(9, pwp, 'SINGLE', 4) &
    condition_item(citex4ds, soil_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
    condition_item(citex4ds, soil_table, select, did, =, 'did of farm_data ', 'SHORT') &
    condition_item(citex4ds, soil_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
    query_fs(citex4ds, soil_table, ['All Fields']) &
    sql_select(soil_table, ['SELECT', '*', 'FROM soil_table WHERE','gid = ',
_35614,and,'did = ', _36135,and,'fid = ', _36632]) :-
            farm_data :: get_value(gid(_35614)),
            farm_data :: get_value(did(_36135)),
            farm_data :: get_value(fid(_36632)) &
    super(soil_ref_table)}.
climate_table :: {
    tab(climate_table) &
    col(1, gid, 'SHORT', 2) &
    col(2, did, 'SHORT', 2) &
    col(3, fid, 'SHORT', 2) &
    col(4, month, 'TEXT', 50) &
    col(5, avg_tc, 'SINGLE', 4) &
    col(6, avg_rh, 'SINGLE', 4) &
    col(7, ash, 'SINGLE', 4) &
    col(8, msh, 'SINGLE', 4) &
    col(9, ra, 'SINGLE', 4) &
    condition_item(citex4ds, climate_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
    condition_item(citex4ds, climate_table, select, did, =, 'did of farm_data ', 'SHORT') &
    condition_item(citex4ds, climate_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
    condition_item(citex4ds,climate_table,select,month,=,'month of farm_data',
'SHORT') &
    query_fs(citex4ds, climate_table, ['All Fields']) &
    sql_select(climate_table, ['SELECT', '*', 'FROM climate_table WHERE','gid = ',
_43686,and,'did = ', _44207,and,'fid = ', _44728,and,'month = ', _45225]) :-
            farm_data :: get_value(gid(_43686)),
            farm_data :: get_value(did(_44207)),
            farm_data :: get_value(fid(_44728)),
            farm_data :: get_value(month(_45225)) &
    super(climate_ref_table)}.
water_table :: {
    tab(water_table) &
    col(1, gid, 'SHORT', 2) &
    col(2, did, 'SHORT', 2) &
    col(3, fid, 'SHORT', 2) &
    col(4, eciw, 'SINGLE', 4) &
    condition_item(citex4ds, water_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
    condition_item(citex4ds, water_table, select, did, =, 'did of farm_data ', 'SHORT') &
    condition_item(citex4ds, water_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
    query_fs(citex4ds, water_table, ['All Fields']) &
    sql_select(water_table, ['SELECT', '*', 'FROM water_table WHERE','gid = ',
_49788,and,'did = ', _50309,and,'fid = ', _50806]) :-
            farm_data :: get_value(gid(_49788)),
            farm_data :: get_value(did(_50309)),
            farm_data :: get_value(fid(_50806)) &
    super(water_ref_table)}.
soil_assessment_table :: {
    tab(soil_assessment_table) &
    col(1, gid, 'SHORT', 2) &
    col(2, did, 'SHORT', 2) &
    col(3, fid, 'SHORT', 2) &
    col(4, boron, 'SINGLE', 4) &
    col(5, chloride_sulphate, 'SINGLE', 4) &
    col(6, rsc, 'SINGLE', 4) &
    col(7, sar, 'SINGLE', 4) &
    col(8, profile_depth, 'SINGLE', 4) &
    col(9, ca_carbonate, 'SINGLE', 4) &
    col(10, max_d_tc_ss, 'SINGLE', 4) &
```

```
     col(11, min_d_rh_ss, 'SINGLE', 4) &
     col(12, esp, 'SINGLE', 4) &
     condition_item(citex4ds, soil_assessment_table, select, gid, =, 'gid of farm_data ',
'SHORT') &
     condition_item(citex4ds, soil_assessment_table, select, did, =, 'did of farm_data ',
'SHORT') &
     condition_item(citex4ds, soil_assessment_table, select, fid, =, 'fid of farm_data ',
'SHORT') &
     query_fs(citex4ds, soil_assessment_table, ['All Fields']) &
     sql_select(soil_assessment_table, ['SELECT', '*', 'FROM soil_assessment_table
WHERE','gid = ', _59414,and,'did = ', _59935,and,'fid = ', _60432]) :-
             farm_data :: get_value(gid(_59414)),
             farm_data :: get_value(did(_59935)),
             farm_data :: get_value(fid(_60432)) &
     super(citex4ds)}.
select_table :: {
     tab(select_table) &
     col(1, sid, 'SHORT', 2) &
     col(2, gid, 'SHORT', 2) &
     col(3, did, 'SHORT', 2) &
     col(4, fid, 'SHORT', 2) &
     condition_item(citex4ds, select_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
     condition_item(citex4ds, select_table, select, did, =, 'did of farm_data ', 'SHORT') &
     condition_item(citex4ds, select_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
     query_fs(citex4ds, select_table, ['All Fields']) &
     sql_select(select_table, ['SELECT', '*', 'FROM select_table']) &
     super(citex4ds)
}.
```

8. Multimedia

8.1. Multimedia Component

These names of indexes are updated in the implementation report

Book no/ page	new Impl. report	Old Impl. report	Page in imp. report
Front page	الفهرس الرئيسى	العودة للوسائط المتعددة للموالح	٤

8.2. Multimedia Linking

- The following words are updated in implementation report

رقم الصورة	الكلمة old Implementation report	الكلمة new Implementation report	موقع الكلمة السطر	موقع الكلمة الفقرة	صفحة	الكتاب
٣٧	اضافة الحشائش	ازالة الحشائش	٣	٤	٢٦	٣
٣٥	مقاومة الحشائش	مقاومة الحشرة	١	٣	٤٩	٣
٣٢	الحشرات القشرية	مقاومة الحشرات القشرية و البق الدقيقى	٨	١	٨	١١
٤٣	المن	مقاومته(علاج المن)	٥	١	٧	١٣
٣٢	بق الموالح الدقيقى	وتتم المقاومة(البق الدقيقى)	٣	١	١٤	١٣
٣٣	الأشنات	تقاوم كيماويا بالرش(الأشنات)	٧	١	٢٥	١٣

- The following words page number reference is updated in implementation report.

Page reference in new impl. report	Page reference in old impl. report	الكلمة	الصفحة in design	الكتاب
٣٦	٣٢	ذبابة الفاكهة	٢٣	١١

- The following words have "الفقرة" number is updated in implementation report.

design report Page	الفقرة in old impl. report	الكلمة	الفقرة in new implementation report	الصفحة	الكتاب
٣٤	١	اليافاوى	٢	٢٣	٩
٣٤	٢	الفالنشيا	٣	٢٣	٩
٣٤	١٢	الجريب فروت	١	٣٦	٩
٣٤	١	البق الدقيقى	٢	٤٦	٩
٣٤	٢	ذبابة الموالح البيضاء	٣	٤٦	٩
٣٤	٣	دودة أزهار الموالح	١	٤٧	٩

٣٤	١	المن	٣	٤٧	٩
٣٤	٣	ذبابة الفاكهة	٤	٤٧	٩
٣٧	٢	الأشنات	٣	٤٦	١١
٣٧	٣	الأشنات	١	٤٧	١١

- **The following images file name is updated in the implementation report**

Old Imp. report Page	الكتاب	الصور	الفقرة	الصفحة	الكلمة
٨	١	Pic6.gif	٢	٣٢	الليمون المخرفش
٨	١	Pic6.gif	١	٣٩	الليمون المخرفش
١٢	١	Dfrb1.gif	٢	١١٦	عفن الثمار الأخضر
٢٨	٨	Nml.gif	٣	١٣	نقص فى عنصر الماغنسيوم
٢٩	٨	Nml.gif	٣	١٤	أعراض النقص
٢٩	٨	Ni1.gif	١	١٧	نقص تركيز عنصر الحديد
٢٩	٨	Ni1.gif	١	١٧	أعراض نقص الحديد
٢٩	٨	Nm1.gif	١	١٧	المغنيسيوم
٢٩	٨	Mzmg1.gif	٢	١٧	أعراض نقص عنصر الزنك
٢٩	٨	Nmg1.gif	٣	١٨	أعراض نقص عنصر المنجنيز
٣١	٩	Var3.gif	٢	١٠	الفالنشيا
٣٢	٩	Var3.gif	٤	١٠	الخليل الأبيض
٣٢	٩	Var3.gif	١	١١	الفالنشيا
٣٢	٩	Var12a.gif	٤	١٤	جريب فروت
٤٤	١٣	Ps1.gif	١	٣٢	لسعة الشمس
٤٤	١٣	Mzmg1.gif	١	٣٦	نقص الزنك
٤٤	١٣	Ni1.gif	١	٣٧	نقص الحديد
٤٤	١٣	Nmg1.gif	١	٣٨	نقص المنجنيز
٤٤	١٣	Nm1.gif	١	٣٩	نقص المغنيسيوم
٤٤	١٣	Pf2.gif	١	٤١	تشقق الثمار
٤٤	١٣	Pfr1.gif	١	٥٦	الصقيع
٤٦	١٤	Po1.gif	٢	٢٥	التبقع الزيتى
٤٦	١٤	Pe1.gif	٢	٣٠	عفن الطرف
٤٦	١٤	Pe1.gif	٣	٣٣	عفن الطرف
٤٧	١٥	Mzmg1.gif	٢	٦	نقص الزنك
٤٧	١٥	Nmg1.gif	٢	٦	نقص المنجنيز
٤٧	١٥	Ni1.gif	٢	٦	نقص الحديد

- the following image links are added to the implementation report

رقم الصورة	الكلمة	موقع الكلمة		صفحة	الكتاب
		السطر	الفقرة		

رقم الصورة	الكلمـة	موقع الكلمة		صفحة	الكتاب
		السطر	الفقرة		
٣	الهاملن	١	٢	٧	١٢
٨	فالنشيا	١	٢	١٠	١٢
٨	فالنشيا	١	٢	١١	١٢
١٥	اليافاوى	١	١	١٣	١٢
١	اليوسفى الكينو	٢	١	٢٣	١٢
١١	الجريب فروت روبى رد	١	١	٣٣	١٢
٩	الكمكوات	١	١	٣٧	١٢
١٠	اليوسفى كليوباترا	١	٢	٤٠	١٢
١٠	اليوسفى كليوباترا	١	٢	٤١	١٢
٦	الليمون المخرفش	١	١	٤٢	١٢
٧	البرتقال ثلاثى الاوراق	١	١	٤٤	١٢
٧	البرتقال ثلاثى الاوراق	١	٢	٤٤	١٢

- the following number of references is updated in the implementation report

design report Page	رقم الصورة أو لقطة الفيديو	Number of ref in Old Implementation report	Number of ref in new implementation report	الكلمة	صفحة	الكتاب
٣٠	٦ص	١	٣	الليمون المخرفش	٣٩	١
٣٠	٩ص	٢	٣	الكمكوات	٤٠	١
٣١	٨ص	١	٢	فالنشيا	١٠٠	١
٤١	١٢ف	–	١	التطعيم	٢٣	٥
٤٢	٢٦ف	–	١	مقاومة الحشائش كيماويا	٤٣	٩
٣٥	٥٣ص	١	٤	النيماتودا	٧٢	١٠
٣٥,٣٦	٥٣ص	٢	٨	النيماتودا	٧٤	١٠
٣٦	٥٣ص	٢	٤	النيماتودا	٧٨	١٠
٣٨	٢٥ص	–	١	التصمغ	٢٣	١٣

- the following video clips file names is updated in implementation

Old Imp. report Page	الكتاب	اللقطة	الفقرة	الصفحة	الكلمة
٧	١	3.MPG	١	٨	مناطق انتاج الموالح فى العالم
٤٢	١	33.MPG	١	١١٥	الأشنات
٤٦	٣	1. MPG	٢	٢	تبلغ المساحة المنزرعة
٢٣	٥	4.MPG	٤	٢٣	التطعيم

- The number of references for the following video clips are updated in the implementation report.

design report Page	رقم الصورة	Number of ref in Old Implementation report	Number of ref in New implementation report	الكلمة	صفحة	الكتاب
٣٩	١٧	١	٣	تجهيز الجور و غرس أشجار الموالح	٦٧	١

- the following code in implementation report is updated

File name	old Implementation report	new Implementation report
Image.htm	<p align="right"><b><span style='mso-bidi-font-family: "Traditional Arabic"'><a href="Index.htm#bb">	<p align="right"><b><span style='mso-bidi-font-family: "Traditional Arabic"'><a href="multimedia.htm#bb">

9. Integration User Interface

File name: main.pl

```
:- use_module(library(ordsets), [ord_subtract/3]).
:- use_module(library(charsio), [format_to_chars/3,read_from_chars/2]).
:- use_module(library(system), [delete_file/1,file_exists/1, exec/3,
                        make_directory/1, working_directory/2, system/1, environ/2]).
:- ensure_loaded('$KROL/lib/menubar').
:- ensure_loaded('$KROL/lib/directory').
:- ensure_loaded([
        '$KROL/lib/date',
        '$KROL/lib/log',
        '$KROL/lib/krol_init',
        '$KROL/lib/stack',
        '$KROL/lib/msgs',
        '$KROL/lib/tk_user',
        '$KROL/lib/back_dlg',
        '$KROL/lib/database',
        '$KROL/lib/history',
        '$KROL/lib/gt',
        '$KROL/lib/rule_exp',
        '$KROL/lib/inferenc',
        '$KROL/lib/tab',
        '$KROL/lib/fun']).
:- ensure_loaded('diag_system').
:- ensure_loaded('asesment/ass_system').
:- ensure_loaded('plantcar/pl_main').
appl_pdw :: {
attributes([
% Control fllags
        finding(0),addc(0), disorderc(0),cmf(0), newf(0), node(0), server(0),
% Counters
        dc(0), sc(0),
% Subsystem Flage
        sys([])
]) &
window_title('Citex Expert System') &
widget(gttpdm, []) &
geometry('400x300+100+100') &
menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = db,
        Txt = 'Data Base',
        Bal = 'Data Base\\nMenu',
        Status = 'User' &
menu(db, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'User',
        Comm = 'userdatabase',
        Acc = 'Ctrl+u',
        Acc1 = '<Control-U>' &
menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = mm,
        Txt = 'Multi Media',
        Bal = 'Multi Media\\nMenu',
        Status = 'multimedia' &
menu(mm, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'MultiMedia',
        Comm = 'multimedia',
        Acc = 'Ctrl+m',
        Acc1 = '<Control-M>' &
menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = expert_system,
        Txt = 'Expert System',
        Bal = 'Expert System\\nMenu',
        Status = 'Diagnosis , Treatment' &
menu(expert_system, [Label,Underline,Comm,Acc,Acc1], command) :-
        Label = 'Assessment ',
        Underline = 0,
        Comm = 'assessment',
        Acc = 'Ctrl+a',
        Acc1 = '<Control-A>' &
menu(expert_system, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'PlantCare ',
        Comm = 'plantcare',
        Acc = 'Ctrl+p',
        Acc1 = '<Control-P>' &
menu(expert_system, [Label,Underline,Comm,Acc,Acc1], command) :-
        Label = 'Diagnosis ',
        Underline = 0,
        Comm = 'diagnosis',
        Acc = 'Ctrl+d',
        Acc1 = '<Control-D>' &
```

```
menu(expert_system, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'Treatment ',
        Comm = 'treatment',
        Acc = 'Ctrl+t',
        Acc1 = '<Control-T>' &
menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = exit,
        Txt = 'Exit',
        Bal = 'Exit\\nMenu',
        Status = 'Exit' &
menu(exit, [Label,1,Comm,Acc,Acc1], command) :-
        Label = 'Exit ',
        Comm = 'exit',
        Acc = 'Ctrl+x',
        Acc1 = '<Control-x>' &
status(toolbar, S) :-
        tcl :: eval('set tbar', S1),
        :name(S, S1) &
super(pdwmenu)}.
main:-
        tcl :: init,
        appl_pdw :: display,
        tcl :: end.
userdatabase:-
        exec('CitexDb.exe', [null,null,null], _).
multimedia:-
        environ('KROL', KROL),
        format_to_chars('~w/bin/IEXPLORE.EXE ~w/multimedia/multimedia.htm', [KROL,
KROL], CS),
        name(C, CS),
        exec(C, [null,null,null], _).
assessment:-
        ass_start.
plantcare:-
        plant_main.
diagnosis:-
        appl_pdw :: set(sys(diag)),diag_main.
treatment:-
        appl_pdw :: set(sys(treat)),treat_main.
exit:-
        appl_pdw :: destroy.
```