

Started on	Friday, 9 May 2025, 8:38 AM
State	Finished
Completed on	Friday, 9 May 2025, 10:46 AM
Time taken	2 hours 8 mins
Overdue	8 mins 19 secs
Grade	80.00 out of 100.00

Question 1

Not answered

Mark 0.00 out of 20.00

Write a python program to implement quick sort on the given array values.

For example:

Input	Result
5	left: []
21	right: []
40	left: []
50	right: []
30	left: []
13	right: []
	left: [30]
	right: [50]
	left: [13]
	right: [30, 40, 50]
	[13, 21, 30, 40, 50]
6	left: []
7	right: []
5	left: [4]
21	right: []
63	left: []
4	right: []
9	left: []
	right: []
	left: [9]
	right: [63]
	left: [4, 5]
	right: [9, 21, 63]
	[4, 5, 7, 9, 21, 63]

Answer: (penalty regime: 0 %)

1	
---	--

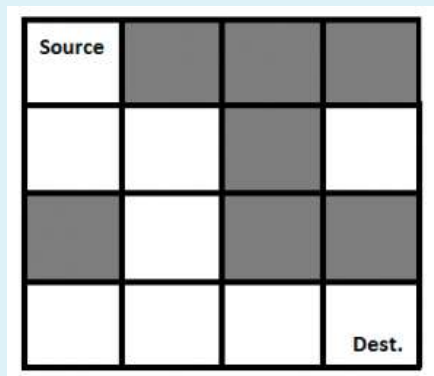
Question 2

Correct

Mark 20.00 out of 20.00

Rat In A Maze Problem

You are given a maze in the form of a matrix of size $n \times n$. Each cell is either clear or blocked denoted by 1 and 0 respectively. A rat sits at the top-left cell and there exists a block of cheese at the bottom-right cell. Both these cells are guaranteed to be clear. You need to find if the rat can get the cheese if it can move only in one of the two directions - down and right. It can't move to blocked cells.



Provide the solution for the above problem Consider $n=4$)

The output (Solution matrix) must be 4×4 matrix with value "1" which indicates the path to destination and "0" for the cell indicating the absence of the path to destination.

Answer: (penalty regime: 0 %)

Reset answer

```

1 N = 4
2
3 def printSolution( sol ):
4
5     for i in sol:
6         for j in i:
7             print(str(j) + " ", end = "")
8         print("")
9
10
11 def isSafe( maze, x, y ):
12
13     if x >= 0 and x < N and y >= 0 and y < N and maze[x][y] == 1:
14         return True
15
16     return False
17
18
19 def solveMaze( maze ):
20
21     sol = [ [ 0 for j in range(4) ] for i in range(4) ]
22

```

	Expected	Got	
✓	<pre> 1 0 0 0 1 1 0 0 0 1 0 0 0 1 1 1 </pre>	<pre> 1 0 0 0 1 1 0 0 0 1 0 0 0 1 1 1 </pre>	✓

Passed all tests! ✓



Marks for this submission: 20.00/20.00.

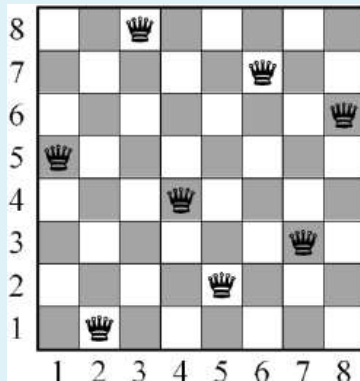
Question 3

Correct

Mark 20.00 out of 20.00

You are given an integer **N**. For a given **N x N** chessboard, find a way to place '**N**' queens such that no queen can attack any other queen on the chessboard.

A queen can be attacked when it lies in the same row, column, or the same diagonal as any of the other queens. **You have to print one such configuration.**



Note :

Get the input from the user for **N** . The value of **N** must be from 1 to 8

If solution exists Print a binary matrix as output that has 1s for the cells where queens are placed

If there is no solution to the problem print "Solution does not exist"

For example:

Input	Result
5	1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0

Answer: (penalty regime: 0 %)

Reset answer

```

1 global N
2 N = int(input())
3
4 def printSolution(board):
5     for i in range(N):
6         for j in range(N):
7             print(board[i][j], end = " ")
8         print()
9
10 def isSafe(board, row, col):
11
12     for i in range(col):
13         if board[row][i] == 1:
14             return False
15
16     for i, j in zip(range(row, -1, -1),
17                   range(col, -1, -1)):
18         if board[i][j] == 1:
19             return False
20
21     for i, j in zip(range(row, N, 1),
22                   range(col, -1, -1)):

```

	Input	Expected	Got	
✓	5	1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0	1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0	✓
✓	2	Solution does not exist	Solution does not exist	✓
✓	8	1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 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 1 0 0 0 0 0	1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 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 1 0 0 0 0 0	✓

Passed all tests! ✓



Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

SUBSET SUM PROBLEM

Given a set of positive integers, and a value sum, determine that the sum of the subset of a given set is equal to the given sum.

Write the program for [subset sum problem](#).

INPUT

- 1.no of elements
- 2.Input the given elements
- 3.Get the target sum

OUTPUT

True , if subset with required sum is found

False , if subset with required sum is not found

For example:

Input	Result
5	4
4	16
16	5
5	23
23	12
12	True,subset found
9	

Answer: (penalty regime: 0 %)

Reset answer

```

1 def SubsetSum(a,i,sum,target,n):
2     if i==n:
3         return sum==target
4     if sum>target:
5         return False
6     if sum==target:
7         return True
8
9     return SubsetSum(a,i+1,sum,target,n) or SubsetSum(a,i+1,sum+a[i],target,n);
10
11 a=[]
12 size=int(input())
13 for i in range(size):
14     x=int(input())
15     a.append(x)
16
17 target=int(input())
18 n=len(a)
19 if(SubsetSum(a,0,0,target,n)==True):
20     for i in range(size):
21         print(a[i])
22     print("True,subset found")

```

	Input	Expected	Got	
✓	5	4	4	✓
	4	16	16	
	16	5	5	
	5	23	23	
	23	12	12	
	12	True,subset found	True,subset found	
	9			

	Input	Expected	Got	
✓	4	1	1	✓
	1	2	2	
	2	3	3	
	3	4	4	
	4	False,subset not found	False,subset not found	
	11			
✓	7	10	10	✓
	10	7	7	
	7	5	5	
	5	18	18	
	18	12	12	
	12	20	20	
	20	15	15	
	15	True,subset found	True,subset found	
	35			

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question 5

Correct

Mark 20.00 out of 20.00

GRAPH COLORING PROBLEM

Given an undirected graph and a number m , determine if the graph can be coloured with at most m colours such that no two adjacent vertices of the graph are colored with the same color. Here coloring of a graph means the assignment of colors to all vertices.

Input-Output format:

Input:

1. A 2D array $graph[V][V]$ where V is the number of vertices in graph and $graph[V][V]$ is an adjacency matrix representation of the graph. A value $graph[i][j]$ is 1 if there is a direct edge from i to j , otherwise $graph[i][j]$ is 0.
2. An integer m is the maximum number of colors that can be used.

Output:

An array $color[V]$ that should have numbers from 1 to m . $color[i]$ should represent the color assigned to the i th vertex.

Example:

Input:

```
graph = {0, 1, 1, 1},
        {1, 0, 1, 0},
        {1, 1, 0, 1},
        {1, 0, 1, 0}
```

Output:

Solution Exists:

Following are the assigned colors

```
1 2 3 2
```

Explanation: By coloring the vertices with following colors, adjacent vertices does not have same colors

Input:

```
graph = {1, 1, 1, 1},
        {1, 1, 1, 1},
        {1, 1, 1, 1},
        {1, 1, 1, 1}
```

Output: Solution does not exist.

Explanation: No solution exists.

Answer: (penalty regime: 0 %)

```
1 class Graph():
2     def __init__(self, vertices):
3         self.V = vertices
4         self.graph = [[0 for column in range(vertices)] for row in range(vertices)]
5
6     def isSafe(self, v, colour, c):
7         for i in range(self.V):
8             if self.graph[v][i] == 1 and colour[i] == c:
9                 return False
10        return True
11
12    def graphColourUtil(self, m, colour, v):
13        if v==self.V:
14            return True
15        for c in range(1,m+1):
16            if self.isSafe(v,colour,c)==True:
17                colour[v]=c
18                if self.graphColourUtil(m,colour,v+1) == True:
```

```
19         return True
20         colour[v]=0
21
22     def graphColouring(self, m):
```

	Test	Expected	Got	
✓	<pre>g = Graph(4) g.graph = [[0, 1, 1, 1], [1, 0, 1, 0], [1, 1, 0, 1], [1, 0, 1, 0]] m = 3 g.graphColouring(m)</pre>	<pre>Solution exist and Following are the assigned colours: 1 2 3 2</pre>	<pre>Solution exist and Following are the assigned colours: 1 2 3 2</pre>	✓

Passed all tests! ✓

Completed

Marks for this submission: 20.00/20.00.

