Questions:

1. **Define a recursive procedure in Python and Prolog to find the sum of 1st n terms of an equal-interval series give the 1st term and the interval.**
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.**
3. **Modify the Python and Prolog code demonstrated above to find h2 and h3 discussed above.**

Solution to the question no 1

The Code for the given problem in **Python** is ,

```
1.  def findSum(N,firstElement,interval):
2.      if(N == 0):
3.          return 0
4.      if(N == 1):
5.          return firstElement
6.      else:
7.          return findSum(N-1,interval,firstElement) + firstElement + (N-1) * interval
8.
9.  #Main
10.
11. t = int(input('How many times?'))
12. for i in range(t):
13.     print('Iteration: ', i+1)
14.     n = int(input('N : '))
15.     f = int(input('First Element : '))
16.     i = int(input('Interval : '))
17.     print('Sum is : ', findSum(n,f,i))
18.     print('\n')
```

A sample input and output is as below:

For the series 10 + 15 + 20 + 25, the output will be,

```
==== RESTART: C:\Users\Syed Sanzam\Desktop\AILab2\Assignment\python\3.py ====
How many times?1
Iteration:  1
N : 4
First Element : 10
Interval : 5
Sum is :  70
```

The Code for the given problem in **Prolog** is ,

```
sum(1,FirstElement,_,FirstElement):- !.
sum(N,FirstElement,Interval,Result):-
    N1 is N-1,
    sum(N1,FirstElement,Interval,TemporaryResult),
    Result is TemporaryResult + FirstElement + (N - 1) * Interval.
```

A sample input and output is as below:

For the series 10 + 15 + 20 + 25, the output will be,

```
?-
% c:/Users/Syed Sanzam/Desktop/AILab2/Assignment/prolog/3.pl compiled 0.00 sec,
0 clauses
?- sum(4,10,5,X).
X = 70.

?-
```

Solution to the question no 2

The Code for the given problem in **Python** is ,

```
graph = [('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),
         ]

totalTuple = len(graph)

def isNeighbor(n1,n2):
    i = 0
    value = -1
    while(i < totalTuple):
        if(graph[i][0] == n1):
            for j in range(totalTuple):
                if(graph[j][0] == n1 and graph[j][1] == n2):
                    value = graph[j][2]
        i = i + 1

    return value

```

```
28.
29. def findCommonNode(n1,n2):
30.     commonNode = []
31.     for i in range(totalTuple):
32.         if(graph[i][0] == n1):
33.             temp = graph[i][1]
34.             for j in range(totalTuple):
35.                 if(graph[j][0] == temp and graph[j][1] == n2):
36.                     commonNode.append(temp)
37.     return commonNode
38.
39.
40. def pathLength(n1,n2):
41.     if(isNeighbor(n1,n2) != -1):
42.         return isNeighbor(n1,n2)
43.     else:
44.         myList = findCommonNode(n1,n2)
45.         m = myList[0]
46.         return isNeighbor(n1,m) + isNeighbor(m,n2)
47.
48.
49. #Main
50. t = int(input('How many times?'))
51. for i in range(t):
52.     print('iteration: ', i+1)
53.     n1 = str(input('Source Node : '))
54.     n2 = str(input('Destination Node : '))
55.     print('Pathlength : ',pathLength(n1,n2))
56.     print('\n')
```

A sample input and output is as below:

```
 RESTART: C:\Users\Syed Sanzam\Desktop\AILab2\Assignment\python\graphtest.py
How many times?3
iteration:  1
Source Node : a
Destination Node : d
Pathlength :  32


iteration:  2
Source Node : i
Destination Node : c
Pathlength :  57


iteration:  3
Source Node : c
Destination Node : g
Pathlength :  47
```

The Code for the given problem in **Prolog** is ,

```
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):- pathLength(X,Z,L1),
                    pathLength(Z,Y,L2),
                    L is L1+L2.
```

A sample input and output is as below:

```
% c:/Users/Syed Sanzam/Desktop/AILab2/Assignment/prolog/4.pl compiled 0.00 sec,
0 clauses
?- pathLength(a,d,L).
L = 32.

?- pathLength(i,c,L).
L = 57.

?- pathLength(c,g,L).
L = 47.

?-
```

Solution to the question no 3

The Code for the determining h2 of the given problem in **Python** is ,

```
def findManhattanDistance():
    goalState=[(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)
    currentState=[(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)
    totalTuple = len(goalState)
    totalDistance = 0
    i = 0
    while(i < totalTuple):
        temp = currentState[i][0]
        for j in range(totalTuple):
            if(goalState[j][0] == temp):
                totalDistance = totalDistance + abs(currentState[j][1] - goalState[j][1
]) + abs(currentState[j][2] - goalState[j][2]) # abs(x1-x2) + abs(y1-y2)
        i = i+1
    return totalDistance

#Main
print('Tota distance :', findManhattanDistance())
print('\n')
```

A sample input and output is as below:

```
>>>
==== RESTART: C:\Users\Syed Sanzam\Desktop\AILab2\Assignment\python\5.py ====
Tota distance : 8
```

The Code for the determining h2 of the given problem in **Prolog** is ,

```
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.
```

A sample input and output is as below:

```
?-
% c:/Users/Syed Sanzam/Desktop/AILab2/Assignment/prolog/5.pl compiled 0.00 sec,
24 clauses
?- go.
Heuristics: 8
true.
```