Started on	Thursday, 20 March 2025, 3:04 PM
State	Finished
Completed on	Thursday, 20 March 2025, 3:14 PM
Time taken	30 mins 41 secs
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a python program using nested loop to find the prime numbers between 2 to 100.

For example:

Result

```
2  is prime
3  is prime
5  is prime
7  is prime
11  is prime
13  is prime
17  is prime
19  is prime
23  is prime
29  is prime
31  is prime
37  is prime
41  is prime
43  is prime
47  is prime
53  is prime
59  is prime
61  is prime
67  is prime
71  is prime
73  is prime
79  is prime
83  is prime
89  is prime
97  is prime
Good bye!
```

Answer: (penalty regime: 0 %)

```python
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, num):
        if num % i == 0:
            return False
    return True

for num in range(2, 101):
    if is_prime(num):
        print(f"{num}  is prime")

print("Good bye!")
```

	Expected	Got	
✓	2 is prime 3 is prime 5 is prime 7 is prime 11 is prime 13 is prime 17 is prime 19 is prime 23 is prime 29 is prime 31 is prime 37 is prime 41 is prime 43 is prime 47 is prime 53 is prime 59 is prime 61 is prime 67 is prime 71 is prime 73 is prime 79 is prime 83 is prime 89 is prime 97 is prime Good bye!	2 is prime 3 is prime 5 is prime 7 is prime 11 is prime 13 is prime 17 is prime 19 is prime 23 is prime 29 is prime 31 is prime 37 is prime 41 is prime 43 is prime 47 is prime 53 is prime 59 is prime 61 is prime 67 is prime 71 is prime 73 is prime 79 is prime 83 is prime 89 is prime 97 is prime Good bye!	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

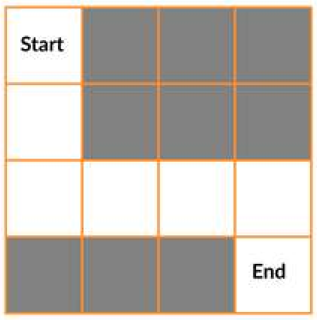
Question **2**

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 ):

```

	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.

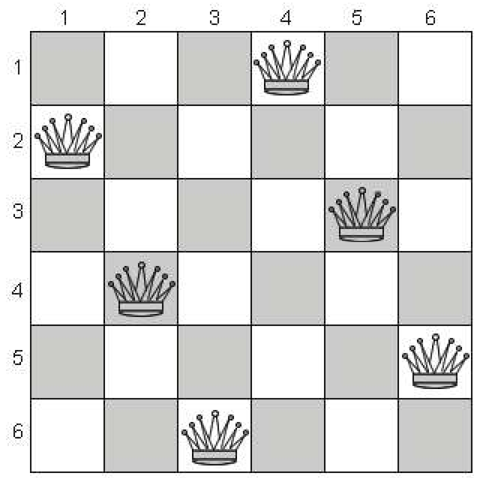
Question **3**

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 6

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

6	0 0 0 1 0 0
	1 0 0 0 0 0
	0 0 0 0 1 0
	0 1 0 0 0 0
	0 0 0 0 0 1
	0 0 1 0 0 0

Answer: (penalty regime: 0 %)

Reset answer

```python

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):

    for i in range(col):
        if board[row][i] == 1:
            return False

    for i, j in zip(range(row, -1, -1),
                    range(col, -1, -1)):
```

```
        if board[i][j] == 1:
            return False

    for i, j in zip(range(row, N, 1),
```

	Input	Expected	Got	
✓	2	Solution does not exist	Solution does not exist	✓
✓	3	Solution does not exist	Solution does not exist	✓
✓	6	0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0	0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0	✓

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



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):
```

	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	✓
✓	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**

Incorrect

Mark 0.00 out of 20.00

Greedy coloring doesn't always use the minimum number of colors possible to color a graph. For a graph of maximum degree `x`, greedy coloring will use at most `x+1` color. Greedy coloring can be arbitrarily bad;

Create a python program to implement graph colouring using Greedy algorithm.

For example:

Test	Result
colorGraph(graph, n)	Color assigned to vertex 0 is BLUE Color assigned to vertex 1 is GREEN Color assigned to vertex 2 is BLUE Color assigned to vertex 3 is RED Color assigned to vertex 4 is RED Color assigned to vertex 5 is GREEN

Answer: (penalty regime: 0 %)

Reset answer

```
class Graph:
    def __init__(self, edges, n):
        self.adjList = [[] for _ in range(n)]

        # add edges to the undirected graph
        for (src, dest) in edges:
            self.adjList[src].append(dest)
            self.adjList[dest].append(src)
def colorGraph(graph, n):
    ############ Add your code here ##############
if __name__ == '__main__':
    colors = ['', 'BLUE', 'GREEN', 'RED', 'YELLOW', 'ORANGE', 'PINK',
              'BLACK', 'BROWN', 'WHITE', 'PURPLE', 'VOILET']
    edges = [(0, 1), (0, 4), (0, 5), (4, 5), (1, 4), (1, 3), (2, 3), (2, 4)]
    n = 6
    graph = Graph(edges, n)
    colorGraph(graph, n)
```

Syntax Error(s)

```
Sorry: IndentationError: expected an indented block (__tester__.python3, line 11)
```

Incorrect

Marks for this submission: 0.00/20.00.