Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering

Course No: CSE 4108

Course Title: Artificial Intelligence Lab

Assignment no: 02

Date of Submission: 27.06.2022

Submitted to:

Mr. Md. Siam Ansary
Lecturer, Department of CSE, AUST.

Ms. Tamanna Tabassum
Lecturer, Department of CSE, AUST.

Submitted By,

Name: S. M. Tasnimul Hasan

Student ID: 18.02.04.142

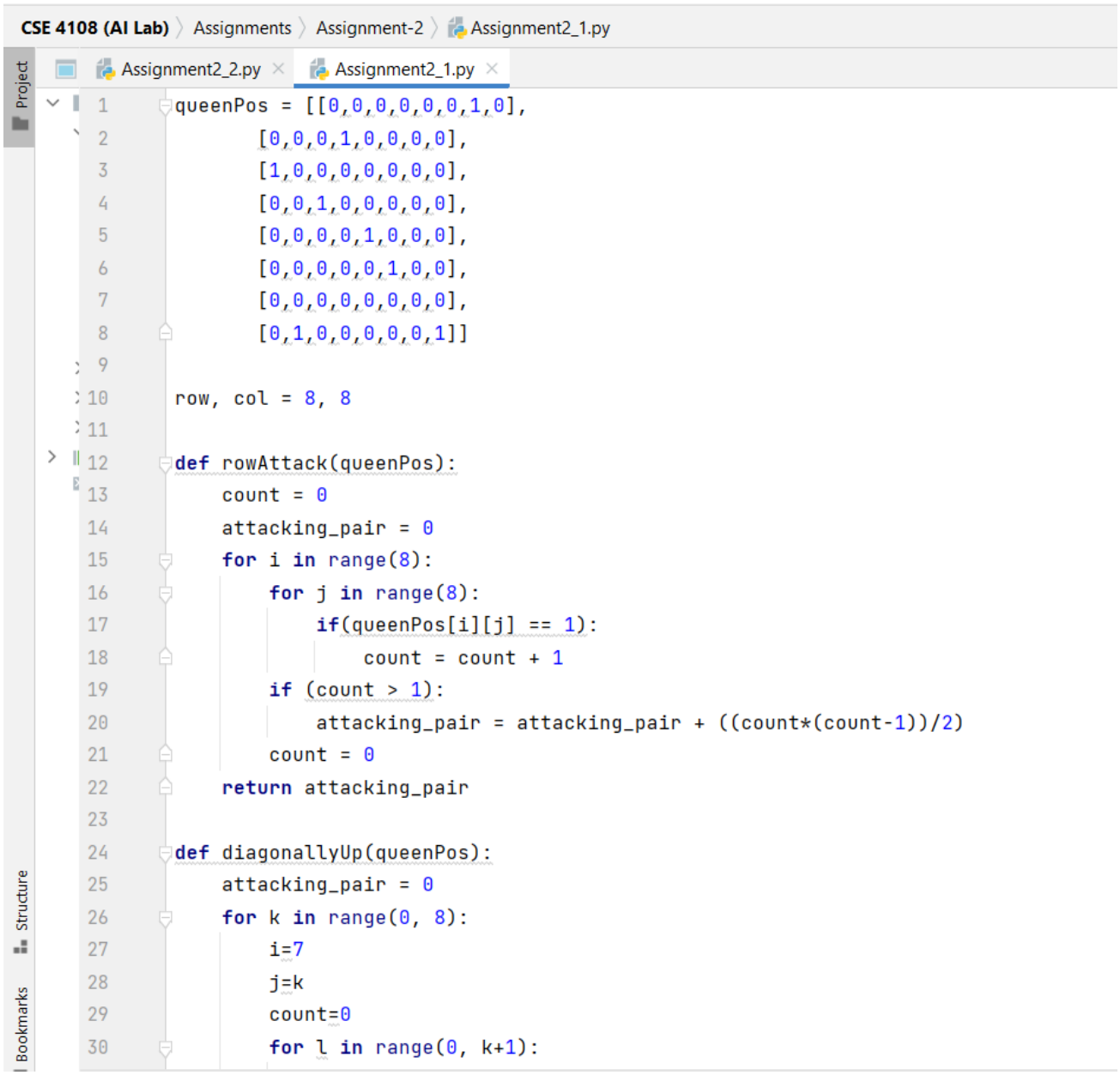
Question 1: For 8 Queens problem, heuristic can be calculated as the number of attacking pairs. Hence, heuristic, h = face to face in the row + face to face diagonally up + face to face diagonally down. Write a Python program for finding out this heuristic.

Solution:

Python Code:

```
queenPos = [[0,0,0,0,0,0,1,0],
      [0,0,0,1,0,0,0,0],
      [1,0,0,0,0,0,0,0],
      [0,0,1,0,0,0,0,0],
      [0,0,0,0,1,0,0,0],
      [0,0,0,0,0,1,0,0],
      [0,0,0,0,0,0,0,0],
      [0,1,0,0,0,0,0,1]]

row, col = 8, 8

def rowAttack(queenPos):
   count = 0
   attacking_pair = 0
   for i in range(8):
      for j in range(8):
         if(queenPos[i][j] == 1):
            count = count + 1
      if (count > 1):
         attacking_pair = attacking_pair + ((count*(count-1))/2)
      count = 0
   return attacking_pair

def diagonallyUp(queenPos):
   attacking_pair = 0
   for k in range(0, 8):
      i=7
      j=k
      count=0
      for l in range(0, k+1):
         if(queenPos[i][j] == 1):
            count = count + 1
         j=j-1
         i=i-1
      if (count > 1):
         attacking_pair = attacking_pair + ((count*(count-1))/2)

   for k in range(0, 7):
      i=k
      j=7
```

```
        count=0
        for l in range(0, k+1):
            if(queenPos[i][j] == 1):
                count = count + 1
            j=j-1
            i=i-1
        if (count > 1):
            attacking_pair = attacking_pair + ((count*(count-1))/2)
    return attacking_pair

def diagonallyDown(queenPos):
    attacking_pair = 0
    for i in range(1, (row+col)):
        start_col = max(0, i-row)
        temp = min(i, (col-start_col), row)
        count = 0
        for j in range(0, temp):
            if(queenPos[min(row, i) - j -1][start_col+j] == 1):
                count = count + 1
        if (count > 1):
            attacking_pair = ((count*(count-1))/2)
    return attacking_pair

total = rowAttack(queenPos) + diagonallyUp(queenPos) + diagonallyDown(queenPos)
print('Total Number of Attacking pairs: ' ,total)
```


CSE 4108 (AI Lab) 〉 Assignments 〉 Assignment-2 〉 Assignment2_1.py
Project
Assignment2_2.py
Assignment2_1.py
queenPos = [[0,0,0,0,0,0,1,0],
[0,0,0,1,0,0,0,0],
[1,0,0,0,0,0,0,0],
[0,0,1,0,0,0,0,0],
[0,0,0,0,1,0,0,0],
[0,0,0,0,0,1,0,0],
[0,0,0,0,0,0,0,0],
[0,1,0,0,0,0,0,1]]
row, col = 8, 8
def rowAttack(queenPos):
count = 0
attacking_pair = 0
for i in range(8):
for j in range(8):
if(queenPos[i][j] == 1):
count = count + 1
if (count > 1):
attacking_pair = attacking_pair + ((count*(count-1))/2)
count = 0
return attacking_pair
def diagonallyUp(queenPos):
attacking_pair = 0
for k in range(0, 8):
i=7
j=k
count=0
for l in range(0, k+1):
Structure
Bookmarks

```
            if(queenPos[i][j] == 1):
                count = count + 1
            j=j-1
            i=i-1
        if (count > 1):
            attacking_pair = attacking_pair + ((count*(count-1))/2)

    for k in range(0, 7):
        i=k
        j=7
        count=0
        for l in range(0, k+1):
            if(queenPos[i][j] == 1):
                count = count + 1
            j=j-1
            i=i-1
        if (count > 1):
            attacking_pair = attacking_pair + ((count*(count-1))/2)
    return attacking_pair

def diagonallyDown(queenPos):
    attacking_pair = 0
    for i in range(1, (row+col)):
        start_col = max(0, i-row)
        temp = min(i, (col-start_col), row)
        count = 0
        for j in range(0, temp):
            if(queenPos[min(row, i) - j -1][start_col+j] == 1):
                count = count + 1
        if (count > 1):
            attacking_pair = ((count*(count-1))/2)
    return attacking_pair

total = rowAttack(queenPos) + diagonallyUp(queenPos) + diagonallyDown(queenPos)
print('Total Number of Attacking pairs: ',total)
```

diagonallyDown()

Run: Assignment2_1

```
"C:\Program Files\Python310\python.exe" "E:/AUST CSE/4.1 (Origin-42)/CSE 4108 (AI Lab)/Assignments/
Total Number of Attacking pairs:  5.0

Process finished with exit code 0
```

Question 2: Write a Python program where the heuristic of 8 puzzle problem is calculated as the Manhattan distances of the tiles.

Solution:

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)
list1 = []

i = 0
h = 0
for i in range(8):
   if ((gtp[i][1] != tp[i][1]) or (gtp[i][2] != tp[i][2])):
      list1.append(abs(gtp[i][1] - tp[i][1]) + abs(gtp[i][2] - tp[i][2]))

for i in list1:
   h = h + i
print("Manhattan Distance:", h)
```