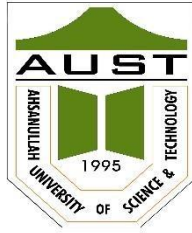


AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY
DHAKA-1208, BANGLADESH.



Department of Computer Science and Engineering
Spring 2019

Program: Bachelor of Science in Computer Science and Engineering

Course No: CSE 4108

Course Title: Artificial Intelligence Lab

Assignment No: 02

Date of Submission: 05/08/2019

Submitted to Dr. S.M. Abdullah Al-Mamun
Professor, Department of CSE, AUST.

Md. Siam Ansary
Adjunct Faculty, Department of CSE, AUST.

Submitted By

Name : Robiul Hasan Nowshad
Student ID : 16.01.04.061
Lab Group : B1

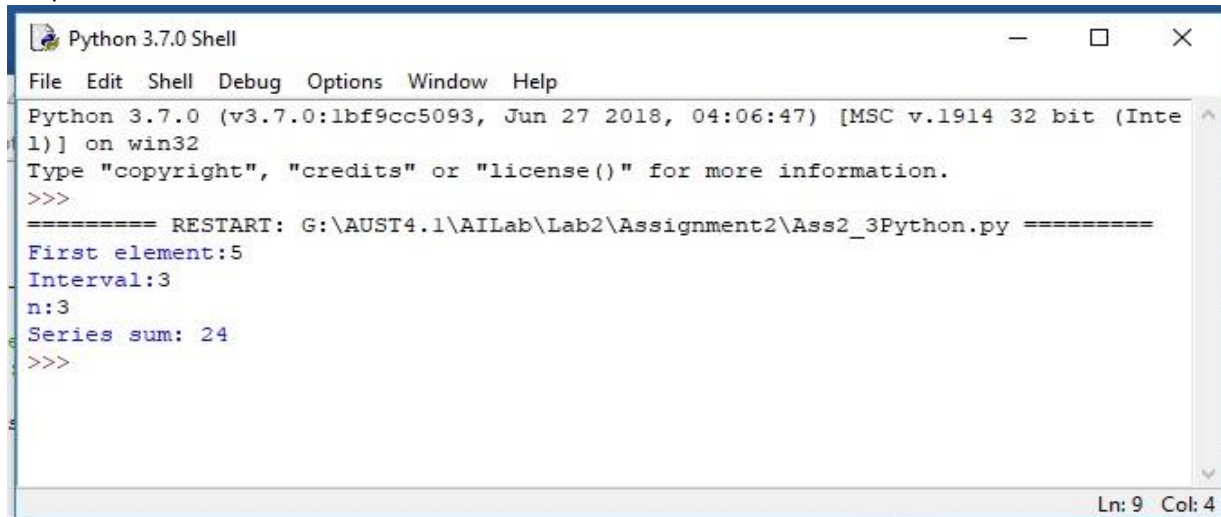
- 1) Define a recursive procedure in Python and in Prolog to find the sum of 1st n terms of an equal-interval series given the 1st term and the interval.

Solution:

Python

```
def ssum(N,I,F):
    if (N==0):
        return 0
    elif (N>=1):
        return ssum(N-1,I,F)+F+(N-1)*I
# Main
f=int(input('First element:'))
d=int(input('Interval:'))
n=int(input('n:'))
print('Series sum:', ssum(n,d,f))
```

Output



The screenshot shows a Python 3.7.0 Shell window with the following content:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: G:\AUST4.1\AILab\Lab2\Assignment2\Ass2_3Python.py =====
First element:5
Interval:3
n:3
Series sum: 24
>>>
```

The status bar at the bottom right indicates "Ln: 9 Col: 4".

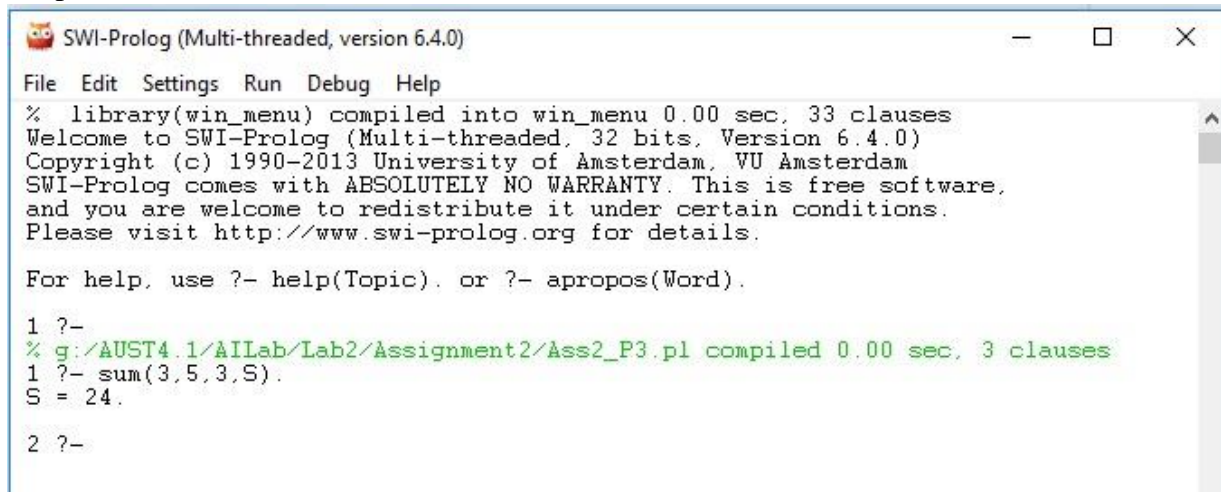
Prolog

sum(1,_,S,S):-!.

sum(N,I,T,S) :-

 N > 1 ,
 T1 is T+I ,
 N1 is N-1 ,
 sum(N1,I,T1,S1),
 S is S1+T.

Output



```
SWI-Prolog (Multi-threaded, version 6.4.0)
File Edit Settings Run Debug Help
% library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% g:/AUST4.1/AIILab/Lab2/Assignment2/Ass2_P3.pl compiled 0.00 sec, 3 clauses
1 ?- sum(3,5,3,S).
S = 24.

2 ?-
```

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

Solution:

Python

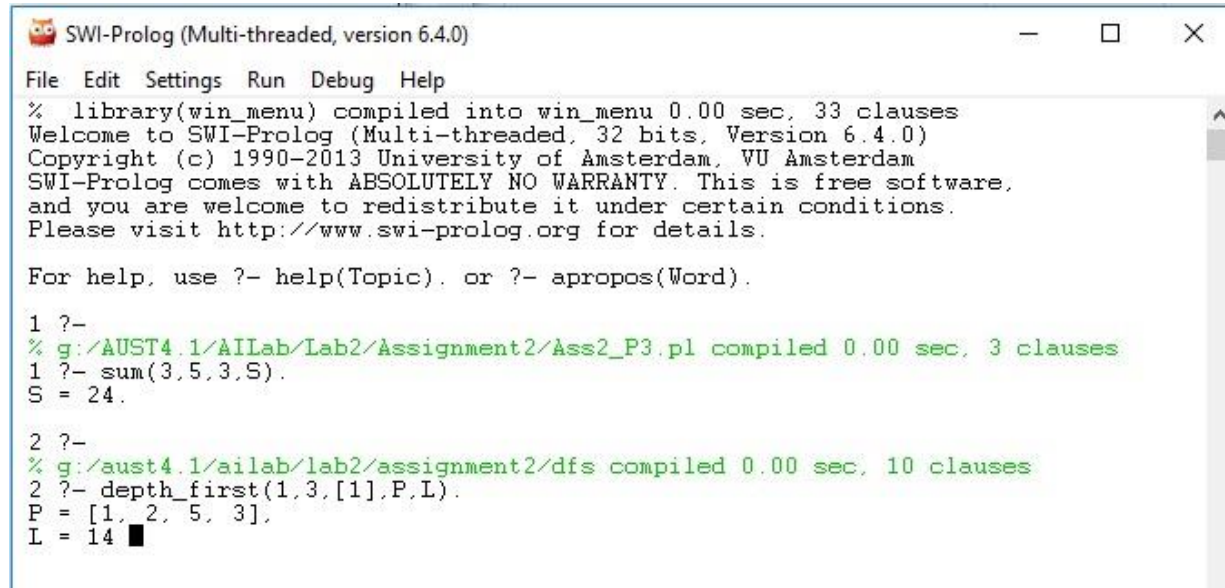
```
edge(1,2,10).
edge(1,4,11).
edge(2,5,3).
edge(5,4,8).
edge(5,3,1).
```

```
connected(X,Y,D) :- edge(X,Y,D).
```

```
next_node(Current, Next, Path) :-
    connected(Current, Next, _),
    not(member(Next, Path)).
```

```
depth_first(Goal, Goal, _, [Goal],0).
depth_first(Start, Goal, Visited, [Start|Path],L) :-
    next_node(Start, Next_node, Visited),
    edge(Start,Next_node,D),
    depth_first(Next_node, Goal, [Next_node|Visited], Path,L1),
    L is L1+D.
```

Output:



```
SWI-Prolog (Multi-threaded, version 6.4.0)
File Edit Settings Run Debug Help
% library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% g:/AUST4.1/AiLab/Lab2/Assignment2/Ass2_P3.pl compiled 0.00 sec, 3 clauses
1 ?- sum(3,5,3,S).
S = 24.

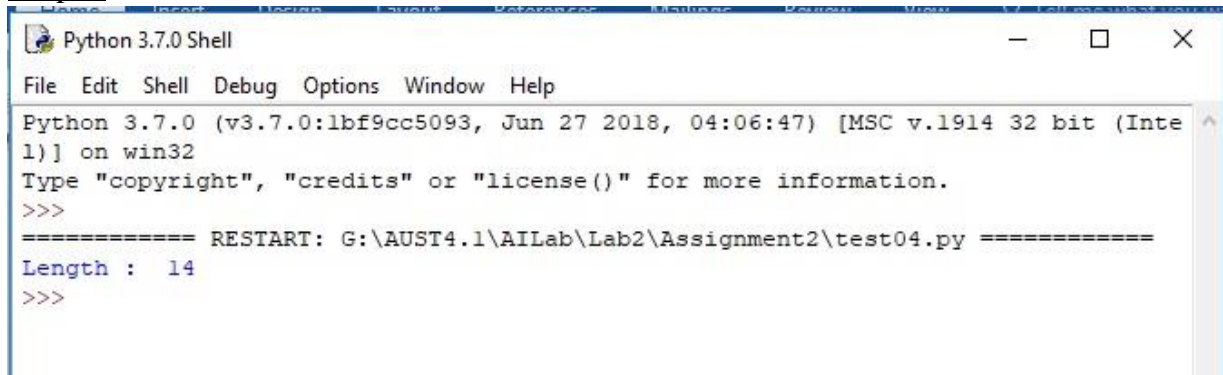
2 ?-
% g:/aust4.1/ailab/lab2/assignment2/dfs compiled 0.00 sec, 10 clauses
2 ?- depth_first(1,3,[1],P,L).
P = [1, 2, 5, 3],
L = 14
```

Python:

```
from array import *
v = array('i', [0,0,0,0,0])
Graph=[(1,2,10),
        (1,4,11),
        (2,5,3),
        (4,5,8),
        (5,3,1)]
def dfs(S,G,v):
    if(S==G):
        return 0
    else :
        for i in range(5):
            if((Graph[i][0]==S)&(v[Graph[i][1]]==0)):
                v.insert(Graph[i][1],1)
                return Graph[i][2]+dfs(Graph[i][1],G,v)

#Main
print('Length : ',dfs(1,3,v))
```

Output:



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: G:\AUST4.1\AILab\Lab2\Assignment2\test04.py =====
Length : 14
>>>
```

3) Modify the Python and Prolog codes demonstrated above to find h_2 and h_3 discussed above.

Solution:

Heuristic function (H2):

```
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.
```

Heuristic function (H3):

```
:-dynamic(hval/1).
```

```
/* Evaluates a 8-queens' state given as list of 8 digits */
```

```
evalState(L,V):- assert(hval(0)),hl(1,L), di_up(1,L),di_dn(1,L),hval(V),
    retractall(hval(_)).
```

```
hl(8,_):-!.
```

```
hl(I,L):- nthel(I,L,X), chk_incr(I,L,X), I1 is I+1, hl(I1,L).
```

```
chk_incr(8,_,_):-!.
```

```
chk_incr(I,L,X):- I1 is I+1, nthel(I1,L,Y),
```

```
do_incr(X,Y),chk_incr(I1,L,X).
```

```
do_incr(X,Y):- X=Y, incr_hval. do_incr(,_,_).
```

```
incr_hval:-hval(V), V1 is V+1, retract(hval(_)), assert(hval(V1)).
```

```
di_up(8,_,_):-!.
```

```
di_up(I,L):- nthel(I,L,X), chkup_incr(I,L,X,0), I1 is I+1,
```

```
di_up(I1,L).
```

```
chkup_incr(8,_,_,_):-!.
```

```
chkup_incr(I,L,X,K):- I1 is I+1, nthel(I1,L,Y), K1 is K+1,
```

```
doup_incr(X,Y,K1), chkup_incr(I1,L,X,K1).
```

```
doup_incr(X,Y,K1):- X1 is X+K1, Y=X1, incr_hval. doup_incr(,_,_).
```

```
di_dn(8,_,_):-!.
```

```
di_dn(I,L):- nthel(I,L,X), chkdn_incr(I,L,X,0), I1 is I+1,
```

```
di_dn(I1,L).
```

```
chkdn_incr(8,_,_,_):-!.
```

```
chkdn_incr(I,L,X,K):- I1 is I+1, nthel(I1,L,Y), K1 is K+1,
```

```
dodn_incr(X,Y,K1), chkdn_incr(I1,L,X,K1).
```

```
dodn_incr(X,Y,K1):- X1 is X-K1, Y=X1, incr_hval. dodn_incr(,_,_).
```

```
% A procedure to find the nth element of a list
```

```
nthel(N,[_|T],El):- N1 is N-1, nthel(N1,T,El).
```

```
nthel(1,[H|_],H):-!.
```