Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

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

Submitted To :

Md. Siam Ansary
Department of CSE, AUST

Tonmoy Hossain
Department of CSE, AUST

Submitted By:

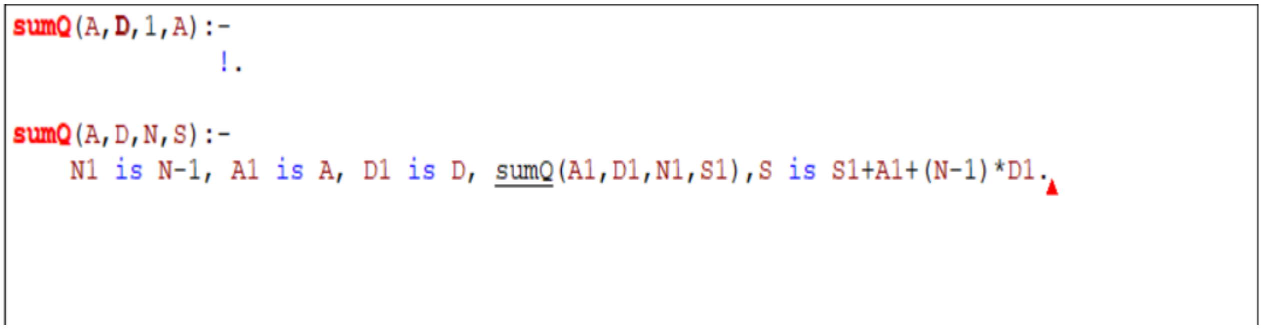
Name	:	Sanjida Akter Ishita
ID No.	:	170204089
Session	:	Fall 2020
Section	:	B
Lab group	:	B2

Date of Submission: August 18, 2021

QUESTION 01: 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.

ANSWER:

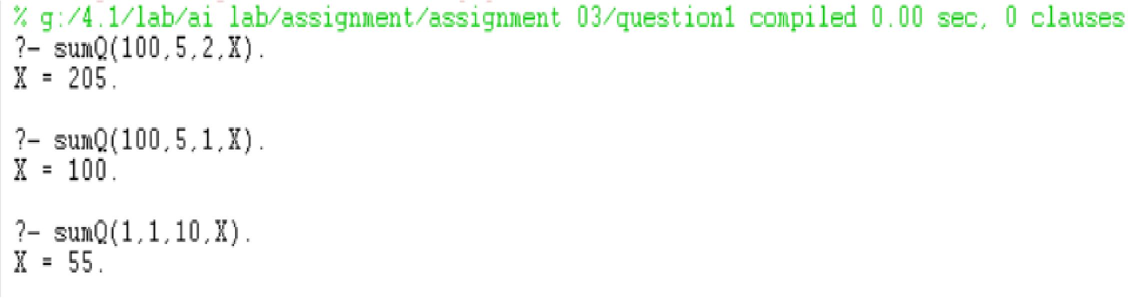
Code in Prolog:



```
sumQ(A,D,1,A):-
              !.

sumQ(A,D,N,S):-
    N1 is N-1, A1 is A, D1 is D, sumQ(A1,D1,N1,S1),S is S1+A1+(N-1)*D1.
```

Output:

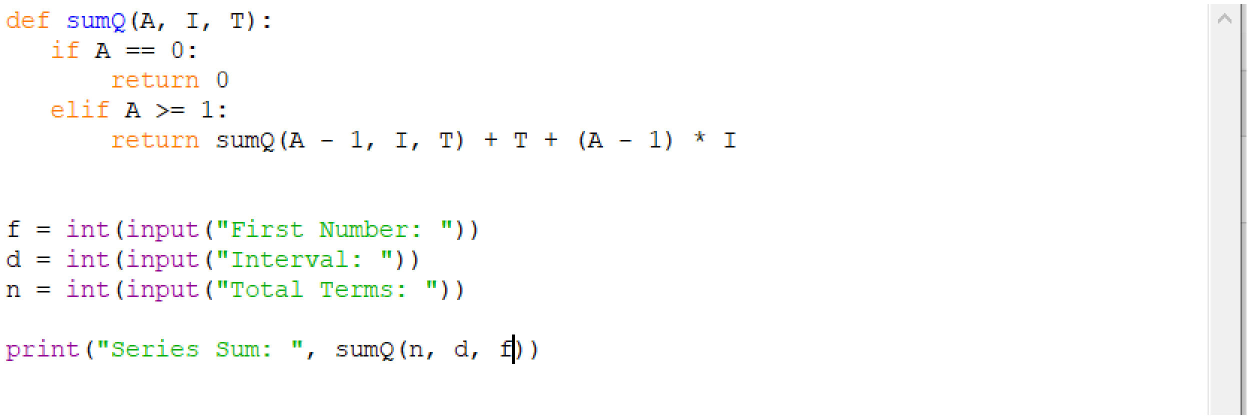


```
% g:/4.1/lab/ai lab/assignment/assignment 03/question1 compiled 0.00 sec, 0 clauses
?- sumQ(100,5,2,X).
X = 205.

?- sumQ(100,5,1,X).
X = 100.

?- sumQ(1,1,10,X).
X = 55.
```

Code in Python:

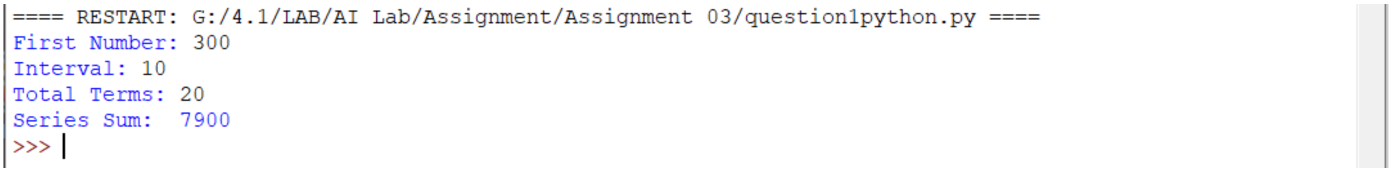


```
def sumQ(A, I, T):
    if A == 0:
        return 0
    elif A >= 1:
        return sumQ(A - 1, I, T) + T + (A - 1) * I


f = int(input("First Number: "))
d = int(input("Interval: "))
n = int(input("Total Terms: "))

print("Series Sum: ", sumQ(n, d, f))
```

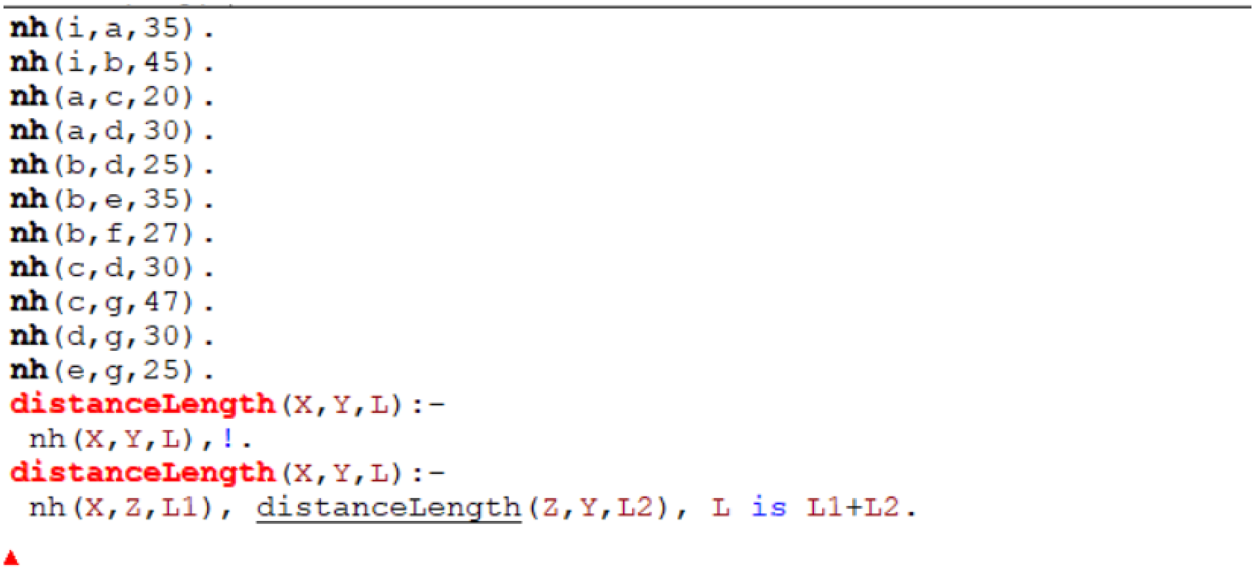
Output:



```
==== RESTART: G:/4.1/LAB/AI Lab/Assignment/Assignment 03/question1python.py ====
First Number: 300
Interval: 10
Total Terms: 20
Series Sum:  7900
>>> 
```

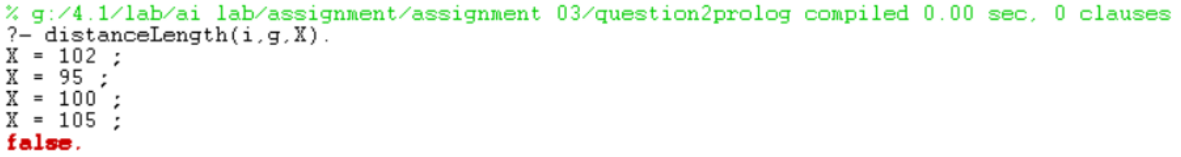
QUESTION 02: Define a recursive procedure in Python and in Prolog to find the length of path between two vertices of a directed weighted graph.

Prolog Code:



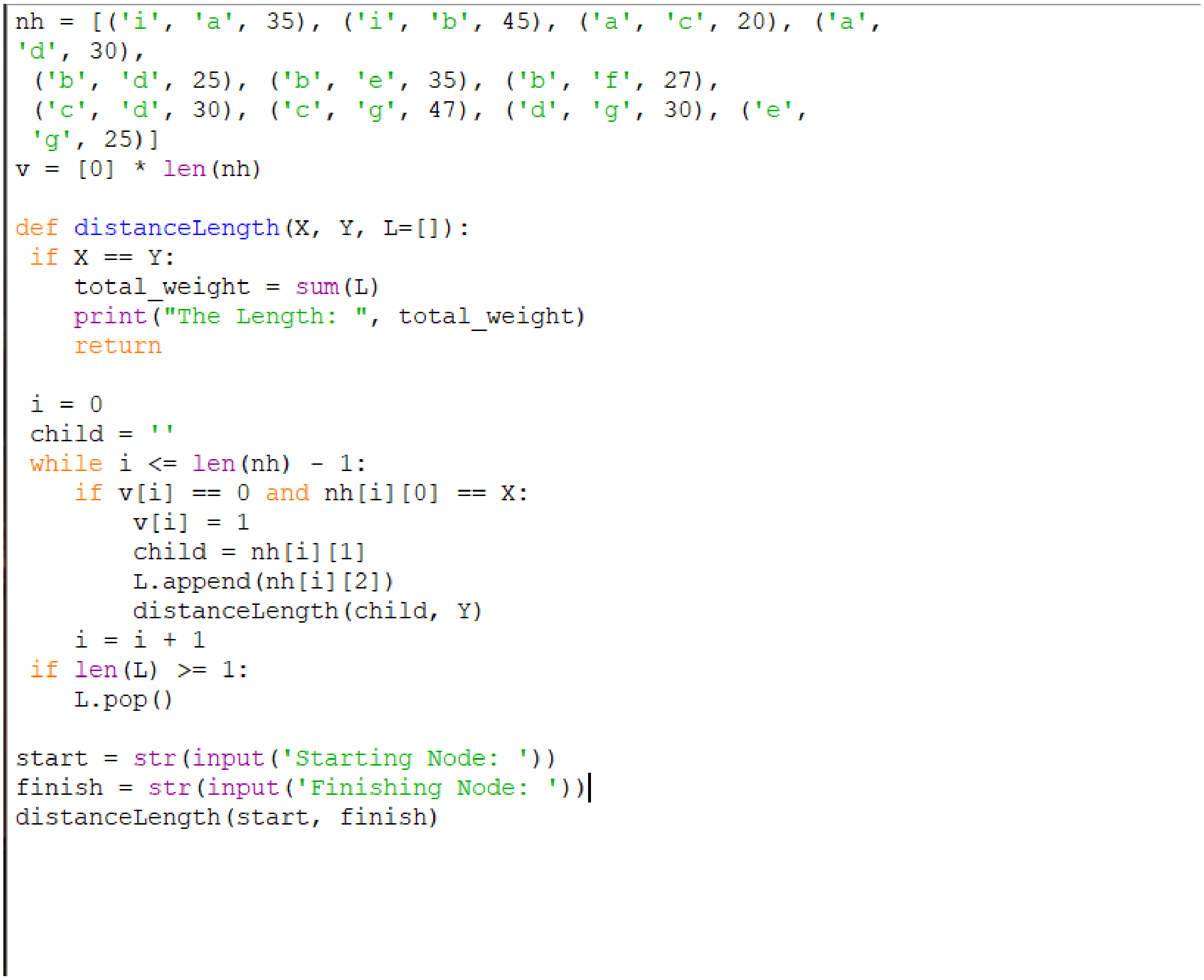
```
nh(i,a,35).
nh(i,b,45).
nh(a,c,20).
nh(a,d,30).
nh(b,d,25).
nh(b,e,35).
nh(b,f,27).
nh(c,d,30).
nh(c,g,47).
nh(d,g,30).
nh(e,g,25).
distanceLength(X,Y,L):-
 nh(X,Y,L),!.
distanceLength(X,Y,L):-
 nh(X,Z,L1), distanceLength(Z,Y,L2), L is L1+L2.
```

Output:



```
% g:/4.1/lab/ai lab/assignment/assignment 03/question2prolog compiled 0.00 sec, 0 clauses
?- distanceLength(i,g,X).
X = 102 ;
X = 95 ;
X = 100 ;
X = 105 ;
false.
```

Python Code:



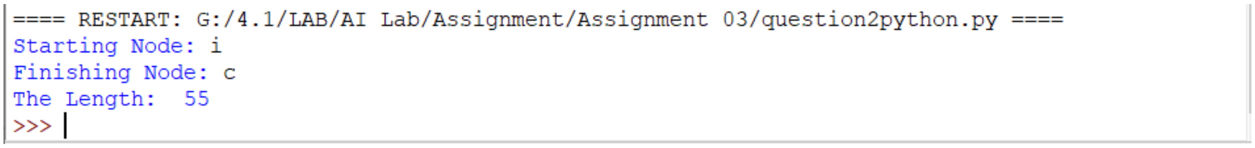
```
nh = [('i', 'a', 35), ('i', 'b', 45), ('a', 'c', 20), ('a',
'd', 30),
 ('b', 'd', 25), ('b', 'e', 35), ('b', 'f', 27),
 ('c', 'd', 30), ('c', 'g', 47), ('d', 'g', 30), ('e',
 'g', 25)]
v = [0] * len(nh)

def distanceLength(X, Y, L=[]):
 if X == Y:
    total_weight = sum(L)
    print("The Length: ", total_weight)
    return

 i = 0
 child = ''
 while i <= len(nh) - 1:
    if v[i] == 0 and nh[i][0] == X:
        v[i] = 1
        child = nh[i][1]
        L.append(nh[i][2])
        distanceLength(child, Y)
    i = i + 1
 if len(L) >= 1:
    L.pop()

start = str(input('Starting Node: '))
finish = str(input('Finishing Node: '))
distanceLength(start, finish)
```

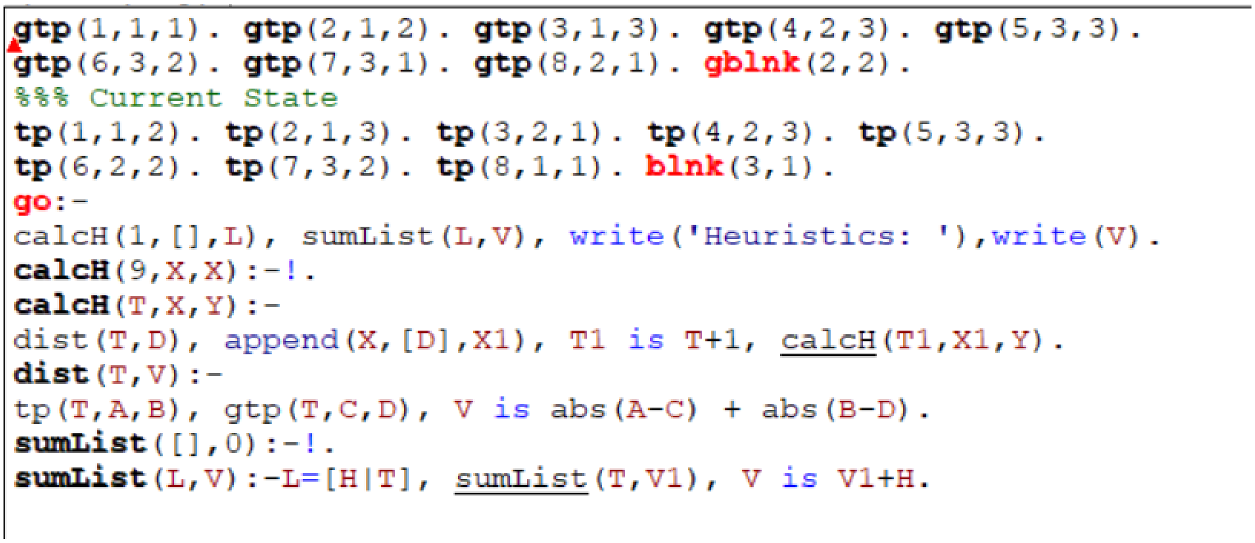
Output



```
==== RESTART: G:/4.1/LAB/AI Lab/Assignment/Assignment 03/question2python.py ====
Starting Node: i
Finishing Node: c
The Length:  55
>>> 
```

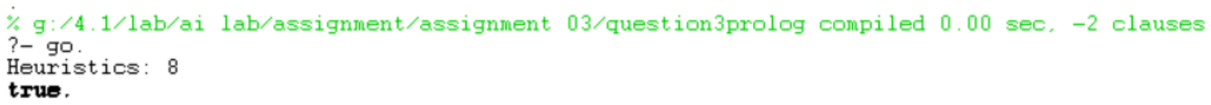
QUESTION 03: Write the Python and Prolog codes to find discussed h2.

Prolog Code:



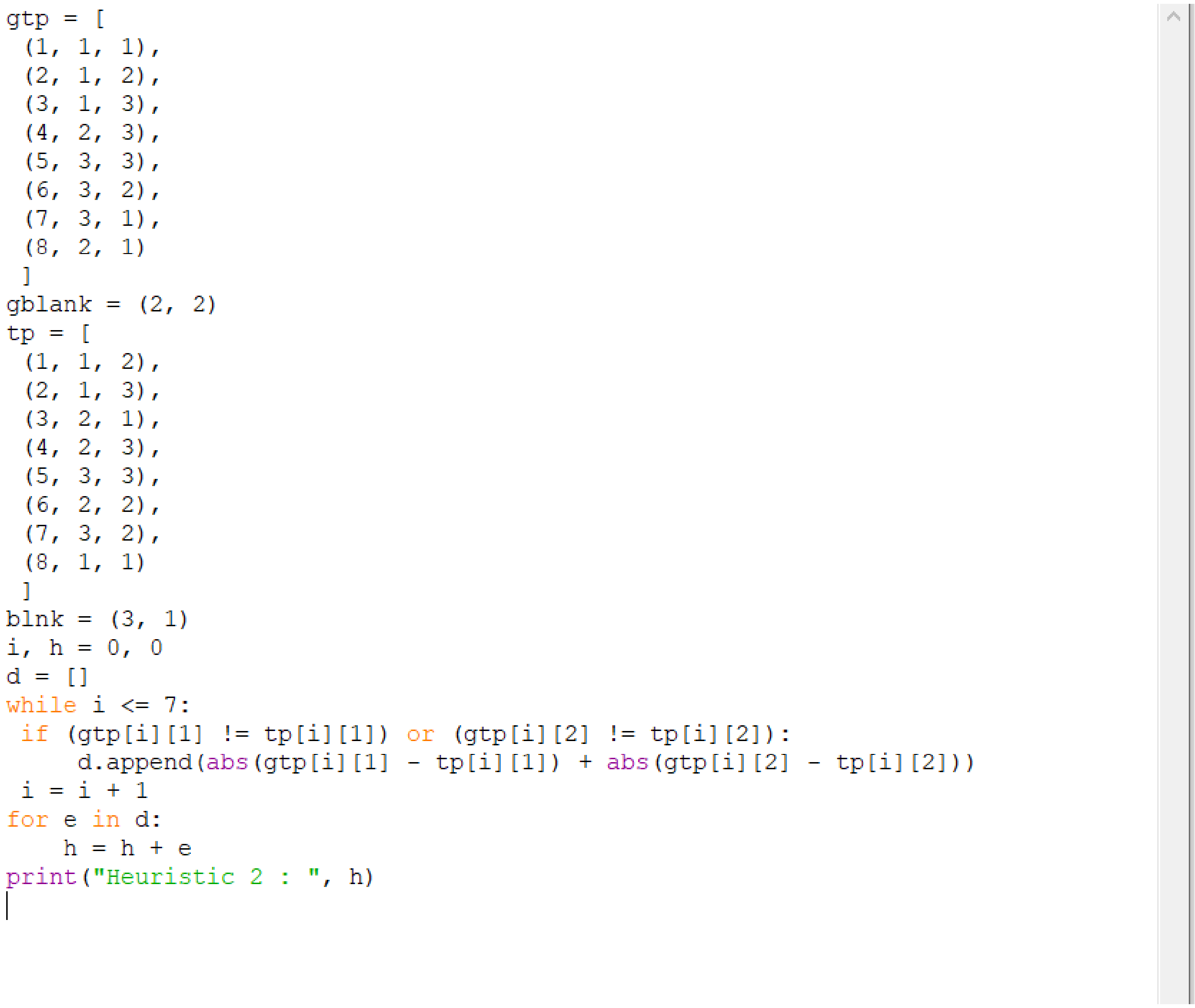
```
gtp(1,1,1). gtp(2,1,2). gtp(3,1,3). gtp(4,2,3). gtp(5,3,3).
gtp(6,3,2). gtp(7,3,1). gtp(8,2,1). gblnk(2,2).
%%% Current State
tp(1,1,2). tp(2,1,3). tp(3,2,1). tp(4,2,3). tp(5,3,3).
tp(6,2,2). tp(7,3,2). tp(8,1,1). blnk(3,1).
go:-
calcH(1,[],L), sumList(L,V), write('Heuristics: '),write(V).
calcH(9,X,X):-!.
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.
```

Output:



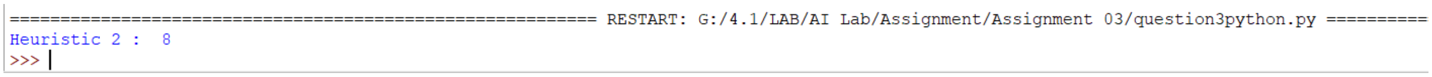
```
% g:/4.1/lab/ai lab/assignment/assignment 03/question3prolog compiled 0.00 sec, -2 clauses
?- go.
Heuristics: 8
true.
```

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)
 ]
gblank = (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)
```

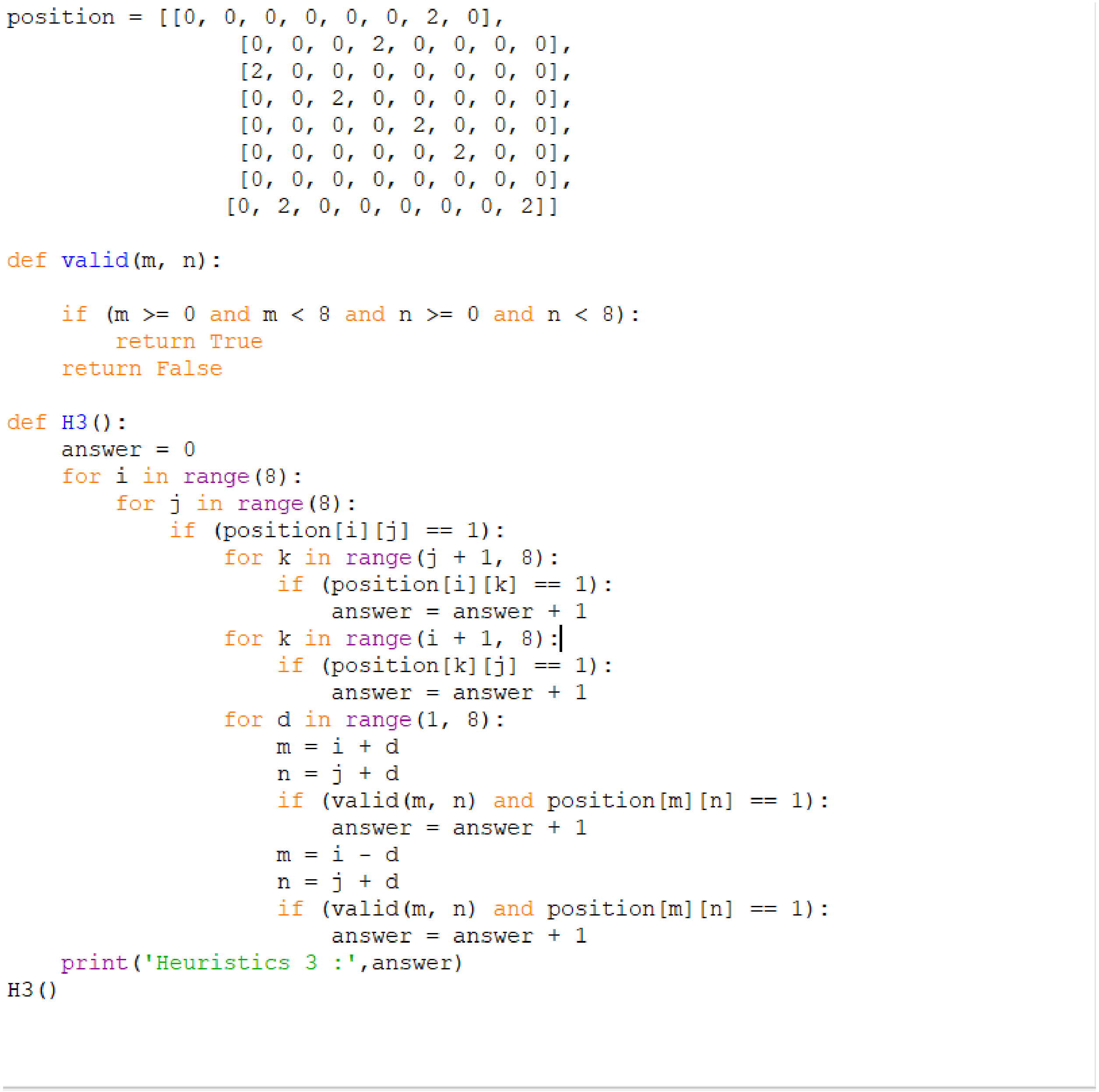
Output:



```
========================================================== RESTART: G:/4.1/LAB/AI Lab/Assignment/Assignment 03/question3python.py ==========
Heuristic 2 :  8
>>> 
```

Question 4: Write the Python code to find discussed h3.

Python Code:



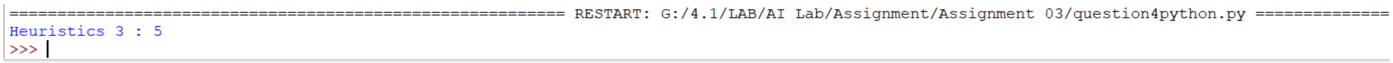
```
position = [[0, 0, 0, 0, 0, 0, 2, 0],
                 [0, 0, 0, 2, 0, 0, 0, 0],
                 [2, 0, 0, 0, 0, 0, 0, 0],
                 [0, 0, 2, 0, 0, 0, 0, 0],
                 [0, 0, 0, 0, 2, 0, 0, 0],
                 [0, 0, 0, 0, 0, 2, 0, 0],
                 [0, 0, 0, 0, 0, 0, 0, 0],
                [0, 2, 0, 0, 0, 0, 0, 2]]

def valid(m, n):

    if (m >= 0 and m < 8 and n >= 0 and n < 8):
        return True
    return False

def H3():
    answer = 0
    for i in range(8):
        for j in range(8):
            if (position[i][j] == 1):
                for k in range(j + 1, 8):
                    if (position[i][k] == 1):
                        answer = answer + 1
                for k in range(i + 1, 8):
                    if (position[k][j] == 1):
                        answer = answer + 1
                for d in range(1, 8):
                    m = i + d
                    n = j + d
                    if (valid(m, n) and position[m][n] == 1):
                        answer = answer + 1
                    m = i - d
                    n = j + d
                    if (valid(m, n) and position[m][n] == 1):
                        answer = answer + 1
    print('Heuristics 3 :',answer)
H3()
```

Output:



```
========================================================== RESTART: G:/4.1/LAB/AI Lab/Assignment/Assignment 03/question4python.py ==============
Heuristics 3 : 5
>>> |
```