Started on	Monday, 21 April 2025, 2:13 PM
State	Finished
Completed on	Thursday, 24 April 2025, 10:11 AM
Time taken	2 days 19 hours
Overdue	2 days 17 hours
Grade	**80.00** out of 100.00

Question **1**

Not answered

Mark 0.00 out of 20.00

Write a python program for the implementation of merge sort on the given list of values.

For example:

Input	Result
5 12 10 61 2 3	Given array is 12 10 61 2 3 Sorted array is 2 3 10 12 61
6 20 10 31 49 87 6	Given array is 20 10 31 49 87 6 Sorted array is 6 10 20 31 49 87

Answer: (penalty regime: 0 %)

```
1
```

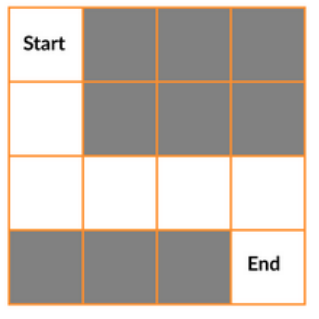
Question **2**

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 * 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

```
N = 4

def printSolution( sol ):

    for i in sol:
        for j in i:
            print(str(j) + " ", end ="")
        print("")


def isSafe( maze, x, y ):

    if x >= 0 and x < N and y >= 0 and y < N and maze[x][y] == 1:
        return True

    return False


def solveMaze( maze ):

    # Creating a 4 * 4 2-D list
    sol = [ [ 0 for j in range(4) ] for i in range(4) ]
```

	Expected	Got	
✔	1 0 0 0 1 1 0 0 0 1 0 0 0 1 1 1	1 0 0 0 1 1 0 0 0 1 0 0 0 1 1 1	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

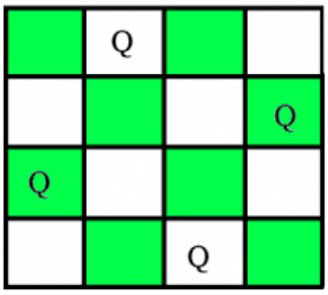
Question **3**

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 4

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
4	0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0

Answer: (penalty regime: 0 %)

Reset answer

```
global N
N = int(input())

def printSolution(board):
    for i in range(N):
        for j in range(N):
            print(board[i][j], end = " ")
        print()

def isSafe(board, row, col):

    # Check this row on left side
    for i in range(col):
        if board[row][i] == 1:
            return False

    # Check upper diagonal on left side
    for i, j in zip(range(row, -1, -1),
                    range(col, -1, -1)):
        if board[i][j] == 1:
            return False

```

	Input	Expected	Got	
✔	4	0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0	0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0	✔

	Input	Expected	Got	
✔	2	Solution does not exist	Solution does not exist	✔

Passed all tests! ✔

Correct

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

Answer: (penalty regime: 0 %)

Reset answer

```
def SubsetSum(a,i,sum,target,n):
    if i==n:
        return sum==target
    if SubsetSum(a,i+1,sum+a[i],target,n):
        return True
    if SubsetSum(a,i+1,sum,target,n):
        return True
    return False

a=[]
size=int(input())
for i in range(size):
    x=int(input())
    a.append(x)

target=int(input())
n=len(a)
if(SubsetSum(a,0,0,target,n)==True):
    for i in range(size):
        print(a[i])
    print("True,subset found")
else:
```

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

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

Passed all tests! ✔

Correct

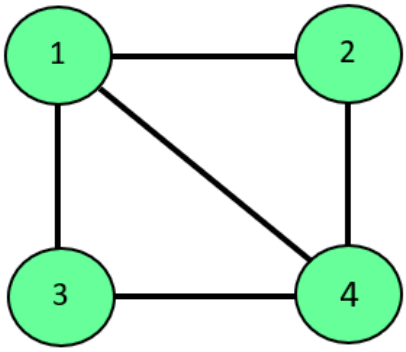
Marks for this submission: 20.00/20.00.

Question **5**

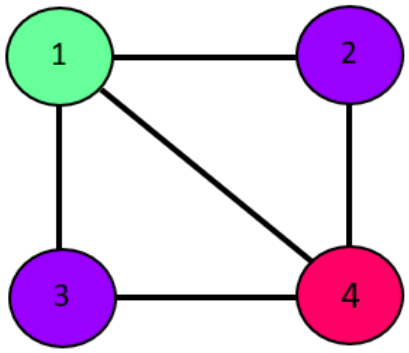
Correct

Mark 20.00 out of 20.00

The m-coloring problem states, **"We are given an undirected graph and m number of different colors. We have to check if we can assign colors to the vertices of the graphs in such a way that no two adjacent vertices have the same color."**



0	1	1	1
1	0	0	1
1	0	0	1
1	1	1	0



Node 1 -> color 1
Node 2 -> color 2
Node 3 -> color 2
Node 4-> color 3

For example:

Result
Solution Exists: Following are the assigned colors Vertex 1 is given color: 1 Vertex 2 is given color: 2 Vertex 3 is given color: 3 Vertex 4 is given color: 2

Answer: (penalty regime: 0 %)

Reset answer

```
def isSafe(graph, color):
    for i in range(4):
        for j in range(i + 1, 4):
            if graph[i][j] and color[j] == color[i]:
                return False
    return True

def graphColoring(graph, m, i, color):
    if i == 4:
        if isSafe(graph, color):
            display(color)
            return True
        return False

    for j in range(1, m + 1):
        color[i] = j
        if graphColoring(graph, m, i + 1, color):
            return True
        color[i] = 0  # Backtrack
    return False

def display(color):
```

	Expected	Got	
✔	Solution Exists: Following are the assigned colors Vertex 1 is given color: 1 Vertex 2 is given color: 2 Vertex 3 is given color: 3 Vertex 4 is given color: 2	Solution Exists: Following are the assigned colors Vertex 1 is given color: 1 Vertex 2 is given color: 2 Vertex 3 is given color: 3 Vertex 4 is given color: 2	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.