



Ahsanullah University of Science and Technology

Department of Computer Science & Engineering

Course No.	CSE 4108
Course Name	Artificial Intelligence Lab
Assignment No.	03

Submitted To:

Md. Siam Ansary	Tonmoy Hossain
Department of CSE, AUST	Department of CSE, AUST

Submitted By:

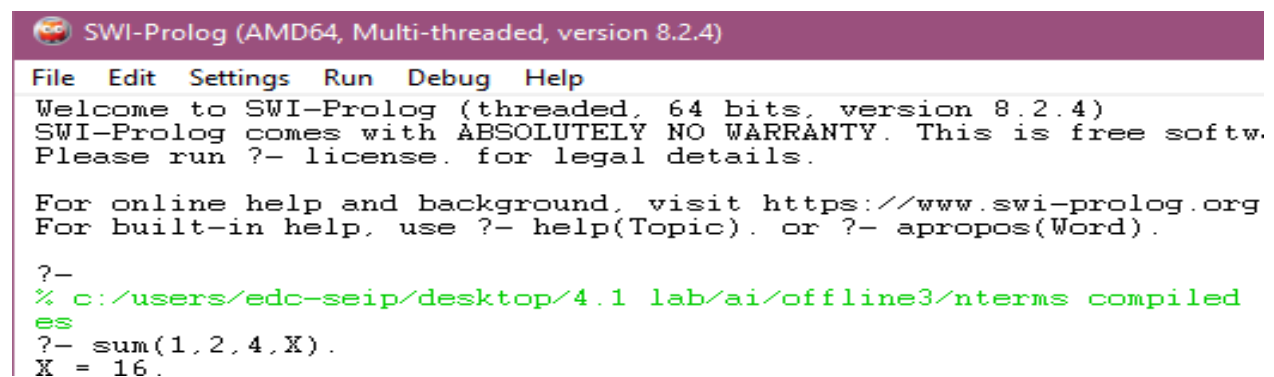
Name	Tahiya Ahmed Chowdhury
ID No.	17.02.04.048
Session	Fall – 2020
Section	A (A2)
Date of Submission:	August 9, 2021

Question 03: Define a recursive procedure in Python and in Prolog to find the sum of 1st n terms of an equal-interval series given the 1st term and the interval.

Prolog code:

```
sum(A,_,1,A):-  
    !.  
sum(A,D,N,S):-  
    N1 is N-1,  
    sum(A,D,N1,S1),  
    S is S1+(A+N1*D).
```

Prolog output:



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.4)  
File Edit Settings Run Debug Help  
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.4)  
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free softw.  
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).  
  
?-  
% c:/users/edc-seip/desktop/4.1 lab/ai/offline3/nterms compiled  
es  
?- sum(1,2,4,X).  
X = 16.
```

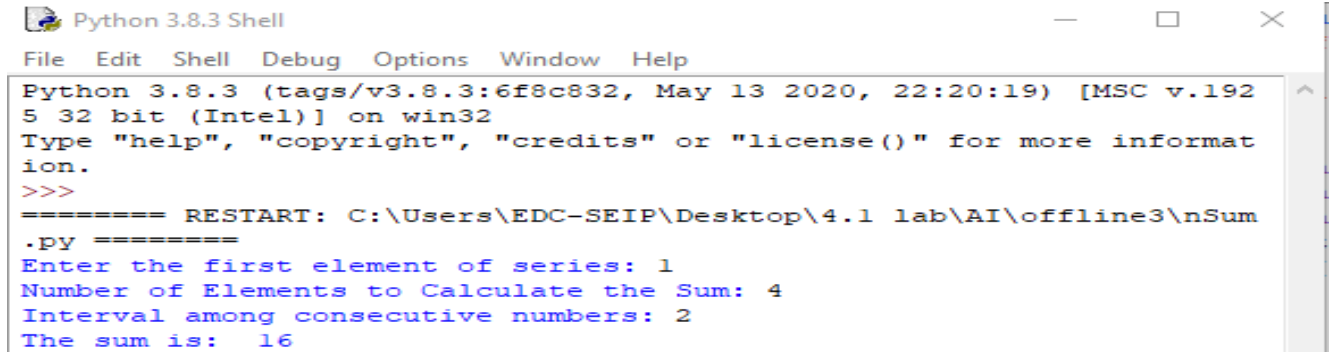
Analyzing code and above input and output:

In the prolog code, a recursive function `sum(A,D,N,S)` has been implemented where `S` is the sum of first `N` terms of a series starting with term `A` and difference `D`. If `N = 1` then `sum = first term` i.e. `A` is the base case condition. For our output, we enter the values for the series `1+3+5+7+ ...` and obtain the sum of first four terms i.e. `16`.

Python code:

```
def sumFunc(a,d,n):  
    if n == 1:  
        return a  
    else:  
        return a + (n-1)*d + sumFunc(a,d,n-1)  
  
a = int(input('Enter the first element of series: '))  
n = int(input('Number of Elements to Calculate the Sum: '))  
d = int(input('Interval among consecutive numbers: '))  
result = sumFunc(a,d,n)  
print('The sum is: ', result)
```

Python output:



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.192
5 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more informat
ion.
>>>
===== RESTART: C:\Users\EDC-SEIP\Desktop\4.1 lab\AI\offline3\nSum
.py =====
Enter the first element of series: 1
Number of Elements to Calculate the Sum: 4
Interval among consecutive numbers: 2
The sum is: 16
```

Analyzing code and above input and output:

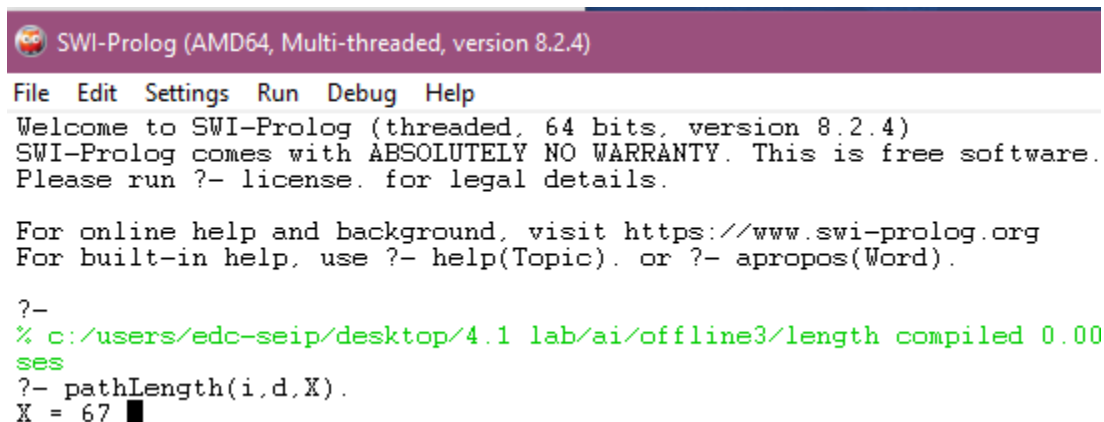
Similar to the prolog code, we implement a recursive function sum with base condition to return the sum of first n numbers of a series with first term a and difference d. After running the code, the values a = 1, n = 4 and d = 2 was entered to obtain result 16.

Question 04: Define a recursive procedure in Python and in Prolog to find the length of a path between two vertices of a directed weighted graph.

Prolog code:

```
neighbor(i,a,35). neighbor(i,b,45). neighbor(a,c,22).
neighbor(a,d,32). neighbor(b,d,28). neighbor(b,e,36).
neighbor(b,f,27). neighbor(c,d,31). neighbor(c,g,47).
neighbor(d,g,30). neighbor(e,g,26).
pathLength(X,Y,L):- neighbor(X,Y,L), !.
pathLength(X,Y,L):- neighbor(X,Z,L1), pathLength(Z,Y,L2), L is L1+L2.
```

Prolog output:



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.4)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.4)
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).

?-
% c:/users/edc-seip/desktop/4.1 lab/ai/offline3/length compiled 0.00
ses
?- pathLength(i,d,X).
X = 67
```

Analyzing code and above input and output:

In our code, we implemented the recursive function `pathlength(X,Y,L)` which works on determining L, the length between path X and Y when the credentials of a directed weighted graph was represented by `neighbor(_,_)`. If there is no direct path between X and Y then the function finds intermediate path between them to connect. But if it fails then none is returned as answer. For checking our code, we provided two vertices of our graph : i and d and got 67.

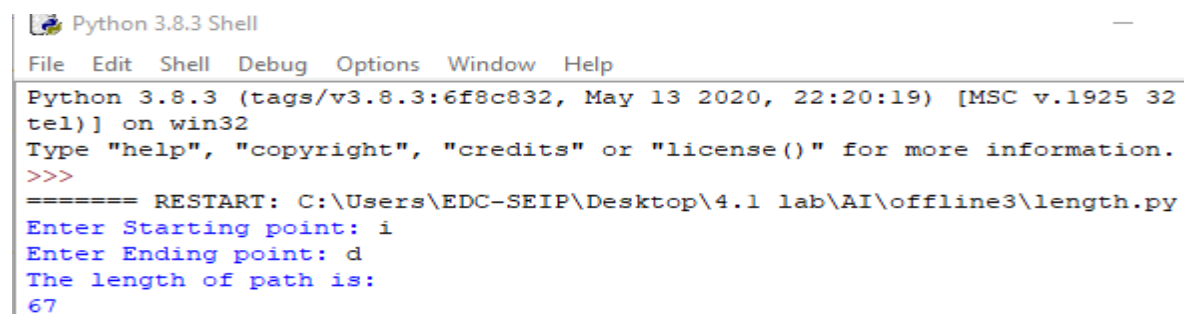
Python code:

```
def pathValue(x,y):
    if (x,y) in neighborList:
        return neighborDict[(x,y)]
    for (i, j) in neighborList:
        if i == x:
            t1 = neighborDict[(x, j)]
            t2 = pathValue(j,y)
            return t1+t2

neighborDict = {('i', 'a') : 35, ('i', 'b') : 45, ('a', 'c') : 22, ('a', 'd') : 32,
                ('b', 'd') : 28, ('b', 'e') : 36, ('b', 'f') : 27, ('c', 'd') : 31,
                ('c', 'g') : 47, ('d', 'g') : 30, ('e', 'g') : 26}
neighborList = [ ('i', 'a'), ('i', 'b'), ('a', 'c'), ('a', 'd'),
                  ('b', 'd'), ('b', 'e'), ('b', 'f'), ('c', 'd'),
                  ('c', 'g'), ('d', 'g'), ('e', 'g')]

start = str(input('Enter Starting point: '))
end = str(input('Enter Ending point: '))
print('The length of path is: ')
print(pathValue(start, end))
```

Python output:



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\EDC-SEIP\Desktop\4.1 lab\AI\offline3\length.py
Enter Starting point: i
Enter Ending point: d
The length of path is:
67
```

Analyzing code and above input and output:

For the python code also the recursive function `pathValue()` was implemented. Here as well the length was returned if direct path was found. Else the length of intermediate paths were stored in the sum and finally returned as output. Like in output, we provided i and d. Since no direct path exists among them hence the path calculated was $i \rightarrow a$ and $a \rightarrow d$ which was $32+35$ i.e. 67.

Question 05: Write the Python and Prolog codes to find discussed h2

Prolog code:

```
gtp(1,1,1). gtp(2,1,2). gtp(3,1,3). gtp(4,2,3).
gtp(5,3,3). gtp(6,3,2). gtp(7,3,1). gtp(8,2,1).
gblnk(2,2).

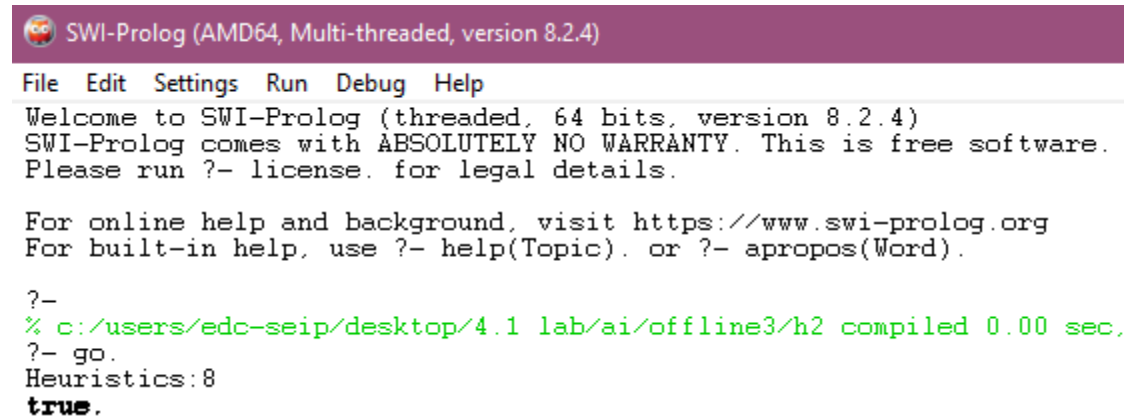
tp(1,1,2). tp(2,1,3). tp(3,2,1). tp(4,2,3).
tp(5,3,3). tp(6,2,2). tp(7,3,2). tp(8,1,1).
blnk(3,1).

go:-
    calcH(1,[],L),sumList(L,V),write('Heuristics:'),write(V).

calcH(9,X,X):-!.
calcH(T,X,Y):-
    dist(T,D), append(X,[D],X1), T1 is T+1, calcH(T1,X1,Y).

dist(T,V):-
    tp(T,A,B), gtp(T,C,D),V is abs(A-C) + abs(B-D).
sumList([],0):-!.
sumList(L,V):-L=[H|T],sumList(T,V1),V is V1+H.
```

Prolog output:



The screenshot shows the SWI-Prolog (AMD64, Multi-threaded, version 8.2.4) interface. The menu bar includes File, Edit, Settings, Run, Debug, and Help. The main window displays a welcome message and instructions. The user has entered the command `?- go.` and the output shows the heuristic value 8 and the result `true.`.

```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.4)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.4)
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).

?-
% c:/users/edc-seip/desktop/4.1 lab/ai/offline3/h2 compiled 0.00 sec.
?- go.
Heuristics:8
true.
```

Analyzing code and above input and output:

gtp represented goal state and tp represented current state. The differences in positions for the entries were calculated using Manhattan distance formula in dist() function. For our given set of input, the heuristic value i.e. h2 was 8 which means 8 entries were mismatched from the goal state.

Python code:

```
gtp = [(1,1,1), (2,1,2), (3,1,3), (4,2,3), (5,3,3), (6,3,2), (7,3,1), (8,2,1)]
gblnk = (2,2)

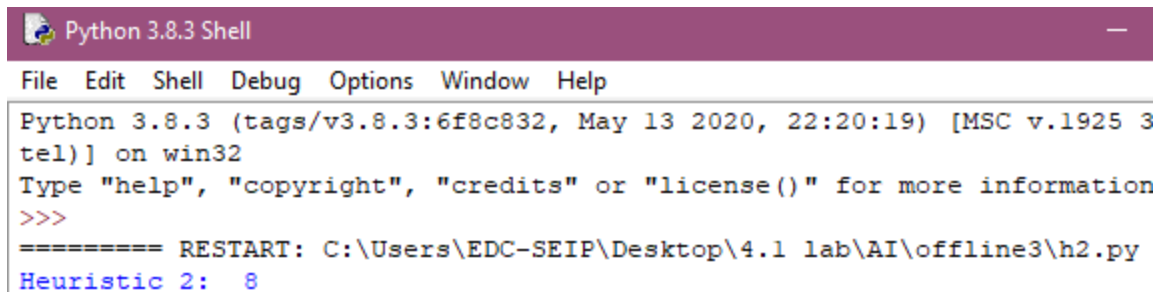
tp=[(1,1,2), (2,1,3), (3,2,1), (4,2,3), (5,3,3), (6,2,2), (7,3,2), (8,1,1)]
blnk = (3,1)

i,h = 0,0
d = []

while(i<=7):
    if( (gtp[i][1] != tp[i][1]) or (gtp[i][2] != tp[i][2]) ):
        d.append( abs(gtp[i][1] - tp[i][1]) + abs(gtp[i][2] - tp[i][2]) )
        i = i+1

for e in d:
    h = h+e
print("Heuristic 2: ",h)
```

Python output:



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 3
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information
>>>
===== RESTART: C:\Users\EDC-SEIP\Desktop\4.1 lab\AI\offline3\h2.py
Heuristic 2: 8
```

Analyzing code and above input and output:

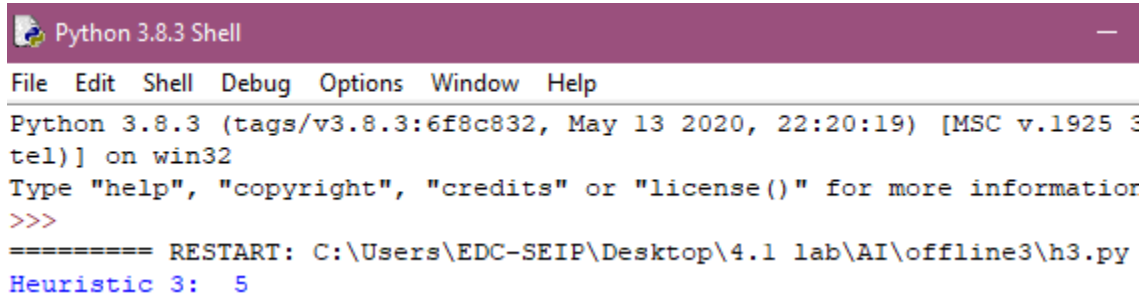
The Manhattan distance was calculated in a loop by comparing the vertices of each current state entries with that of goal state and stored in a list d. Finally the sum of list d was calculated which was 8 i.e. our h2.

Question 06: Write Python code to find the discussed h3.

Python code:

```
positionList = [(1,2),(1,8),(3,6),(4,5),(5,3),(6,1),(7,4),(8,7)]
h = 0
for i in range(0,7):
    for j in range(i+1,7):
        if positionList[i][0] == positionList[j][0]:
            h = h+1
        else:
            t1 = abs(positionList[i][0] - positionList[j][0])
            t2 = abs(positionList[i][1] - positionList[j][1])
            if t1 == t2:
                h = h+1
print("Heuristic 3: ",h)
```

Python output:



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32-bit Intel] on win32
Type "help", "copyright", "credits" or "license()" for more information
>>>
===== RESTART: C:\Users\EDC-SEIP\Desktop\4.1 lab\AI\offline3\h3.py
Heuristic 3: 5
```

Analyzing code and above input and output:

For finding an heuristic function (h3) that returns the number of attacking pairs of queens in an 8 Queens problem, we compared the positions of all queens present to the others starting from bottom to top. If any two were horizontally or diagonally aligned then h3 was incremented. Finally h3 was 5 for 4 diagonal matches and one horizontal match.