

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

CZ3005 ARTIFICIAL INTELLIGENCE

**IMPLEMENTING A TALKING BOX WITH PROLOG
ASSIGNMENT 4 REPORT**

Shearman Chua Wei Jie (U1820058D)

LAB GROUP: TSP1

Contents

Introduction.....	3
Sympathetic Doctor interactive Talking Box.....	3
Scope of interactive Talking Box.....	3
Implementation of Sympathetic Doctor interactive Talking Box	3
Starting the Sympathetic Doctor interactive Talking Box	4
nextQuestion Predicates Functionality.....	6
answer_h and answer Predicates Functionality.....	7
diagnose_h and diagnose Predicates Functionality	8
human_symptom and human_illness Predicates Functionality	9
openings and opening Predicates Functionality	10
question_starts, question_start and human_diagnose Predicates Functionality	11
pain_level, mood_level, symptom_library and illness Predicates Functionality	12
reply_question, reply_diagnose and response Predicates Functionality	13
Demonstration of the Sympathetic Doctor interactive Talking Box.....	14
Initial Start Up.....	14
After Answering Pain Level	14
After Answering Mood Consideration.....	14
Diagnosis	15
Exiting the Sympathetic Doctor interactive Talking Box Script	16
Conclusions.....	16

Question 4: Patient with a Sympathetic Doctor

Introduction

With the advancement of technology, as well as the increasing accessibility of the internet, many in the modern day and age will often choose to look up on the internet to find out what illness or disease they may have, based on the symptoms they present, over going to visit a doctor. This is especially so if their symptoms are mild. The reason behind this phenomenon is to avoid spending unnecessary money to see a doctor, if the cure for their disease is readily available in pharmacies and cost less, as well as to avoid long waiting times to see a doctor at clinics or hospitals.

This is especially the case with the ongoing Covid-19 pandemic. Most people would want to avoid going to see a doctor, especially if their disease is non-flu related (such as physical pain, rashes, etc.), to minimize the risk of being in contact with patients presenting Covid-19 symptoms who may also be visiting the doctor.

Therefore, in times like this, it would be beneficial to have an interactive Talking Box, which is able to ask questions about a patient's symptoms and provide a diagnosis, for commonly encountered illnesses and diseases. This interactive Talking Box will help provide an initial diagnosis for patients before they decide if they require further medical attention. Having an interactive Talking Box that is able to diagnose diseases will help ease the crowd at clinics and hospitals, which helps minimize the contact of patients presenting Covid-19 symptoms with others, as well as reduce the workload of doctors during these tumultuous times.

Sympathetic Doctor interactive Talking Box

Scope of interactive Talking Box

The interactive Talking Box will be programmed in the Prolog language. The Prolog script will take-up the persona of a sympathetic doctor, where he is required to:

1. Converse with a patient who can answer only yes or no.
2. Diagnose the patient's condition while asking questions sensitively depending upon patient's pain level and mood level.
3. Able to choose 5 or more different mood considerations (calm, angry, etc.) and 5 or more levels of pain.
4. Able to diagnose five or more diseases, where each disease is characterized by 5 or more symptoms

Implementation of Sympathetic Doctor interactive Talking Box

As mentioned, the interactive Talking Box will be implemented using the Prolog programming language. The Talking Box will take the form of an automated Chatbot, where the user will answer yes or no to the questions that the Talking Box ask, and at the end of the questioning, provide a diagnosis for the user.

From my implementation of the Sympathetic Doctor interactive Talking Box, the Sympathetic Doctor, named Dr Bot, is able to diagnose these common illnesses and diseases:

1. Allergic reaction to bee sting
2. Moderate reaction to bee sting
3. Bee sting
4. Mild allergic reaction
5. General allergic reaction

6. Eczema
7. Cut
8. Moderate cut
9. Deep cut
10. General injury
11. General severe injury
12. Blunt force injury
13. Severe blunt force injury
14. Fracture
15. Sprain
16. Dengue Fever
17. Food poisoning
18. Cold
19. Flu
20. Severe Flu
21. Headache
22. Migraine
23. Viral fever
24. Heat exhaustion

The Sympathetic Doctor interactive Talking Box will ask the user questions regarding the symptoms that he might be having and uses an elimination method to provide the user with a final diagnosis or diagnoses. The Sympathetic Doctor interactive Talking Box will run continuously for the user to consult the Sympathetic Doctor, Dr Bot, for diagnoses until the letter 'q' is entered to the Prolog program to terminate it.

The code uses the `assert()` function to tell the Knowledge-Based System (KBS) what are questions the user has answered and what are the symptoms the user has. In order to dynamically select subsequent questions to ask, we need to use the "dynamic" command to let Prolog allow such assertions to modify the KBS on the fly:

```
:- dynamic answered/1, have_symptom/1, pain/1, mood/1, ready_to_diagnose/1.
```

For our Prolog script, the **answered**, **have_symptom**, **pain**, **mood**, and **ready_to_diagnose** predicates are required to be asserted when the program is being run, therefore, within the Prolog script, we declare these predicates as dynamic.

Starting the Sympathetic Doctor interactive Talking Box

```
%%first question
ask(0):-nl,write('Hi, I am Dr Bot, let me help diagnose you today!'), reply_start.

%%first reply
reply_start:-
    nl,write('To help me begin, '),           % opening
    question_start(QS),                       % get question start
    nextQuestion(Question),
    human_symptom(Question, Human_question),   % get next question
    write(QS), write(Human_question),write('?'),nl,write('Respond with yes or no (or press q to abort):'),read(Response),response(Question,Response),!
.

?-ask(0). ▲
```

The Sympathetic Doctor interactive Talking Box Prolog script is able automatically started asking the user questions when the Prolog file is opened in Prolog. This is due to the fact that the line **?-ask(0)** is declared in the script which will automatically run the first function **ask(0)**, when the script is opened.

In the Prolog script, we make use of the **write()** predicate that is built-in to Prolog to display messages to the users to tell them what to do. The **ask(0)** predicate writes the introductory greeting to the user, as the persona of Dr Bot, a Sympathetic Doctor who helps to diagnose a user's illness. The **ask(0)** function then calls the **reply_start** function to start asking the user questions. In the **reply_start** function, we call the **question_start** function to get the head of the question, **nextQuestion** function to get the question to be asked, **human_symptom** function to convert the question to be asked into a string and finally write it onto the Prolog console using the **write()** predicate. The **read()** predicate is then used to read the user's response and then the response is passed to the **response()** predicate to handle the user's response.



Display in Prolog when Prolog Script is started

As can be seen from the figure above, when the Sympathetic Doctor interactive Talking Box Prolog script is opened in Prolog, the Prolog displays the initial greeting and begins asking the user questions. The first question the Sympathetic Doctor interactive Talking Box will ask is regarding the level of pain the user is feeling. There are 5 levels of pain in the Sympathetic Doctor Prolog script, namely:

1. no_pain (No pain)
2. mild_pain (Mild pain)
3. moderate_pain (Moderate pain)
4. severe_pain (Severe pain)
5. excruciating_pain (Excrutiating pain)

Only when the user answers 'yes' to one of the pain levels, or when the 5 levels of pain are asked finished and the user answers 'no' to all of them, then the Prolog script proceeds to ask the user regarding the mood the user is feeling. There are 5 different mood considerations in the Sympathetic Doctor Prolog script, namely:

1. angry
2. upset
3. frustrated
4. fatigued
5. calm

Only when the user answers ‘yes’ to one of the mood considerations, or when the 5 mood considerations are asked finished and the user answers ‘no’ to all of them, then the Prolog script proceeds to ask the user about the symptoms the user have.

```
pain.      % store selected pain, only one will be selected
mood.     % store selected mood, only one will be selected
ready_to_diagnose. % flag setted when all questions are asked and answered
have_symptom. % symptoms with positive answer, including symptoms, pain, and mood
answered.      % answered items, including symptoms, pain, and mood
```

From the figure above, we can see the predicates, **answered**, **have_symptom**, **pain**, **mood**, and **ready_to_diagnose**, where the **pain** predicate is used to store the level of pain the user has selected, the **mood** predicate is used to store the mood the user has selected, the **ready_to_diagnose** predicate is used to store whether the Sympathetic Doctor Prolog script is ready to make a diagnosis, the **answered** predicate is used to store the questions the user has answered, and the **have_symptom** predicate is used to store the symptoms the user answered ‘yes’ to.

nextQuestion Predicates Functionality

```
%% ask symptom
nextQuestion(Next):-
    % pain and mood has finished, should ask Symptom
    pain_level(Pain_level),
    mood_level(Mood_level),
    symptom_library(Symptom_library),
    list_finished(Pain_level, _, If_pain_finished), % determine Pain_level has all been answered
    list_finished(Mood_level, _, If_mood_finished), % determine Mood_level has all been answered
    list_finished(Symptom_library, ValidChoices, _),
    (
        (pain_selected;If_pain_finished), % if one of pain is selected or Pain_level is answered through
        (mood_selected;If_mood_finished) % if one of mood is selected or Mood_level is answered through
    ),!,
    random_member(Next, ValidChoices).

%% ask mood
nextQuestion(Next):-
    % pain has finished, should ask mood
    pain_level(Pain_level),
    mood_level(Mood_level),
    list_finished(Pain_level, _, If_pain_finished), % determine Pain_level has all been answered
    list_finished(Mood_level, ValidChoices, _),
    (
        pain_selected; If_pain_finished % if one of pain is selected or Pain_level is answered through
    ),!,
    random_member(Next, ValidChoices).

%% ask pain
nextQuestion(Next):-
    % pain have not been selected
    pain_level(Pain_level),
    list_finished(Pain_level, ValidChoices, _),!,
    random_member(Next, ValidChoices).
```

The figure above shows the 3 **nextQuestion** Predicates that are used to decide the next question to be asked to the user.

The very first **nextQuestion** predicate first retrieves the 5 levels of pain by calling the **pain_level** predicate, retrieves the 5 mood considerations by calling the **mood_level** predicate, and retrieves the available symptoms in our script by calling the **symptom_library** predicate. The three **list_finished** predicates are then used to help determine if the 5 levels of pain, 5 mood considerations are asked finished and stores the results as **If_pain_finished** and **If_mood_finished** respectively, and also, helps return the choices of symptoms that had not been asked as **ValidChoices**. Next, the **nextQuestion** predicate checks if one of the pain level is selected or if the 5 levels of pain are asked finished, as well as if one of the mood is selected or if the 5 mood considerations are asked finished, if yes, the **nextQuestion** predicate proceed to choose one of the symptoms to ask the user from the list of symptoms that had not been asked in **ValidChoices** by calling the **random_member** predicate to choose one of the symptoms from **ValidChoices**, if not we proceed to the second **nextQuestion** predicate.

The second **nextQuestion** predicate first retrieves the 5 levels of pain by calling the **pain_level** predicate, retrieves the 5 mood considerations by calling the **mood_level** predicate. The two **list_finished** predicates are then used to help determine if the 5 levels of pain are asked finished and stores the results as **If_pain_finished**, and also, helps return the choices of moods that had not been asked as **ValidChoices**. Next, the **nextQuestion** predicate checks if one of the pain level is selected or if the 5 levels of pain are asked finished, , if yes, the **nextQuestion** predicate proceed to choose one of the moods to ask the user from the list of moods that had not been asked in **ValidChoices** by calling the **random_member** predicate to choose one of the moods from **ValidChoices**, if not we proceed to the third **nextQuestion** predicate.

The third **nextQuestion** predicate first retrieves the 5 levels of pain by calling the **pain_level** predicate. The **list_finished** predicate is then used to helps return the choices of pain that had not been asked as **ValidChoices**. Then, the **nextQuestion** predicate proceed to choose one of the pain level to ask the user from the list of pain that had not been asked in **ValidChoices** by calling the **random_member** predicate to choose one of the pain level from **ValidChoices**.

The 3 **nextQuestion** Predicates helps ensure that the doctor has determined the pain level and the mood of the patient before moving on to ask the user about his symptoms.

answer_h and answer Predicates Functionality

```
%% helper function for positive answer of answer/2, depending of answering a pain, mood or symptom, make different action
answer_h(Q):-
    pain_level(Pain_level),
    mood_level(Mood_level),
    symptom_library(Symptom_library),
    (
        member(Q, Pain_level) -> (assert(pain(Q)),assert(have_symptom(Q)));    % if Q is a pain
        member(Q, Mood_level) -> assert(mood(Q));    % if Q is a mood
        member(Q, Symptom_library)->assert(have_symptom(Q))    % otherwise Q is a symptom
    ).
    % add Q to have_symptom

% Interface for answering question, eg. query answer(cough, yes) in prolog will tell the system you have cough
answer(Q, Answer):-
    assert(answered(Q)),
    (
        Answer == yes -> answer_h(Q); Answer == q -> abort; true
    ),
    % check whether wvery thing has been answered, in other words whether ready to make diagnos
    symptom_library(Symptom_library),
    (
        list_finished(Symptom_library, _, If_symptom_finished),    % whether symptom finished
        diagnose(Result),
        (same(Result) -> assert(ready_to_diagnose(yes));true),
        (If_symptom_finished -> assert(ready_to_diagnose(yes));true)
    ).

% check if all illnesses same
same([]).
same([_]).
same([X,X|T]) :- same([X|T]).
```

In the figure above, we have the **answer_h()** and **answer()** predicates. When the **response()** predicate is called, the response by the user is passed to the **answer()** predicate to handle the answer given by the user. The **answer()** predicate first asserts the answered question by calling **assert(answered(Q))** which tells the Knowledge-Based System (KBS) what questions have been answered by the user. Next, if the user has answered ‘yes’ for that question, the **answer_h()** predicate is called.

The **answer_h()** predicate then helps to determine if the question answered is a **pain**, **mood** or a **symptom** and uses the assert function to assert the **pain**, **mood** or **have_symptom** predicates, respectively.

We then go back to the **answer()** predicate to check if either, there is only one possible diagnosis the Sympathetic Doctor interactive Talking Box Prolog script has narrowed down to by first calling the **diagnose()** predicate to get the possible diagnoses based on the symptoms answered ‘yes’ and the using the **same()** predicate to check if the diagnoses that were returned are the same illness, or if all of the symptoms have been asked. If either condition is true, the **ready_to_diagnose** predicate is set to yes to inform the Prolog script to write out the diagnosis to the user.

diagnose_h and diagnose Predicates Functionality

```
%% helper function for making diagnosis, determine whether one illness is satisfied
diagnose_h(X):-
    findall(A, have_symptom(A), Symptom_list),
    illness(X, L),
    is_subset(Symptom_list,L).

% Interface for making diagnosis, collect all satisfied illnesses.
diagnose(L):-
    findall(A, diagnose_h(A), L).

^%% determine whether a list is empty
list_empty([], true).
list_empty([_|_], false).

%% determine whether a list L1 is a subset of L2
% eg. is_subset(L1, L2).
is_subset([], _).
is_subset([H|T], L):-
    member(H,L), is_subset(T,L).

pain_selected:-
    pain(excruciating_pain);
    pain(severe_pain);
    pain(moderate_pain);
    pain(mild_pain);
    pain(no_pain).

mood_selected:-
    mood(angry);
    mood(upset);
    mood(frustrated);
    mood(fatigued);
    mood(calm).

check_diagnose:- ready_to_diagnose(yes).
```

Next, we have the **diagnose()** and **diagnose_h()** predicates to help use to help us retrieve all the possible diagnoses that matches the symptoms that the user has answered ‘yes’ to.

First, the **diagnose()** predicate is called, which uses the **findall()** predicate to find all the possible diagnoses given by **diagnose_h()** predicate and puts them into a list. The **diagnose_h()** predicate first retrieves the symptoms that the user has answered ‘yes’ to in the Knowledge-Based System (KBS), then it also retrieves an illness from the Prolog knowledge base, and finally determine if the symptoms the user has answered ‘yes’ to is a subset of the symptoms of the illness that was retrieved, if yes, the illness is returned to the calling function. Using the **diagnose()** and **diagnose_h()** predicates, we are able to retrieve all the possible diagnoses that matches the symptoms that the user has answered ‘yes’ to.

human_symptom and human_illness Predicates Functionality

```
human_symptom(excruciating_pain , 'excrutiating pain ').
human_symptom(severe_pain , 'severe pain ').
human_symptom(moderate_pain , 'moderate pain ').
human_symptom(mild_pain , 'mild pain ').
human_symptom(no_pain , 'no pain ').

human_symptom(angry , 'angry ').
human_symptom(upset , 'upset ').
human_symptom(frustrated , 'frustrated ').
human_symptom(fatigued , 'fatigued ').
human_symptom(calm , 'calm ').

human_symptom(physical_pain , 'that you have localized external physical pain(e.g. abdominal, arm, head, stomach), excluding headaches and full body aches ').
human_symptom(insect , 'that you have been stung by an insect or have been in an area around insects ').
human_symptom(swollen , 'that there is swelling on affected area ').
human_symptom(stinging , 'a stinging sensation on affected area ').
human_symptom(sting_area , 'an instant, sharp burning pain at the pain site(if no open wound) ').
human_symptom(welt , 'that there is a red welt at the pain area(if no open wound) ').
human_symptom(extreme_redness , 'there is extreme redness at affected area ').
human_symptom(numb , 'that there is localized numbness ').
human_symptom(skin_reactions , 'there are skin reactions, including hives and itching and flushed or pale skin ').
human_symptom(difficulty_breathing , 'that you have difficulty breathing ').
human_symptom(tongue_swell , 'that you have swelling of the throat and tongue ').
human_symptom(weak_pulse , 'that you have a weak, rapid pulse ').
human_symptom(nauseous , 'that you are nauseous ').

human_illness(blunt_force_injury , 'a blunt force injury(seek medical attention if conditions worsen)').
human_illness(severe_blunt_force_injury , 'a severe blunt force injury(advised to seek medical attention)').
human_illness(fracture , 'a fracture(seek medical attention immediately, isolated affected limb and restrict movement)').
human_illness(sprain , 'a sprain(ice and rest affected area, seek medical attention if conditions do not improve)').
human_illness(dengue_fever , 'dengue fever(seek medical attention immediately)').
human_illness(food_poisoning , 'food poisoning(rest and hydrate well, seek medical attention if conditions worsen)').
human_illness(cold , 'a cold(stay at home and avoid going to crowded areas in light of covid-19, seek medical attention if high fever or symptoms worsen)').
human_illness(flu , 'a flu(stay at home and avoid going to crowded areas in light of covid-19, seek medical attention if high fever or symptoms worsen)').
human_illness(severe_flu , 'a severe flu(seek medical attention immediately and avoid going to crowded areas in light of covid-19)').
human_illness(viral_fever , 'a viral fever(seek medical attention immediately and avoid going to crowded areas in light of covid-19)').
human_illness(headache , 'a headache(rest well, if no improvements go see a doctor)').
human_illness(migraine , 'a migraine(rest well, if no improvements go see a doctor)').
```

The **human_symptom** and **human_illness** predicates stores the sentence string for each of the symptoms and illnesses in the Prolog Knowledge Base to be written out to the console for the user to see and understand in a more human readable form.

The **human_symptom** predicate stores the symptoms to be asked to the user and the sentence string allows the user to better understand the symptoms to be able to answer the questions more accurately. The **human_illness** predicate stores the possible illnesses sentence strings in the Prolog knowledge base and when a diagnosis is made, the diagnosed result/results are written to the console in human readable form based on the **human_illness** predicate sentence strings as well as the different advise for the various illnesses. For example, due to the Covid-19 pandemic, if the illness diagnosed is **viral_fever**, the user is advised to seek medical attention immediately and stay away from crowded areas to prevent the possible spread of Covid-19.

openings and opening Predicates Functionality

```
openings(no_pain, 'I will take note of that, ').
openings(mild_pain, 'I will take note of that, ').
openings(moderate_pain, 'I will take note of that, ').
openings(no_pain, 'I see,if thats the case, ').
openings(mild_pain, 'I see,if thats the case, ').
openings(moderate_pain, 'I see,if thats the case, ').
openings(excruciating_pain, 'Alright,just bear with me a little longer, ').
openings(severe_pain, 'Alright,just bear with me a little longer, ').
openings(excruciating_pain, 'I can see that you are in pain,but to help me diagnose you better, ').
openings(severe_pain, 'I can see that you are in pain,but to help me diagnose you better, ').
openings(excruciating_pain, 'Poor thing,bear with the pain just a little longer, ').
openings(severe_pain, 'Poor thing,bear with the pain just a little longer, ').
openings(excruciating_pain, 'My dear,we will help you ease the pain soon, ').
openings(severe_pain, 'My dear,we will help you ease the pain soon, ').

openings(angry, 'I know that you are feeling angry,take a deep breath and stay calm, I will get your diagnosis in no time, ').
openings(angry, 'I can see that you are angry,take a deep breath and stay calm, I will get your diagnosis in no time, ').
openings(angry, 'I can see your visible anger,just bear with me a moment, ').
openings(upset, 'My dear,I can see that you are visibly upset,I will get you better soon, ').
openings(frustrated, 'My dear,I understand your frustration,I am here for you, ').
openings(frustrated, 'My dear,I see your frustration,I am here for you, ').
openings(frustrated, 'My dear,I understand your frustration,we will get you better soon, ').

% get a random one from openings
opening(OP):- (
    current_predicate(pain/1)->pain(Q),findall(A, openings(Q,A), PainList),
    current_predicate(mood/1)->mood(Y),findall(A, openings(Y,A), MoodList),
    append(PainList,MoodList,OpeningsList)
),!,
    random_member(OP, OpeningsList).
opening(OP):- (
    current_predicate(pain/1)->pain(Q),findall(A, openings(Q,A), OpeningsList)
),!,
    random_member(OP, OpeningsList).
opening(OP):-
    findall(A, openings(no_pain,A), OpeningsList),
    random_member(OP, OpeningsList).
```

The **openings** predicate stores the sentence string for each of the possible openings for the questions asked by Dr Bot for the Sympathetic Doctor interactive Talking Box. In the scope, we mentioned that the interactive Talking Box should be able to diagnose the patient's condition while asking questions sensitively depending upon patient's pain level and mood level. Therefore, each **openings** predicate stores the sentence string of an opening to a question for a particular pain level or mood, and the Prolog script will choose one of sentence string corresponding to the patient's mood or pain level.

The 3 **opening** predicates are used to decide the **openings** predicate to use, corresponding to the patient's mood or pain level. For the first **opening** predicate, the predicate checks if there is a pain level and mood answered by the user. If true, the **findall()** predicate is used to find all possible openings that corresponds to the patient's mood or pain level and append them into one list, and then the **random_member** predicate is called to randomly choose one of the possible openings to be used for the question. For the second **opening** predicate, the predicate checks if there is a pain level answered by the user. If true, the **findall()** predicate is used to find all possible openings that corresponds to the patient's pain level and put them into one list, and then the **random_member** predicate is called to randomly choose one of the possible openings to be used for the question. For the last **opening** predicate, if both pain level and mood are not answered by the user, we use the **findall()** predicate to get a list of openings that corresponds to the **no_pain** pain level and use the **random_member** predicate is called to randomly choose one of the possible openings to be used for the question. Therefore, by having 3 **opening** predicates are used to decide the **openings** predicate to use, corresponding to the patient's mood or pain level, we achieve the criteria of being able to diagnose the patient's condition while asking questions sensitively depending upon patient's pain level and mood level.

```

Hi, I am Dr Bot, let me help diagnose you today!
To help me begin,do you feel no pain ?
Respond with yes or no (or press q to abort):no.
Ok understood,in that case, do you feel excruciating pain ?
Respond with yes or no (or press q to abort): |: yes.
Alright,just bear with me a little longer, do you feel frustrated ?
Respond with yes or no (or press q to abort): |: yes.
My dear,I understand your frustration,I am here for you, do you feel that you have a weak, rapid pulse ?
Respond with yes or no (or press q to abort): |: yes.
My dear,I understand your frustration,I am here for you, do you feel that you have a cough in the past few days ?
Respond with yes or no (or press q to abort): |: no.
I can see that you are in pain,but to help me diagnose you better, do you feel that there is an open wound on skin ?
Respond with yes or no (or press q to abort): |: no.
My dear,we will help you ease the pain soon, do you feel that there is a red welt at the pain area(if no open wound) ?

```

In the example above, as the user has answered that he is experiencing excruciating pain as well as feeling frustrated, therefore, the openings used subsequently helps to sensitively address the user's pain as well as his frustration.

question_starts, question_start and human_diagnose Predicates Functionality

```

% asking part of a interrogative sentence
question_starts('do you feel ').
question_start(QS):-question_starts(QS).

% map diagnosis result list to human string
human_diagnose(L, H):-
    length(L, Len),
    (
        Len==0 -> human_illness(no_illness,H);
        (
            (Len>4 -> nl, write('there is not enough symptoms to make accurate          diagnosis,please try to be more specific, ')
;true),
            convlist([X,Y]>>human_illness(X,Y), L, HL),
            atomic_list_concat(HL, ',OR,', H)
        )
    ).

```

The **question_start** predicate chooses one of the possible starts to a question from the possible **question_starts** predicates after the opening of the question which is decided by the **opening** predicate. In our case, there is only one possible **question_starts** predicate to choose from. Next, the **human_diagnose** predicate is used to concatenate all the possible diagnoses obtained from the **diagnose()** predicate into a string to be written to the console to tell the user the possible diagnoses the Sympathetic Doctor interactive Talking Box has derived.

pain_level, mood_level, symptom_library and illness Predicates Functionality

```
% Set of pains and moods
pain_level([excruciating_pain, severe_pain, moderate_pain, mild_pain, no_pain]).
mood_level([angry, upset, frustrated, fatigued, calm]).
symptom_library([physical_pain, insect, swollen, stinging, sting_area, welt, extreme_redness, numb, difficulty_breathing, skin_reactions, tongue_swell, weak_pulse, nauseous, diarrhea, dizzy, redness, bumps, dry_skin, blister, itching, rash, runny_nose, red_eyes, stomach_ache, little_to_no_blood, moderate_blood, alot_blood, open_wound, no_open_wound, weakness, chills, sweat, limited_mobility, bone, exercise, high_temperature, mosquito, head, tightness_head, body_ache, fever, mild_fever, behind_eyes, sneeze, cough, sore_throat, chest_pain, tenderness_head, fatigue, dehydration, sensitive, heavy_sweat, cramps, bruising]).

illness(allergic_bee_sting, [moderate_pain, severe_pain, excruciating_pain, physical_pain, insect, swollen, stinging, sting_area, welt, extreme_redness, numb, fatigue, little_to_no_blood, head, tightness_head, itching, skin_reactions, difficulty_breathing, tongue_swell, weak_pulse, nauseous, diarrhea, dizzy, redness, no_open_wound, weakness, sweat, rash, chills]).

illness(moderate_bee_sting, [moderate_pain, severe_pain, excruciating_pain, physical_pain, insect, swollen, stinging, sting_area, welt, extreme_redness, redness, numb, little_to_no_blood, no_open_wound]).

illness(mild_bee_sting, [mild_pain, physical_pain, insect, swollen, stinging, sting_area, welt, numb, extreme_redness, redness, little_to_no_blood, no_open_wound]).

illness(allergic_reaction, [moderate_pain, severe_pain, excruciating_pain, physical_pain, runny_nose, red_eyes, difficulty_breathing, skin_reactions, tongue_swell, stomach_ache, diarrhea, weak_pulse, fatigue, head, tightness_head, tenderness_head, itching, no_open_wound, little_to_no_blood, rash, nauseous, chills, sweat]).

illness(eczema, [no_pain, itching, redness, dry_skin, bumps, rash, blister, little_to_no_blood]).

illness(cut, [mild_pain, physical_pain, little_to_no_blood, stinging, open_wound]).

illness(moderate_cut, [moderate_pain, physical_pain, moderate_blood, stinging, open_wound]).

illness(moderate_cut, [severe_pain, physical_pain, moderate_blood, stinging, open_wound]).

illness(deep_cut, [moderate_pain, physical_pain, alot_blood, stinging, open_wound, weakness, chills, dizzy, sweat, weak_pulse]).

illness(deep_cut, [severe_pain, physical_pain, alot_blood, stinging, open_wound, weakness, chills, dizzy, sweat, weak_pulse]).

illness(deep_cut, [excruciating_pain, physical_pain, alot_blood, stinging, open_wound, weakness, chills, dizzy, sweat, weak_pulse]).

illness(injury, [mild_pain, physical_pain, little_to_no_blood, open_wound, swollen, numb]).
```

The **pain_level** predicate stores the list of the 5 levels of pain, and the **mood_level** predicate stores the 5 mood considerations that are available in the Prolog Knowledge-Based System (KBS). The **symptom_library** predicate then stores all the defined symptoms in the Prolog Knowledge-Based System (KBS).

Next, we have the **illness** predicates. Each of the illness predicate stores an illness that can be diagnosed by the Sympathetic Doctor interactive Talking Box, followed by the list of symptoms that the illness presents. Notice that some of the illnesses are repeated but given different pain level, the reason for that is that different users have different pain tolerance and concept of pains, therefore, for some illnesses, there are a few predicates, but each with a different pain level. For example, for the deep_cut **illness** predicate, there are 3 predicates with 3 different pain levels, moderate_pain, severe_pain, and excruciating_pain, this is due to the fact that even though the user is experiencing a deep cut wound, he may have a higher pain tolerance and think that he has a moderate amount of pain, whereas a different user may have a lower pain tolerance and think that he has an excruciating amount of pain. We also did not use mood as a symptom for the various illnesses, this is given the fact that different users may have different moods with regards to the illnesses and judging if a user has an illness by his mood may not give an accurate diagnosis.

reply_question, reply_diagnose and response Predicates

Functionality

```
%%function to ask user questions
reply_question:-
    opening(OP),                % get Opening
    question_start(QS),         % get question start
    nextQuestion(Question),
    human_symptom(Question, Human_question), % get next question
    write(OP),write(QS), write(Human_question),write('?'),nl,write('Respond with yes or no (or press q to abort): '),read(Response),response(Question,
Response),!.

%% reply the diagnosis result page
reply_diagnose:-
    nl,write('With the answers you have provided me, '),
    diagnose(X),                % get Opening
    sort(X, Result),
    human_diagnose(Result, Human_result), % get diagnosis Result
    write('you might have: '), write(Human_result),write('.'),nl,retractall(answered(_)),retractall(have_symptom(_)),retractall(ready_to_diagnose(_)),
    retractall(pain(_)),retractall(mood(_)),ask(0),!.

%%handle user response
response(Q, A):-
    answer(Q,A),
    (
        check_diagnose -> reply_diagnose;reply_question
    ).
```

Finally, we come to the last 3 predicates that ensure that the Sympathetic Doctor interactive Talking Box Prolog script is able to run continuously and ask the user questions.

The **reply_question** predicate is used to write the question onto the Prolog console for the user to view and reply to. The **reply_question** predicate first calls the **opening**, **question_start** and **nextQuestion** predicates to retrieve the opening, the question header as well as the question to be asked to the user from the Prolog knowledge base. Next, because the **nextQuestion** predicate returns the symptom to be asked in a not so human readable form, we use the **human_symptom** predicate to get the human readable sentence string to be asked to the user. Then, finally, we use the **write()** predicate to write the question to the user on the Prolog console. When the user provides a response in the form of ‘yes.’ or ‘no.’, we will read the response given by using the **read()** predicate and then pass the response and the corresponding question into the **response** predicate to be handled.

The **response** predicate first calls the **answer** predicate to log the user’s response into the Prolog knowledge base. Then, the **response** predicate calls the **check_diagnose** predicate to check if the Sympathetic Doctor interactive Talking Box Prolog script is ready to make a diagnosis based on the symptoms provided by the user so far, if yes, we call the **reply_diagnose** predicate to write the diagnosis to the user, if not, the **reply_question** predicate is called to ask the user another question. The **response** predicate iteratively calls the **reply_question** predicate is called to ask the user questions until the Sympathetic Doctor interactive Talking Box Prolog script is ready to make a diagnosis which makes the script runs automatically and iteratively.

The **reply_diagnose** predicate is used to write the diagnosis derived by the Sympathetic Doctor interactive Talking Box Prolog script onto the Prolog console for the user. The predicate first retrieves the possible illnesses diagnosed by the Prolog script by calling the **diagnose** predicate. Next, the sort predicate is used to sort the list of diagnosed illnesses and remove any duplicate diagnosis, we then call the **human_diagnose** predicate to retrieve the human readable sentence string of the concatenated list of diagnosed illnesses which is then written onto the Prolog console.

Also, to ensure that the Prolog script is able to run if the user does not enter ‘q’ to quit the Sympathetic Doctor interactive Talking Box Prolog script, we first use the **retractall()** predicate to clear the Prolog knowledge base of the dynamically declared predicates that we have asserted to ensure that the questioning by the Sympathetic Doctor interactive Talking Box starts from scratch. The **ask(0)** predicate is called to allow the Sympathetic Doctor interactive Talking Box to run again.

Demonstration of the Sympathetic Doctor interactive Talking Box

Initial Start Up

SWI-Prolog -- c:/Users/Shearman Chua/Desktop/Com Sci Courses/Year 3 Semester 1/CZ3005-Artificial Intelligence/Labs/Lab 4/Dr Bot - final.pl

File Edit Settings Run Debug Help

```
Hi, I am Dr Bot, let me help diagnose you today!  
To help me begin,do you feel mild pain ?  
Respond with yes or no (or press q to abort):■
```

When the Sympathetic Doctor interactive Talking Box is initially run by double clicking on the Prolog file, the Prolog console is opened up and the user is greeted by Dr Bot, which tells the user he will be diagnosing him today. Then the script begins by asking the user the pain level he is experiencing. The user is required to answer ‘yes.’ or ‘no.’ to the question asked.

After Answering Pain Level

SWI-Prolog -- c:/Users/Shearman Chua/Desktop/Com Sci Courses/Year 3 Semester 1/CZ3005-Artificial Intelligence/Labs/Lab 4/Dr Bot - final.pl

File Edit Settings Run Debug Help

```
Hi, I am Dr Bot, let me help diagnose you today!  
To help me begin,do you feel mild pain ?  
Respond with yes or no (or press q to abort):no.  
I will take note of that, do you feel severe pain ?  
Respond with yes or no (or press q to abort): |: no.  
I see,if thats the case, do you feel moderate pain ?  
Respond with yes or no (or press q to abort): |: yes.  
I will take note of that, do you feel upset ?  
Respond with yes or no (or press q to abort): |: ■
```

After the user has provided Dr Bot with an answer ‘yes.’ to the pain level he is experiencing, or that all 5 pain levels are asked, he then proceed to ask about the user’s mood. The user is required to answer ‘yes.’ or ‘no.’ to the question asked.

After Answering Mood Consideration

SWI-Prolog -- c:/Users/Shearman Chua/Desktop/Com Sci Courses/Year 3 Semester 1/CZ3005-Artificial Intelligence/Labs/Lab 4/Dr Bot - final.pl

File Edit Settings Run Debug Help

```
Hi, I am Dr Bot, let me help diagnose you today!  
To help me begin,do you feel mild pain ?  
Respond with yes or no (or press q to abort):no.  
I will take note of that, do you feel severe pain ?  
Respond with yes or no (or press q to abort): |: no.  
I see,if thats the case, do you feel moderate pain ?  
Respond with yes or no (or press q to abort): |: yes.  
I will take note of that, do you feel upset ?  
Respond with yes or no (or press q to abort): |: yes.  
My dear,I can see that you are visibly upset,I will get you better soon, do you feel that you are having persistent pain or pressure in the chest or abdomen ?  
Respond with yes or no (or press q to abort): |:
```

After the user has provided Dr Bot with an answer ‘yes.’ to the mood that he is experiencing, or that all 5 mood considerations available are asked, he then proceed to ask about the user’s symptoms. The user is required to answer ‘yes.’ or ‘no.’ to the question asked. Notice that once the user provided his pain level and mood, the opening to the question Dr Bot ask are sympathetic to the mood and pain level of the user. In the example above, Dr Bot knows that the user is ‘upset’ and addresses his mood in the opening to his question.

Diagnosis

```
SWI-Prolog -- c:/Users/Shearman Chua/Desktop/Com Sci Courses/Year 3 Semester 1/CZ3005-Artificial Intelligence/Labs/Lab 4/Dr Bot - final.pl
File Edit Settings Run Debug Help
Respond with yes or no (or press q to abort): |: no.
My dear,I can see that you are visibly upset,I will get you better soon, do you feel that you are having tenderness on your scalp, neck and
shoulder muscles (with no fever, mild fever) ?
Respond with yes or no (or press q to abort): |: no.
Ok understood,in that case, do you feel that you have diarrhea in the past few days ?
Respond with yes or no (or press q to abort): |: no.
My dear,I can see that you are visibly upset,I will get you better soon, do you feel that you are having a sore throat ?
Respond with yes or no (or press q to abort): |: yes.
I see,if thats the case, do you feel that you have a stomach ache ?
Respond with yes or no (or press q to abort): |: no.
I see,if thats the case, do you feel there are small bumps on affected area ?
Respond with yes or no (or press q to abort): |: no.
I will take note of that, do you feel that you have localized bruising on the skin ?
Respond with yes or no (or press q to abort): |: no.
I will take note of that, do you feel that you have been exercising strenuously ?
Respond with yes or no (or press q to abort): |: no.
Ok understood,in that case, do you feel that you are fatigued ?
Respond with yes or no (or press q to abort): |: yes.
I will take note of that, do you feel that you are dehydrated ?
Respond with yes or no (or press q to abort): |: no.
My dear,I can see that you are visibly upset,I will get you better soon, do you feel that you are dizzy ?
Respond with yes or no (or press q to abort): |: yes.
I will take note of that, do you feel that you have pain behind the eyes ?
Respond with yes or no (or press q to abort): |: no.
Ok understood,in that case, do you feel that you have been sneezing frequently as of recent ?
Respond with yes or no (or press q to abort): |: yes.
Ok understood,in that case, do you feel that you are sensitive to light, sound, or smell ?
Respond with yes or no (or press q to abort): |: no.
I see,if thats the case, do you feel that you are nauseous ?
Respond with yes or no (or press q to abort): |: no.
I will take note of that, do you feel that you have been bitten by a mosquito recently or live in a mosquito infested area ?
Respond with yes or no (or press q to abort): |: no.
My dear,I can see that you are visibly upset,I will get you better soon, do you feel that there is NO open wound on skin(reply yes only if
you have localized external physical pain, excluding headaches and full body aches) ?
Respond with yes or no (or press q to abort): |: no.

With the answers you have provided me,you might have: a cold(stay at home and avoid going to crowded areas in light of covid-19, seek medic
al attention if high fever or symptoms worsen).OR,a flu(stay at home and avoid going to crowded areas in light of covid-19, seek medical at
tention if high fever or symptoms worsen).OR,a severe flu(seek medical attention immediately and avoid going to crowded areas in light of c
ovid-19).OR,a viral fever(seek medical attention immediately and avoid going to crowded areas in light of covid-19).

Hi, I am Dr Bot, let me help diagnose you today!
To help me begin,do you feel mild pain ?
Respond with yes or no (or press q to abort):|:
```

After asking the user questions, when **the ready_to_diagnose** predicate is set, the Sympathetic Doctor interactive Talking Box displays the diagnosis results on the Prolog console. In the example above, as the user is presenting flu like symptoms, those flu-related illnesses are diagnosed to the user. This is due to the fact that our Sympathetic Doctor interactive Talking Box implementation uses the elimination method to reduce the possible illnesses that the user may have to a small set and presents to the user the possible illnesses the user may have. After the diagnosis results are presented to the user, the script continues to run again and the user can get another diagnosis.

```
Respond with yes or no (or press q to abort): |: no.
I will take note of that, do you feel that you have pain behind the eyes ?
Respond with yes or no (or press q to abort): |: no.

With the answers you have provided me,
there is not enough symptoms to make accurate diagnosis,please try to be more specific, you might have: an allergic reaction t
o bee sting (seek medical attention immediately).OR,an allergic reaction (seek medical attention immediately).OR,a blunt force injury(seek
medical attention if conditions worsen).OR,a cold(stay at home and avoid going to crowded areas in light of covid-19, seek medical attentio
n if high fever or symptoms worsen).OR,a small cut, you will be fine(do get some medication for it).OR,a deep cut(seek medical attention im
mediately).OR,dengue fever(seek medical attention immediately).OR,an eczema(do get some medication for it and do not scratch affected area,
seek medical attention if no improvements or if symptoms get worse).OR,a flu(stay at home and avoid going to crowded areas in light of cov
id-19, seek medical attention if high fever or symptoms worsen).OR,food poisoning(rest and hydrate well, seek medical attention if conditio
ns worsen).OR,a fracture(seek medical attention immediately, isolated affected limb and restrict movement).OR,a headache(rest well, if no i
mprovements go see a doctor).OR,heat exhaustion(remove excess clothings, hydrate adequately, rest in shaded area, if no improvements go see
a doctor).OR,an injury(if symptoms worsen, get medical attention).OR,a migraine(rest well, if no improvements go see a doctor).OR,a mild b
ee sting (do get some medication for it, seek medical attention if no improvements or if symptoms get worse).OR,a moderate reaction to bee
sting (do get some medication for it, seek medical attention if no improvements or if symptoms get worse).OR,a moderate cut(disinfect wound
and put pressure and bandage the affected area, get medical attention if bleeding do not stop).OR,a severe blunt force injury(advised to s
eek medical attention).OR,a severe flu(seek medical attention immediately and avoid going to crowded areas in light of covid-19).OR,a sever
e injury(seek medical attention immediately).OR,a sprain(ice and rest affected area, seek medical attention if conditions do not improve).O
R,a viral fever(seek medical attention immediately and avoid going to crowded areas in light of covid-19).
```

In the example above, if we were to answer ‘no.’ to all the questions asked by the Prolog script, we will get the message “there is not enough symptoms to make accurate diagnosis, please try to be more specific” and all the possible illnesses in the Prolog knowledge base are presented to the user.

```

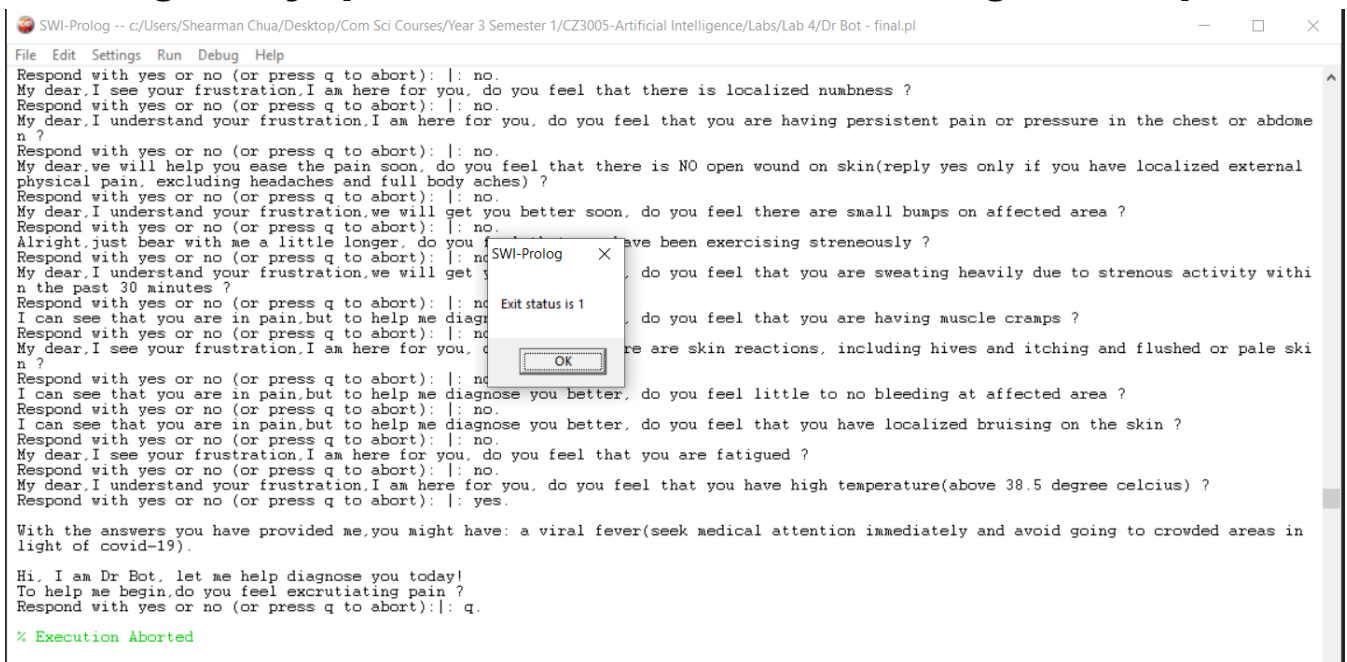
..
Respond with yes or no (or press q to abort): |: no.
I can see that you are in pain,but to help me diagnose you better, do you feel little to no bleeding at affected area ?
Respond with yes or no (or press q to abort): |: no.
I can see that you are in pain,but to help me diagnose you better, do you feel that you have localized bruising on the skin ?
Respond with yes or no (or press q to abort): |: no.
My dear,I see your frustration,I am here for you, do you feel that you are fatigued ?
Respond with yes or no (or press q to abort): |: no.
My dear,I understand your frustration,I am here for you, do you feel that you have high temperature(above 38.5 degree celcius) ?
Respond with yes or no (or press q to abort): |: yes.

With the answers you have provided me,you might have: a viral fever(seek medical attention immediately and avoid going to crowded areas in
light of covid-19).

```

In the example above, as the user has accurately described his symptoms, the Sympathetic Doctor interactive Talking Box is able to give a single illness diagnosis to the user. Therefore, the more specific the symptoms the user presents, the lesser illnesses are being diagnosed to the user by the Sympathetic Doctor.

Exiting the Sympathetic Doctor interactive Talking Box Script



To quit the Sympathetic Doctor interactive Talking Box Prolog script, the user just has to enter 'q.' to the Prolog console and the Prolog script will be exited.

Conclusions

In conclusion, in view of the Covid-19 pandemic, it is useful to have the Sympathetic Doctor interactive Talking Box Prolog script which can thoroughly ask patients questions and from their answers, make an initial diagnosis on what illnesses or diseases they may be having. This allows patients who do not have flu-like symptoms to avoid going to visit the doctor unnecessarily, as well as advise patients to seek medical attention if they have flu-like symptoms.

Also, by using openings that are sensitive to the patient's pain level as well as mood before asking the questions, the patients will feel more assured and feel the human touch to the Sympathetic Doctor interactive Talking Box.

To summarize, the Sympathetic Doctor interactive Talking Box Prolog script is a very useful Knowledge-Based System (KBS) that is able to provide effective diagnosis to patients, based on the symptoms that they present for common illnesses that can be diagnosed by the Sympathetic Doctor interactive Talking Box.