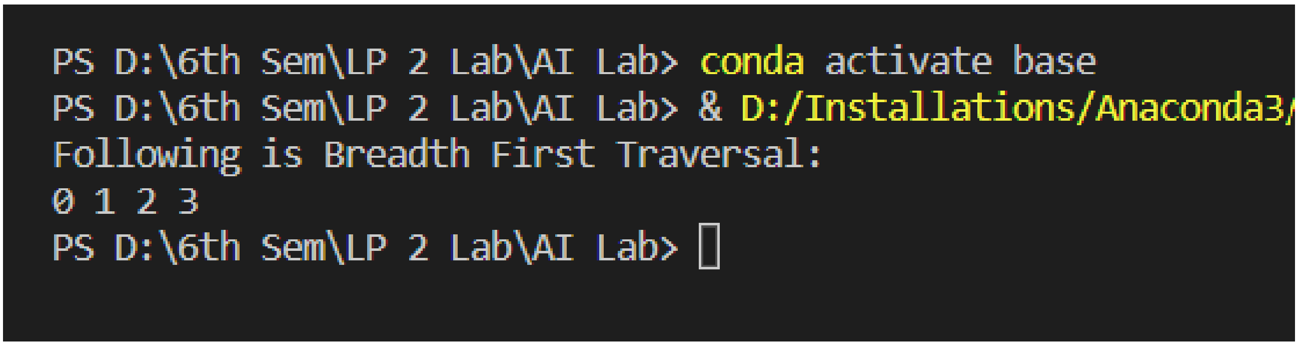
AI – 1.1 BFS Undirected Graph

Code

```
# BFS algorithm in Python
import collections
# BFS algorithm

def bfs(graph, root):
    visited, queue = set(), collections.deque([root])
    visited.add(root)

    while queue:
        # Dequeue a vertex from queue
        vertex = queue.popleft()
        print(str(vertex) + " ", end="")
        # If not visited, mark it as visited, and
        # enqueue it
        for neighbour in graph[vertex]:
            if neighbour not in visited:
                visited.add(neighbour)
                queue.append(neighbour)

if __name__ == '__main__':
    graph = {0: [1, 2], 1: [2], 2: [3], 3: [1, 2]}
    print("Following is Breadth First Traversal: ")
    bfs(graph, 0)
```

Output



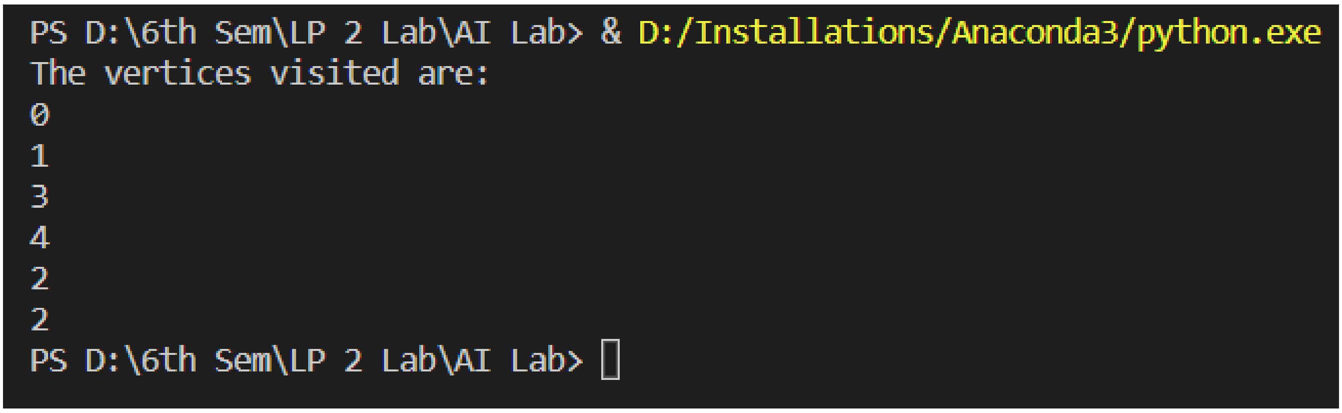
```
PS D:\6th Sem\LP 2 Lab\AI Lab> conda activate base
PS D:\6th Sem\LP 2 Lab\AI Lab> & D:/Installations/Anaconda3/
Following is Breadth First Traversal:
0 1 2 3
PS D:\6th Sem\LP 2 Lab\AI Lab>
```

AI- 1.2 DFS Undirected Graph

Code

```
# DFS algorithm in Python
# DFS algorithm
def dfs(graph, start, visited=None):
    if visited is None:
        visited = set()
    visited.add(start)
    print(start)

    for next in graph[start] - visited:
        dfs(graph, next, visited)
    return visited

graph = {'0': set(['1', '2']),
         '1': set(['0', '3', '4']),
         '2': set(['0']),
         '3': set(['1']),
         '4': set(['2', '3'])}
dfs(graph, '0')
```

Output



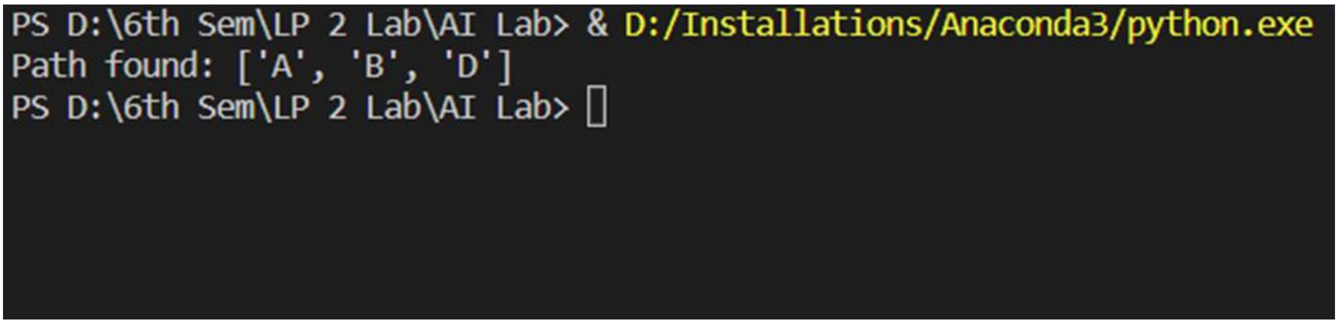
```
PS D:\6th Sem\LP 2 Lab\AI Lab> & D:/Installations/Anaconda3/python.exe
The vertices visited are:
0
1
3
4
2
2
PS D:\6th Sem\LP 2 Lab\AI Lab> 
```

AI- 2 A Star algorithm

Code

```
from collections import deque
class Graph:
    # example of adjacency list (or rather map)
    def __init__(self, adjacency_list):
        self.adjacency_list = adjacency_list

    def get_neighbors(self, v):
        return self.adjacency_list[v]

    # heuristic function with equal values for all nodes
    def h(self, n):
        H = {
            'A': 1,
            'B': 1,
            'C': 1,
            'D': 1
        }
        return H[n]

    def a_star_algorithm(self, start_node, stop_node):
 # open_list is a list of nodes which have been visited, but who's
neighbors haven't all been inspected, starts off with the start node
# closed_list is a list of nodes which have been visited and who's
neighbors have been inspected
        open_list = set([start_node])
        closed_list = set([])


  # g contains current distances from start_node to all other nodes the
default value (if it's not found in the map) is +infinity
        g = {}
        g[start_node] = 0
        # parents contains an adjacency map of all nodes
        parents = {}
        parents[start_node] = start_node
        while len(open_list) > 0:
            n = None
   # find a node with the lowest value of f() - evaluation function
            for v in open_list:
                if n == None or g[v] + self.h(v) < g[n] + self.h(n):
                    n = v;

            if n == None:
                print('Path does not exist!')
                return None

  # if the current node is the stop_node
  # then we begin reconstructin the path from it to the start_node
            if n == stop_node:
                reconst_path = []
                while parents[n] != n:
                    reconst_path.append(n)
                    n = parents[n]
                reconst_path.append(start_node)
                reconst_path.reverse()
                print('Path found: {}'.format(reconst_path))
                return reconst_path

            # for all neighbors of the current node do
            for (m, weight) in self.get_neighbors(n):
     # if the current node isn't in both open_list and closed_list
# add it to open_list and note n as it's parentif m not in open_list and m
not in closed_list:
                    open_list.add(m)
                    parents[m] = n
                    g[m] = g[n] + weight
                # otherwise, check if it's quicker to first visit n, then
m and if it is, update parent data and g data and if the node was in the
closed_list, move it to open_list
                else:
                    if g[m] > g[n] + weight:
                        g[m] = g[n] + weight
                        parents[m] = n

                        if m in closed_list:
                            closed_list.remove(m)
                            open_list.add(m)

            # remove n from the open_list, and add it to closed_list
because all of his neighbors were inspected
            open_list.remove(n)
            closed_list.add(n)
        print('Path does not exist!')
        return None

adjacency_list = {
'A': [('B', 1), ('C', 3), ('D', 7)],
'B': [('D', 5)],
'C': [('D', 12)]
}
graph1 = Graph(adjacency_list)
graph1.a_star_algorithm('A','D')
```

Output



```
PS D:\6th Sem\LP 2 Lab\AI Lab> & D:/Installations/Anaconda3/python.exe
Path found: ['A', 'B', 'D']
PS D:\6th Sem\LP 2 Lab\AI Lab> 
```

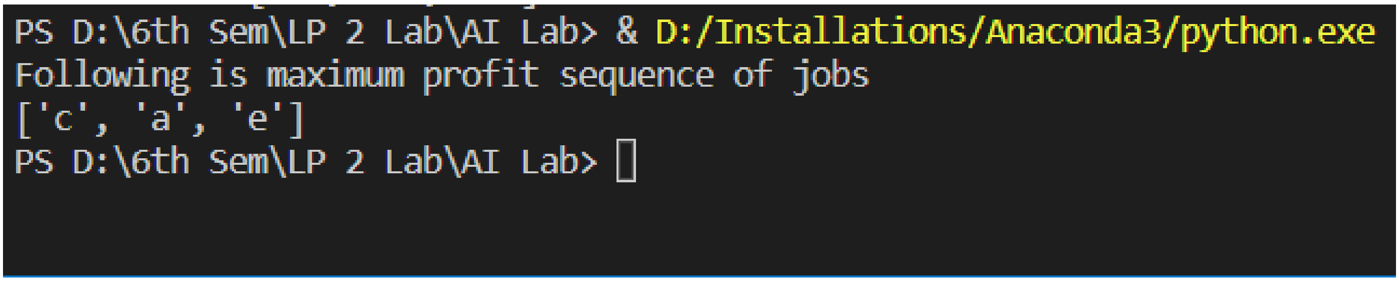
AI- 3 Greedy Search Algorithm- Job Scheduling Problem

Code

```
# Program to find the maximum profit
# job sequence from a given array
# of jobs with deadlines and profit
# function to schedule the jobs take 2
# arguments array and no of jobs to schedule
def printJobScheduling(arr, t):

     # length of array
     n = len(arr)

     # Sort all jobs according to
     # decreasing order of profit
     for i in range(n):
          for j in range(n - 1 - i):
               if arr[j][2] < arr[j + 1][2]:
                    arr[j], arr[j + 1] = arr[j + 1], arr[j]

     # To keep track of free time slots
     result = [False] * t

     # To store result (Sequence of jobs)
     job = ['-1'] * t

     # Iterate through all given jobs
     for i in range(len(arr)):
          # Find a free slot for this job
          # (Note that we start from the
          # last possible slot)
          for j in range(min(t - 1, arr[i][1] - 1), -1, -1):

               # Free slot found
```

```
            if result[j] is False:
                result[j] = True
                job[j] = arr[i][0]
                break
    # print the sequence
    print(job)

# Driver COde
arr = [['a', 2, 100], # Job Array
    ['b', 1, 19],
    ['c', 2, 27],
    ['d', 1, 25],
    ['e', 3, 15]]
print("Following is maximum profit sequence of jobs")
# Function Call
printJobScheduling(arr, 3)
```

Output



```
PS D:\6th Sem\LP 2 Lab\AI Lab> & D:/Installations/Anaconda3/python.exe
Following is maximum profit sequence of jobs
['c', 'a', 'e']
PS D:\6th Sem\LP 2 Lab\AI Lab> 
```

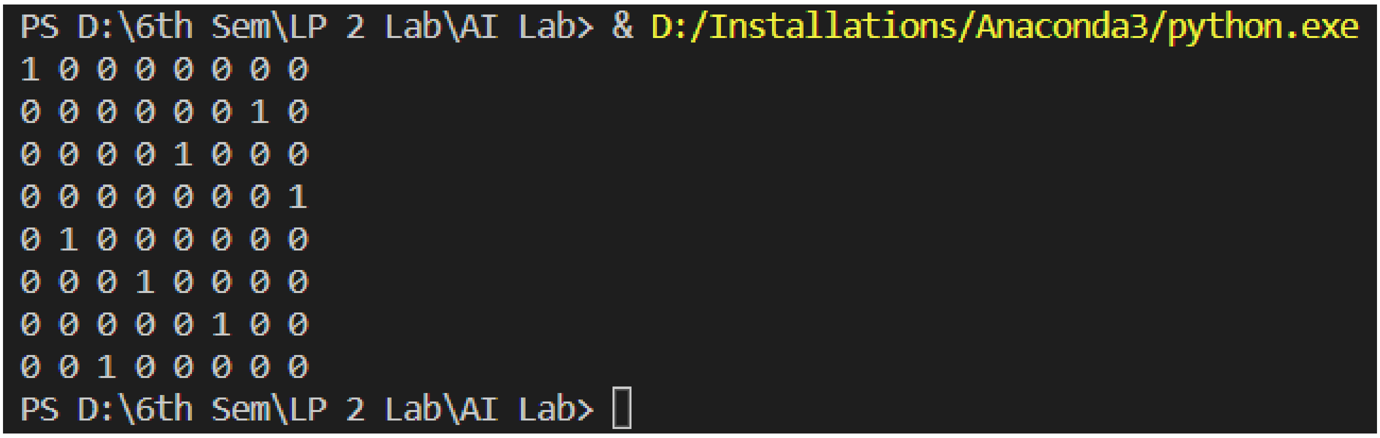
AI- 4. N-queens problem

Code

```
import io

import random

import string

import warnings

import numpy as np

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine_similarity

import warnings

warnings.filterwarnings('ignore')

import nltk

from nltk.stem import WordNetLemmatizer

# nltk.download('popular', quiet=True)

# nltk.download('punkt')

# nltk.download('wordnet')


with open('chatbot.txt','r', encoding='utf8', errors ='ignore') as fin:

    raw = fin.read().lower()


#Tokenisation

sent_tokens = nltk.sent_tokenize(raw)

word_tokens = nltk.word_tokenize(raw)


# Preprocessing

lemmer = WordNetLemmatizer()

def LemTokens(tokens):

    return [lemmer.lemmatize(token) for token in tokens]

remove_punct_dict = dict((ord(punct), None) for punct in
string.punctuation)

def LemNormalize(text):

    return
LemTokens(nltk.word_tokenize(text.lower().translate(remove_punct_dict)))
```

```
# Keyword Matching
GREETING_INPUTS = ("hello", "hi", "greetings", "sup", "what's up","hey","Helo")
GREETING_RESPONSES = ["hi", "hey", "hi there", "hello", "I am glad! You are talking to me"]

def greeting(sentence):
    for word in sentence.split():
        if word.lower() in GREETING_INPUTS:
            return random.choice(GREETING_RESPONSES)

def response(user_response):
    robo_response=''
    sent_tokens.append(user_response)
    TfidfVec = TfidfVectorizer(tokenizer=LemNormalize, stop_words='english')
    tfidf = TfidfVec.fit_transform(sent_tokens)
    vals = cosine_similarity(tfidf[-1], tfidf)
    idx=vals.argsort()[0][-2]
    flat = vals.flatten()
    flat.sort()
    req_tfidf = flat[-2]
    if(req_tfidf==0):
        robo_response=robo_response+"I am sorry! I don't understand you"
        return robo_response
    else:
        robo_response = robo_response+sent_tokens[idx]
        return robo_response

flag=True
print("ROBO: My name is Robo. I will answer your queries about Investments. If you want to exit, type Bye!")
while(flag==True):
    user_response = input()
    user_response=user_response.lower()
    if(user_response!='bye'):
        if(user_response=='thanks' or user_response=='thank you' ):
            flag=False
            print("ROBO: You are welcome..")
        else:
            if(greeting(user_response)!=None):
                print("ROBO: "+greeting(user_response))
            else:
                print("ROBO: ",end="")
                res = response(user_response)
                nlines = res.count('\n')
                if nlines > 0:
                    res = res.split("\n",1)[1]
                print(res)
                sent_tokens.remove(user_response)
    else:
        flag=False
        print("ROBO: Bye! take care..")
```

Output



```
PS D:\6th Sem\LP 2 Lab\AI Lab> & D:/Installations/Anaconda3/python.exe
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0
PS D:\6th Sem\LP 2 Lab\AI Lab> 
```

AI-5 Chatbot Application in Python

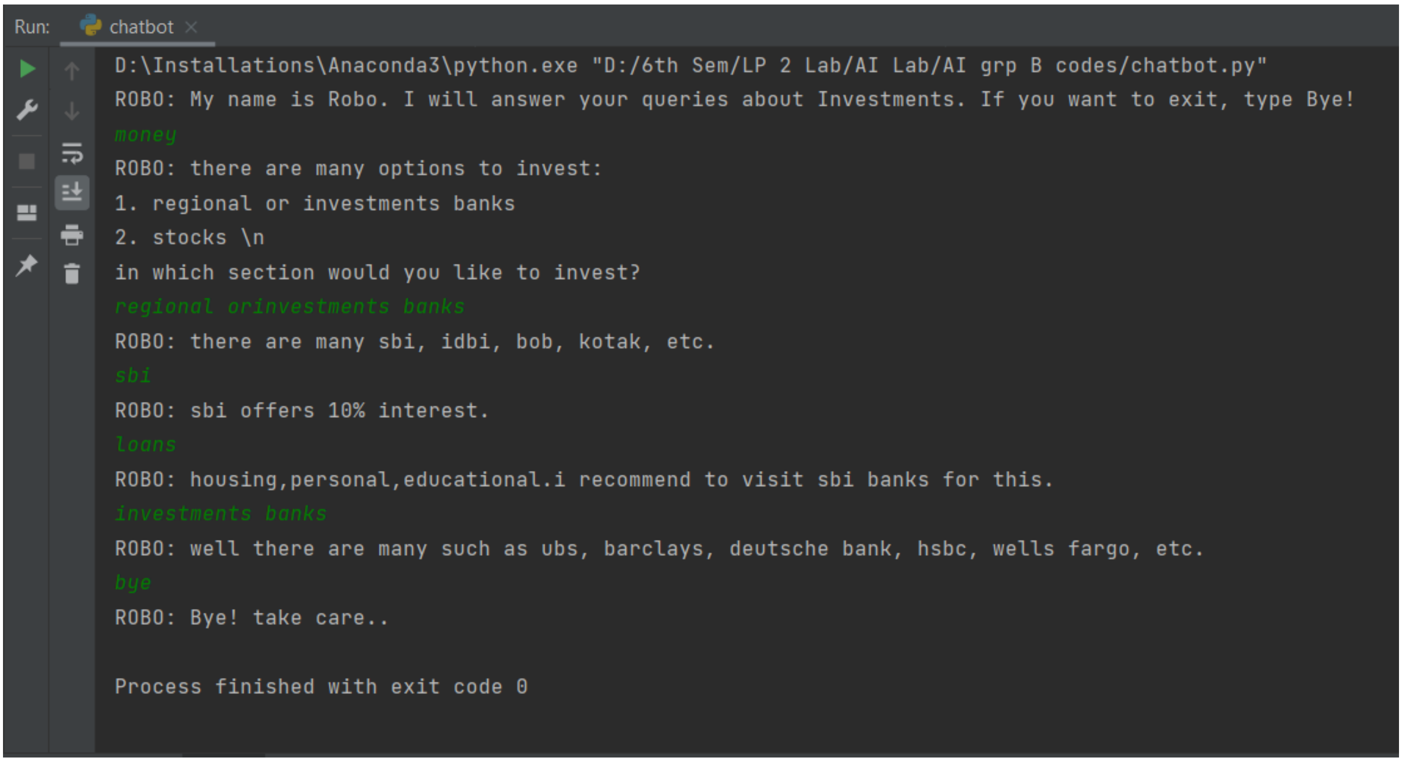
Code

```
import io
import random
import string
import warnings
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import warnings
warnings.filterwarnings('ignore')
import nltk
from nltk.stem import WordNetLemmatizer
# nltk.download('popular', quiet=True)
# nltk.download('punkt')
# nltk.download('wordnet')

with open('chatbot.txt','r', encoding='utf8', errors ='ignore') as fin:
    raw = fin.read().lower()

#Tokenisation
sent_tokens = nltk.sent_tokenize(raw)
word_tokens = nltk.word_tokenize(raw)

# Preprocessing
lemmer = WordNetLemmatizer()
def LemTokens(tokens):
    return [lemmer.lemmatize(token) for token in tokens]
remove_punct_dict = dict((ord(punct), None) for punct in
string.punctuation)
def LemNormalize(text):
    return
LemTokens(nltk.word_tokenize(text.lower().translate(remove_punct_dict)))
```

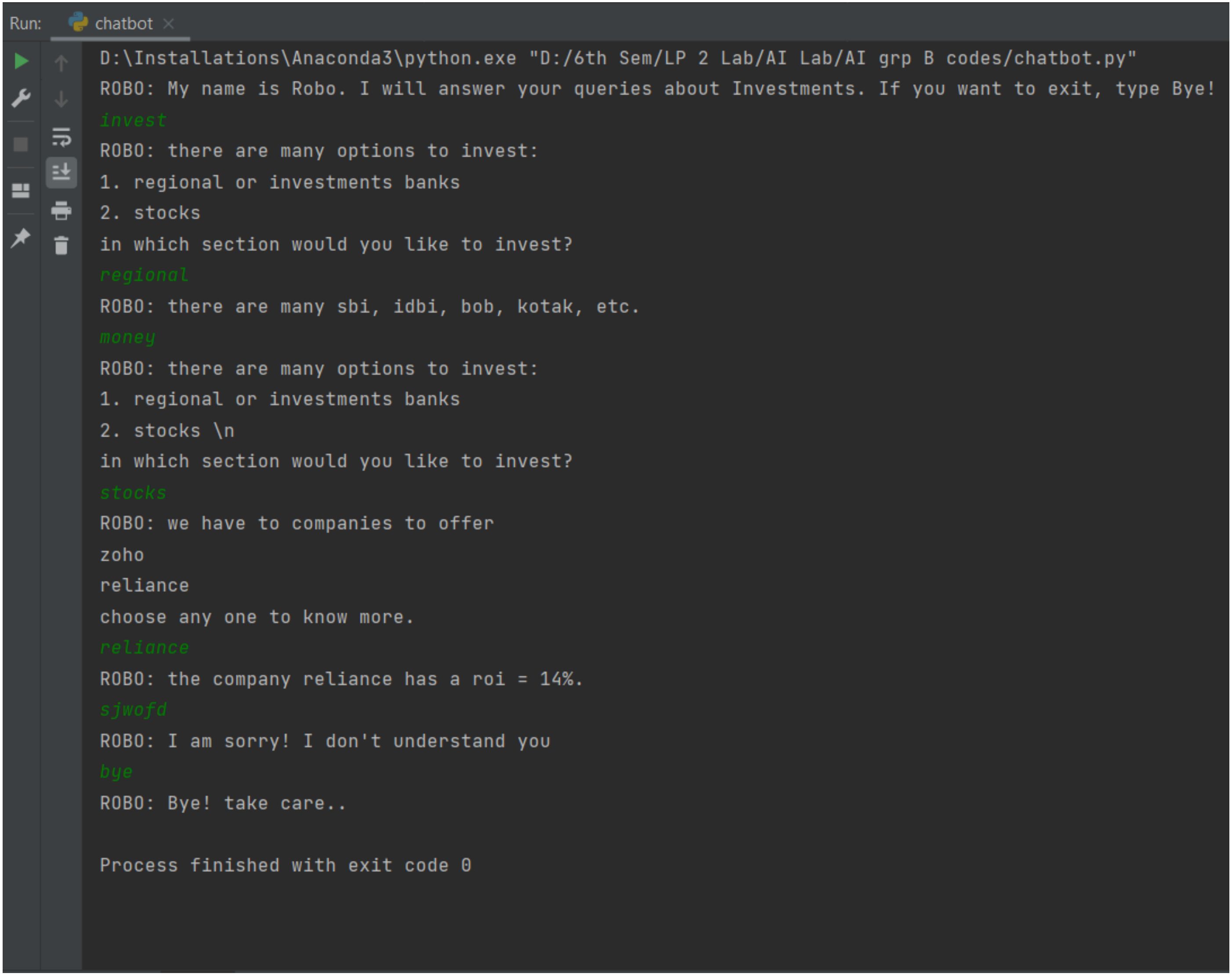
```
# Keyword Matching
GREETING_INPUTS = ("hello", "hi", "greetings", "sup", "what's
up","hey","Helo")
GREETING_RESPONSES = ["hi", "hey", "hi there", "hello", "I am glad! You
are talking to me"]


def greeting(sentence):
    for word in sentence.split():
        if word.lower() in GREETING_INPUTS:
            return random.choice(GREETING_RESPONSES)


def response(user_response):
    robo_response=''
    sent_tokens.append(user_response)
    TfidfVec = TfidfVectorizer(tokenizer=LemNormalize,
stop_words='english')
    tfidf = TfidfVec.fit_transform(sent_tokens)
    vals = cosine_similarity(tfidf[-1], tfidf)
    idx=vals.argsort()[0][-2]
    flat = vals.flatten()
    flat.sort()
    req_tfidf = flat[-2]
    if(req_tfidf==0):
        robo_response=robo_response+"I am sorry! I don't understand you"
        return robo_response
    else:
        robo_response = robo_response+sent_tokens[idx]
        return robo_response


flag=True
print("ROBO: My name is Robo. I will answer your queries about
Investments. If you want to exit, type Bye!")
while(flag==True):
    user_response = input()
    user_response=user_response.lower()
```

```
if(user_response!='bye'):
    if(user_response=='thanks' or user_response=='thank you' ):
        flag=False
        print("ROBO: You are welcome..")
    else:
        if(greeting(user_response)!=None):
            print("ROBO: "+greeting(user_response))
        else:
            print("ROBO: ",end="")
            res = response(user_response)
            nlines = res.count('\n')
            if nlines > 0:
                res = res.split("\n",1)[1]
            print(res)
            sent_tokens.remove(user_response)
else:
    flag=False
    print("ROBO: Bye! take care..")
```

Output



```
D:\Installations\Anaconda3\python.exe "D:/6th Sem/LP 2 Lab/AI Lab/AI grp B codes/chatbot.py"
ROBO: My name is Robo. I will answer your queries about Investments. If you want to exit, type Bye!
money
ROBO: there are many options to invest:
1. regional or investments banks
2. stocks \n
in which section would you like to invest?
regional orinvestments banks
ROBO: there are many sbi, idbi, bob, kotak, etc.
sbi
ROBO: sbi offers 10% interest.
loans
ROBO: housing,personal,educational.i recommend to visit sbi banks for this.
investments banks
ROBO: well there are many such as ubs, barclays, deutsche bank, hsbc, wells fargo, etc.
bye
ROBO: Bye! take care..

Process finished with exit code 0
```

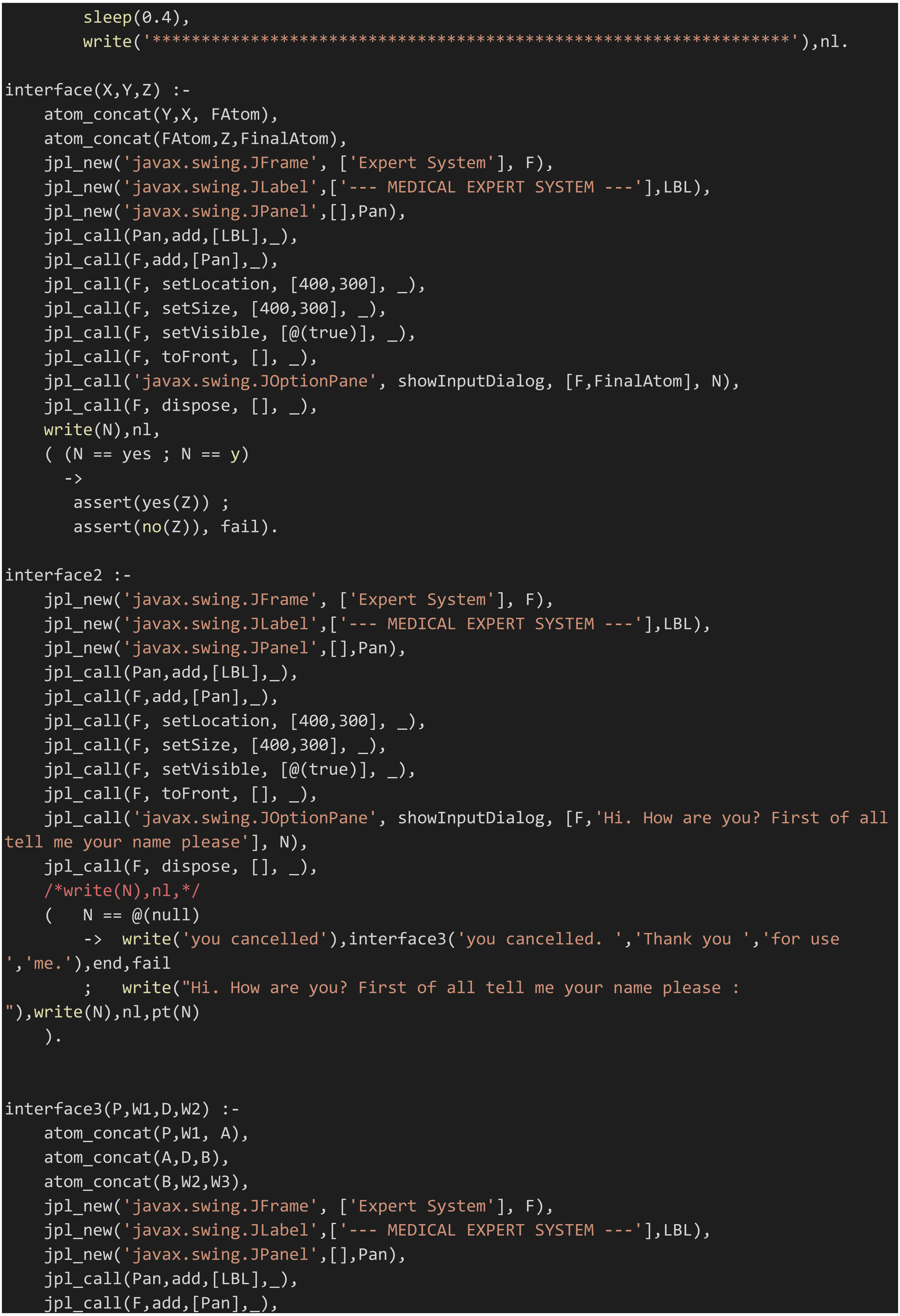


```
D:\Installations\Anaconda3\python.exe "D:/6th Sem/LP 2 Lab/AI Lab/AI grp B codes/chatbot.py"
ROBO: My name is Robo. I will answer your queries about Investments. If you want to exit, type Bye!
invest
ROBO: there are many options to invest:
1. regional or investments banks
2. stocks
in which section would you like to invest?
regional
ROBO: there are many sbi, idbi, bob, kotak, etc.
money
ROBO: there are many options to invest:
1. regional or investments banks
2. stocks \n
in which section would you like to invest?
stocks
ROBO: we have to companies to offer
zoho
reliance
choose any one to know more.
reliance
ROBO: the company reliance has a roi = 14%.
sjwofd
ROBO: I am sorry! I don't understand you
bye
ROBO: Bye! take care..

Process finished with exit code 0
```

AI-6 Medical Diagnosis Facility Expert System

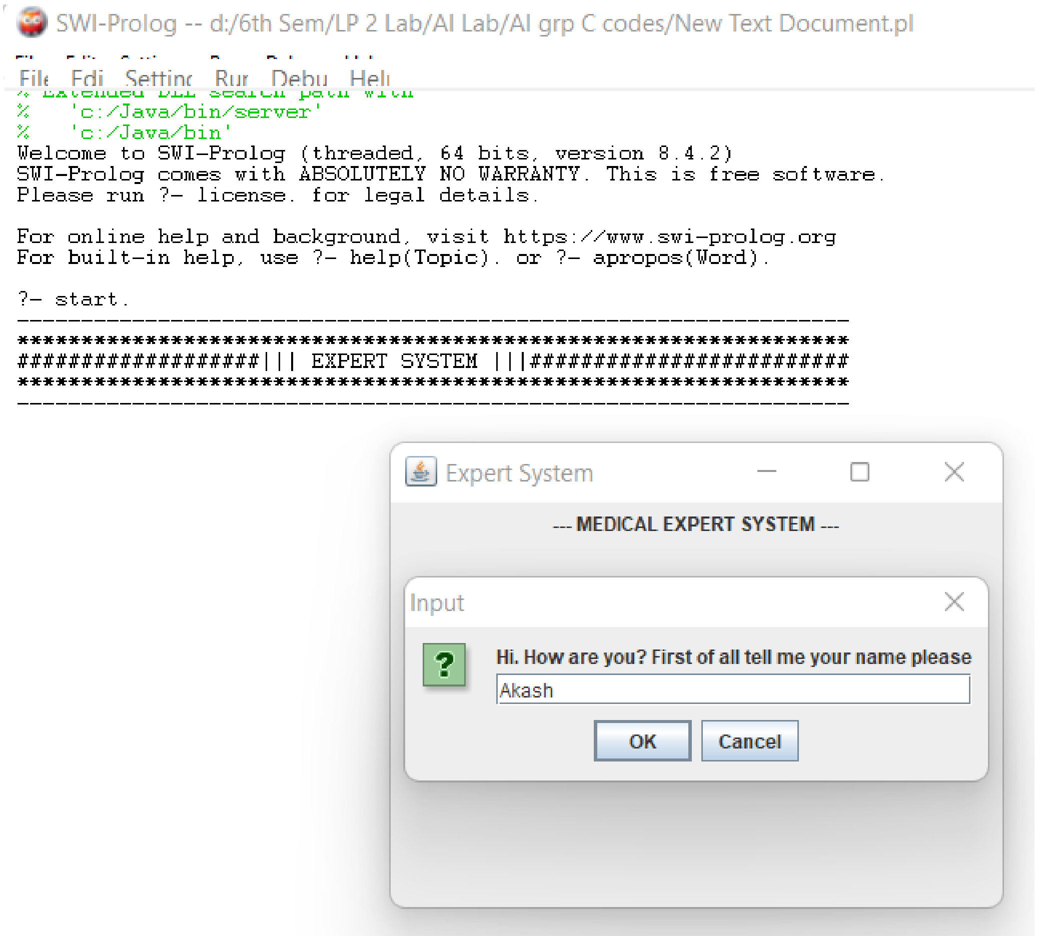
Code

```
%To Start the system type start.
% Name : - Chamara M. Dodandeniya

:- use_module(library(jpl)).
start :-sleep(0.4),
        write('-----------------------------------------------------------------'),nl,
        sleep(0.4),
        write('*****************************************************************'),nl,
        sleep(0.2),
        write("###################||| EXPERT SYSTEM |||#########################"),nl,
        sleep(0.4),
        write('*****************************************************************'),nl,
        sleep(0.4),
        write('-----------------------------------------------------------------
'),nl,nl,nl,


        /*write("Hi. How are you? First of all tell me your name Please : "),
        read(Patient),*/


        interface2.


       /* hypothesis(Patient,Disease),
        write(Patient),write(', you '), write(' probably have
'),write(Disease),write('.'),undo,
        nl,nl,nl,
        sleep(0.7),
        write('*****************************************************************'),nl,
        sleep(0.4),
        write("################||| THANK YOU FOR USE ME |||#####################"),nl,
        sleep(0.4),
        write('*****************************************************************'),nl.*/


    symptom(Patient,fever) :- verify(Patient," have a fever (y/n) ?").

    symptom(Patient,rash) :- verify(Patient," have a rash (y/n) ?").

    symptom(Patient,headache) :- verify(Patient," have a headache (y/n) ?").

    symptom(Patient,runny_nose) :- verify(Patient," have a runny_nose (y/n) ?").

    symptom(Patient,conjunctivitis) :- verify(Patient," have a conjunctivitis (y/n) ?").

    symptom(Patient,cough) :- verify(Patient," have a cough (y/n) ?").

    symptom(Patient,body_ache) :- verify(Patient," have a body_ache (y/n) ?").
    symptom(Patient,chills) :- verify(Patient," have a chills (y/n) ?").

    symptom(Patient,sore_throat) :- verify(Patient," have a sore_throat (y/n) ?").

    symptom(Patient,sneezing) :- verify(Patient," have a sneezing (y/n) ?").

    symptom(Patient,swollen_glands) :- verify(Patient," have a swollen_glands (y/n) ?").

    /*symptom(_,"Sorry, I don't seem to be able to diagnose the disease.").*/


    hypothesis(Patient,measles) :-
        symptom(Patient,fever),
        symptom(Patient,cough),
        symptom(Patient,conjunctivitis),
        symptom(Patient,runny_nose),
        symptom(Patient,rash).

    hypothesis(Patient,german_measles) :-
        symptom(Patient,fever),
        symptom(Patient,headache),
        symptom(Patient,runny_nose),
        symptom(Patient,rash).

    hypothesis(Patient,flu) :-
        symptom(Patient,fever),
        symptom(Patient,headache),
        symptom(Patient,body_ache),
        symptom(Patient,conjunctivitis),
        symptom(Patient,chills),
        symptom(Patient,sore_throat),
        symptom(Patient,runny_nose),
        symptom(Patient,cough).

    hypothesis(Patient,common_cold) :-
        symptom(Patient,headache),
        symptom(Patient,sneezing),
        symptom(Patient,sore_throat),
        symptom(Patient,runny_nose),
        symptom(Patient,chills).

    hypothesis(Patient,mumps) :-
        symptom(Patient,fever),
        symptom(Patient,swollen_glands).

    hypothesis(Patient,chicken_pox) :-
        symptom(Patient,fever),
        symptom(Patient,chills),
        symptom(Patient,body_ache),
        symptom(Patient,rash).

    hypothesis(Patient,measles) :-
        symptom(Patient,cough),
        symptom(Patient,sneezing),
        symptom(Patient,runny_nose).

    hypothesis(_,"disease. But I'm Sorry, I don't seem to be able to diagnose the
disease").

    response(Reply) :-
        read(Reply),
        write(Reply),nl.

ask(Patient,Question) :-
    write(Patient),write(', do you'),write(Question),
    /*read(N),
    ( (N == yes ; N == y)
      ->
       assert(yes(Question)) ;
       assert(no(Question)), fail),*/

    interface(', do you',Patient,Question),
    write('Loading.'),nl,
    sleep(1),
    write('Loading..'),nl,
    sleep(1),
    write('Loading...'),nl,
    sleep(1),
    nl.

:- dynamic yes/1,no/1.

verify(P,S) :-
   (yes(S)
    ->
    true ;
    (no(S)
     ->
     fail ;
     ask(P,S))).

undo :- retract(yes(_)),fail.
undo :- retract(no(_)),fail.
undo.


pt(Patient):-

        hypothesis(Patient,Disease),
        interface3(Patient,', you probably have ',Disease,'.'),
        write(Patient),write(', you probably have '),write(Disease),write('.'),undo,end.

end :-
        nl,nl,nl,
        sleep(0.7),
        write('*****************************************************************'),nl,
        sleep(0.4),
        write("################||| THANK YOU FOR USE ME |||#####################"),nl,
```



```
        sleep(0.4),
        write('******************************************************************'),nl.

interface(X,Y,Z) :-
    atom_concat(Y,X, FAtom),
    atom_concat(FAtom,Z,FinalAtom),
    jpl_new('javax.swing.JFrame', ['Expert System'], F),
    jpl_new('javax.swing.JLabel',['--- MEDICAL EXPERT SYSTEM ---'],LBL),
    jpl_new('javax.swing.JPanel',[],Pan),
    jpl_call(Pan,add,[LBL],_),
    jpl_call(F,add,[Pan],_),
    jpl_call(F, setLocation, [400,300], _),
    jpl_call(F, setSize, [400,300], _),
    jpl_call(F, setVisible, [@(true)], _),
    jpl_call(F, toFront, [], _),
    jpl_call('javax.swing.JOptionPane', showInputDialog, [F,FinalAtom], N),
    jpl_call(F, dispose, [], _),
    write(N),nl,
    ( (N == yes ; N == y)
      ->
       assert(yes(Z)) ;
       assert(no(Z)), fail).

interface2 :-
    jpl_new('javax.swing.JFrame', ['Expert System'], F),
    jpl_new('javax.swing.JLabel',['--- MEDICAL EXPERT SYSTEM ---'],LBL),
    jpl_new('javax.swing.JPanel',[],Pan),
    jpl_call(Pan,add,[LBL],_),
    jpl_call(F,add,[Pan],_),
    jpl_call(F, setLocation, [400,300], _),
    jpl_call(F, setSize, [400,300], _),
    jpl_call(F, setVisible, [@(true)], _),
    jpl_call(F, toFront, [], _),
    jpl_call('javax.swing.JOptionPane', showInputDialog, [F,'Hi. How are you? First of all
tell me your name please'], N),
    jpl_call(F, dispose, [], _),
    /*write(N),nl,*/
    (   N == @(null)
        ->  write('you cancelled'),interface3('you cancelled. ','Thank you ','for use
','me.'),end,fail
        ;   write("Hi. How are you? First of all tell me your name please :
"),write(N),nl,pt(N)
    ).


interface3(P,W1,D,W2) :-
    atom_concat(P,W1, A),
    atom_concat(A,D,B),
    atom_concat(B,W2,W3),
    jpl_new('javax.swing.JFrame', ['Expert System'], F),
    jpl_new('javax.swing.JLabel',['--- MEDICAL EXPERT SYSTEM ---'],LBL),
    jpl_new('javax.swing.JPanel',[],Pan),
    jpl_call(Pan,add,[LBL],_),
    jpl_call(F,add,[Pan],_),
    jpl_call(F, setLocation, [400,300], _),
    jpl_call(F, setSize, [400,300], _),
    jpl_call(F, setVisible, [@(true)], _),
    jpl_call(F, toFront, [], _),
    jpl_call('javax.swing.JOptionPane', showMessageDialog, [F,W3], N),
    jpl_call(F, dispose, [], _),
    /*write(N),nl,*/
    (   N == @(void)
        ->  write('')
        ;   write("")
    ).

help :- write("To start the expert system please type 'start.' and press Enter key").
```

Output



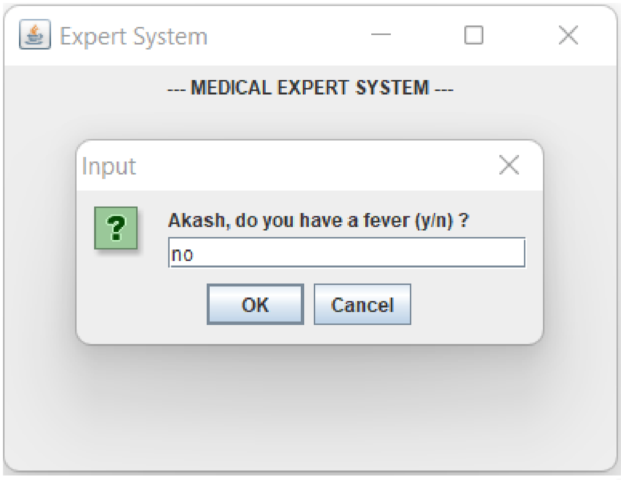
SWI-Prolog -- d:/6th Sem/LP 2 Lab/AI Lab/AI grp C codes/New Text Document.pl
File Edi Settinc Rur Debu Heli

```
% Extended DLL search path with
%   'c:/Java/bin/server'
%   'c:/Java/bin'
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- start.
-----------------------------------------------------------------
*****************************************************************
###################||| EXPERT SYSTEM |||#########################
*****************************************************************
-----------------------------------------------------------------
```

Expert System
--- MEDICAL EXPERT SYSTEM ---
Input

Hi. How are you? First of all tell me your name please

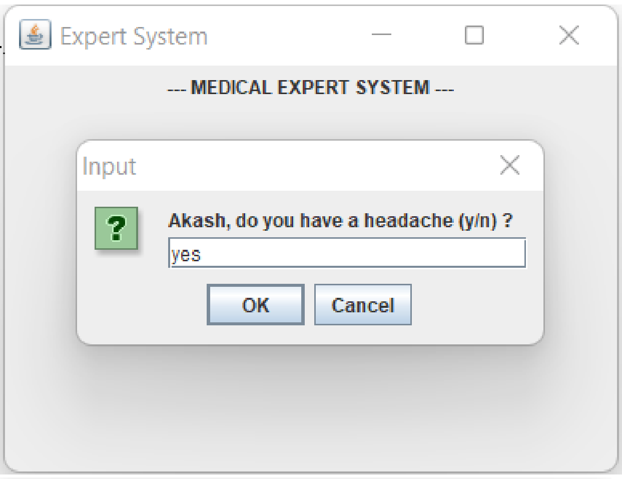
Akash



--- MEDICAL EXPERT SYSTEM ---

Akash, do you have a fever (y/n) ?

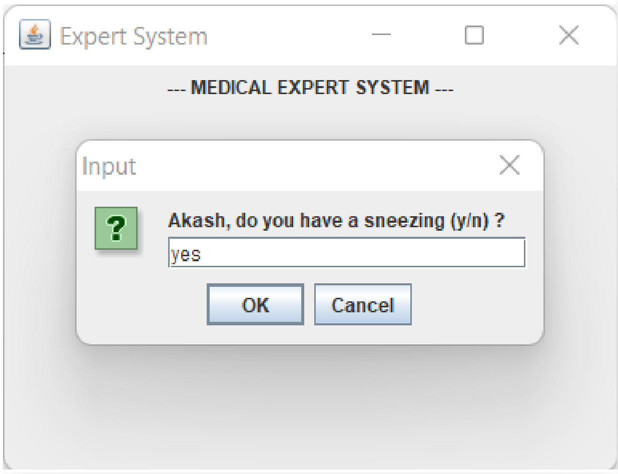
no



--- MEDICAL EXPERT SYSTEM ---

Akash, do you have a headache (y/n) ?

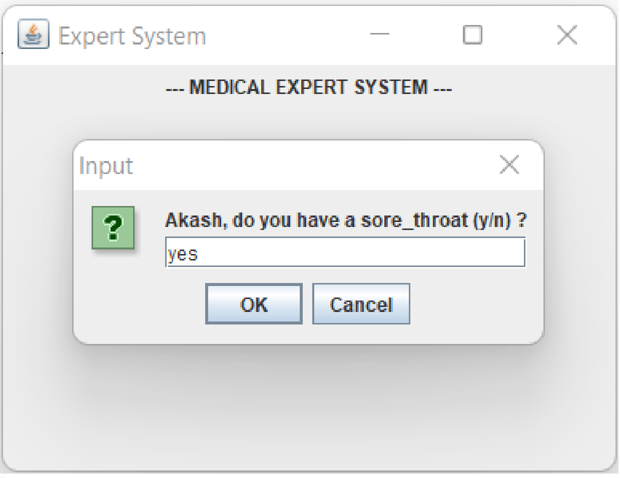
yes



--- MEDICAL EXPERT SYSTEM ---

Akash, do you have a sneezing (y/n) ?

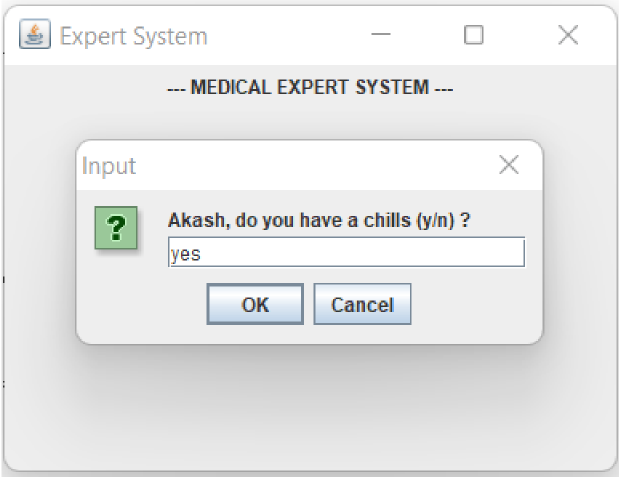
yes



--- MEDICAL EXPERT SYSTEM ---

Akash, do you have a sore_throat (y/n) ?

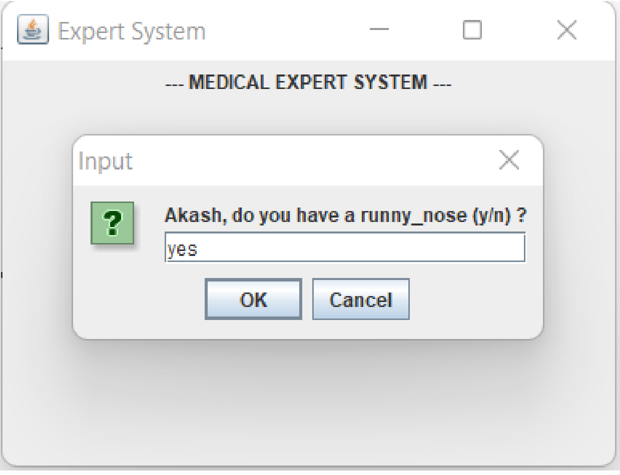
yes



--- MEDICAL EXPERT SYSTEM ---

Akash, do you have a chills (y/n) ?

yes

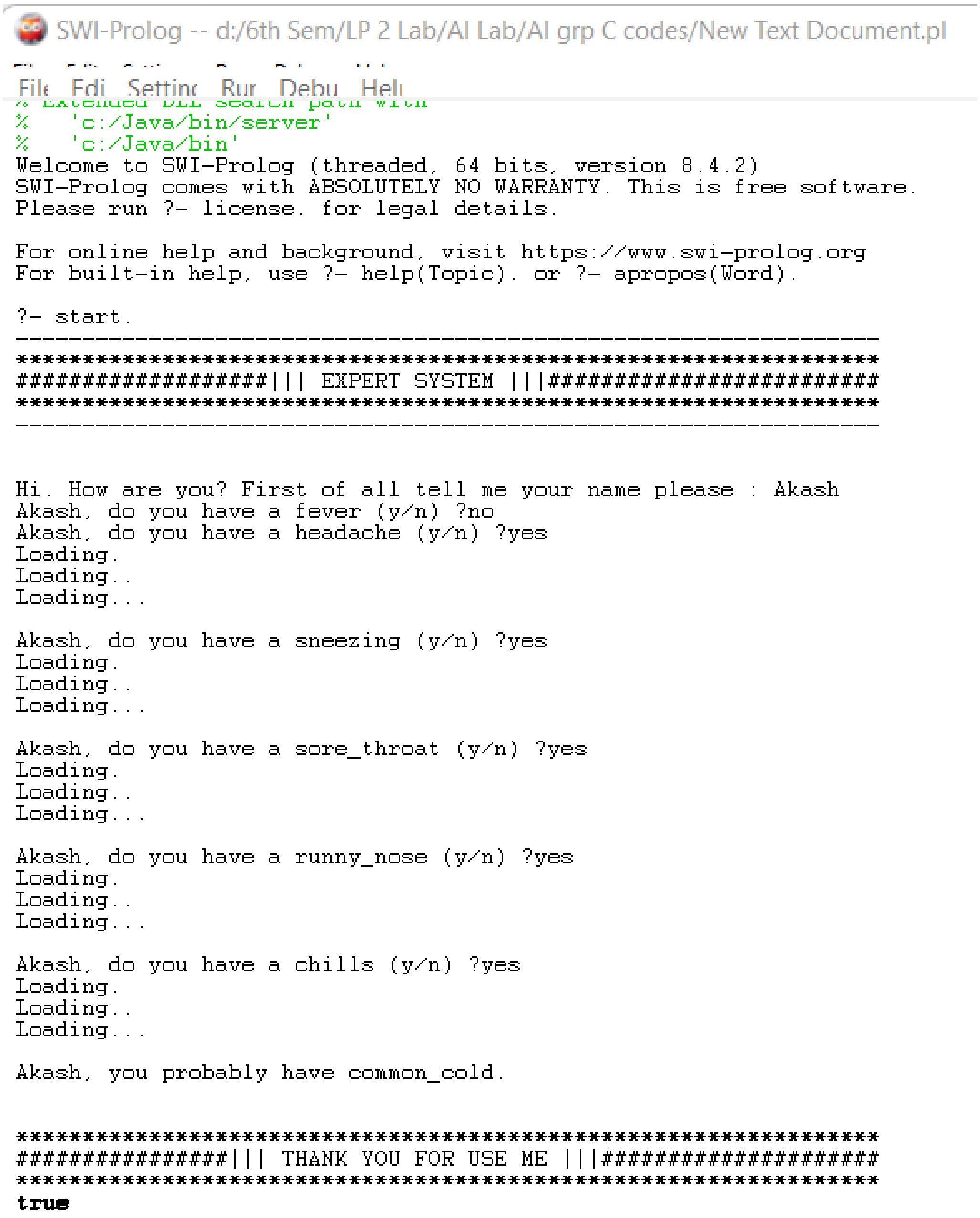


--- MEDICAL EXPERT SYSTEM ---

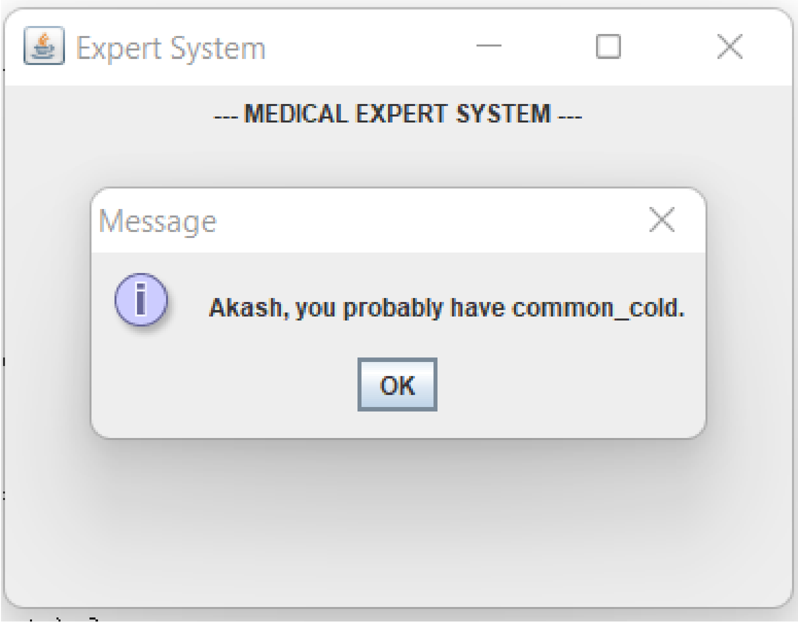
Akash, do you have a runny_nose (y/n) ?

yes

Successfully Diagnosis:


SWI-Prolog -- d:/6th Sem/LP 2 Lab/AI Lab/AI grp C codes/New Text Document.pl
File Edi Settinc Rur Debu Hel

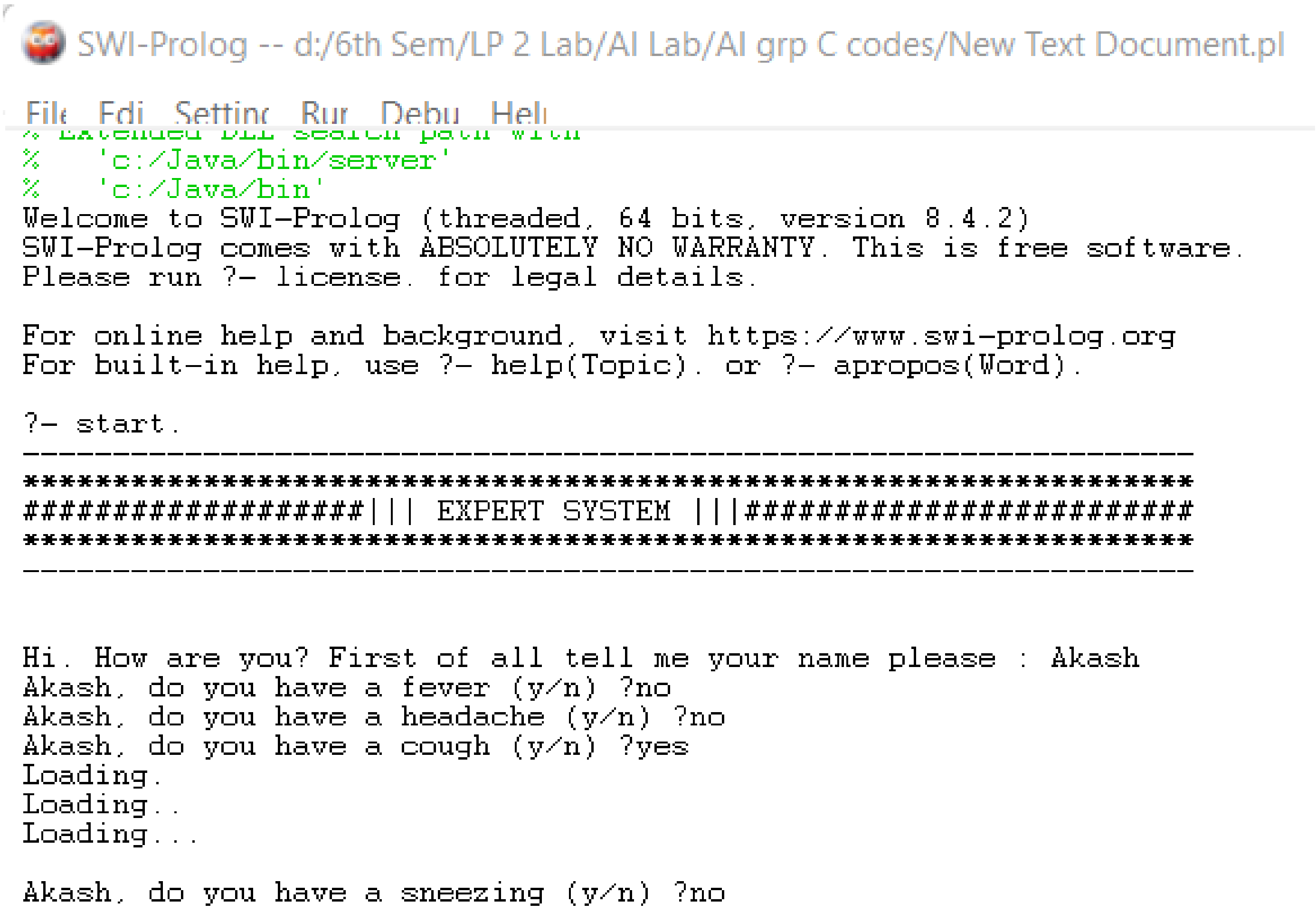
```
% Extended DLL search path with
%   'c:/Java/bin/server'
%   'c:/Java/bin'
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- start.
-----------------------------------------------------------------
*****************************************************************
###################||| EXPERT SYSTEM |||#########################
*****************************************************************
-----------------------------------------------------------------


Hi. How are you? First of all tell me your name please : Akash
Akash, do you have a fever (y/n) ?no
Akash, do you have a headache (y/n) ?yes
Loading.
Loading..
Loading...

Akash, do you have a sneezing (y/n) ?yes
Loading.
Loading..
Loading...

Akash, do you have a sore_throat (y/n) ?yes
Loading.
Loading..
Loading...

Akash, do you have a runny_nose (y/n) ?yes
Loading.
Loading..
Loading...

Akash, do you have a chills (y/n) ?yes
Loading.
Loading..
Loading...

Akash, you probably have common_cold.


*****************************************************************
################||| THANK YOU FOR USE ME |||#####################
*****************************************************************
true
```



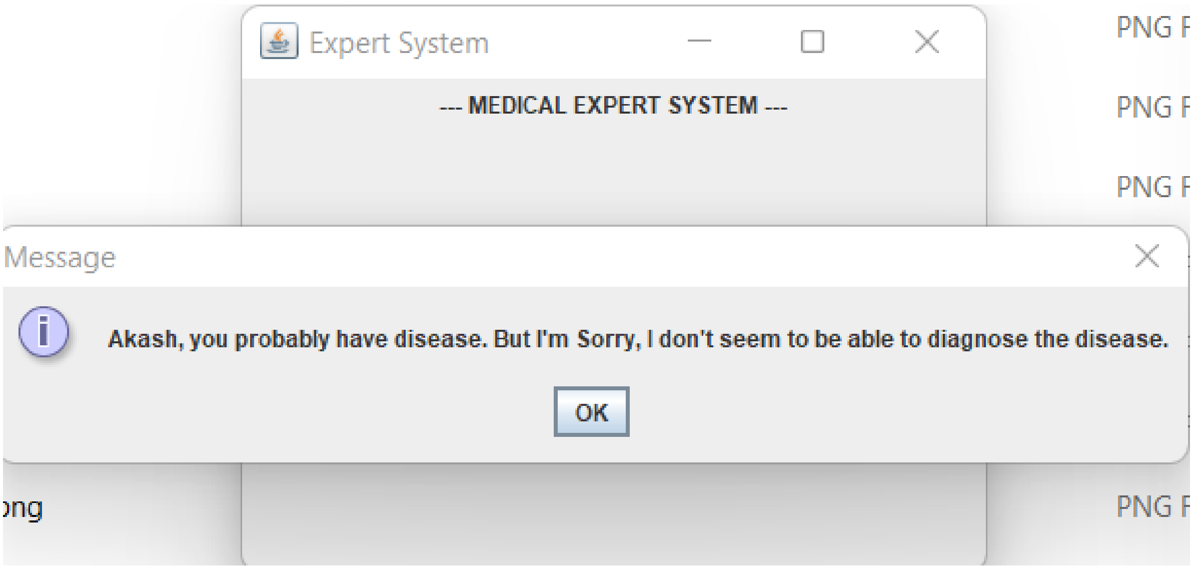
--- MEDICAL EXPERT SYSTEM ---

Akash, you probably have common_cold.

Unsuccessful/Exceptional Diagnosis:


SWI-Prolog -- d:/6th Sem/LP 2 Lab/AI Lab/AI grp C codes/New Text Document.pl
File Edi Settinc Rur Debu Heli

```
% Extended DLL search path with
%   'c:/Java/bin/server'
%   'c:/Java/bin'
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- start.
-----------------------------------------------------------------
*****************************************************************
###################||| EXPERT SYSTEM |||#########################
*****************************************************************
-----------------------------------------------------------------


Hi. How are you? First of all tell me your name please : Akash
Akash, do you have a fever (y/n) ?no
Akash, do you have a headache (y/n) ?no
Akash, do you have a cough (y/n) ?yes
Loading.
Loading..
Loading...

Akash, do you have a sneezing (y/n) ?no
```



Akash, you probably have disease. But I'm Sorry, I don't seem to be able to diagnose the disease.