```python
# Name: Dinesh Rathd (TA57) - Experiment 1: DFS & BFS

from collections import deque

MAX_VERTICES = 20
adjacencyMatrix = [[0] * MAX_VERTICES for _ in range(MAX_VERTICES)]
visited = [0] * MAX_VERTICES

def depthFirstSearchRecursive(currentVertex, numVertices):
    print(currentVertex, end=" ")
    visited[currentVertex] = 1
    for i in range(1, numVertices + 1):
        if adjacencyMatrix[currentVertex][i] != 0 and visited[i] == 0:
            depthFirstSearchRecursive(i, numVertices)

def breadthFirstSearchRecursive(vertexQueue, numVertices):
    if not vertexQueue:
        return
    currentVertex = vertexQueue.popleft()
    print(currentVertex, end=" ")
    for i in range(1, numVertices + 1):
        if adjacencyMatrix[currentVertex][i] != 0 and visited[i] == 0:
            visited[i] = 1
            vertexQueue.append(i)
    breadthFirstSearchRecursive(vertexQueue, numVertices)

def main():
    numVertices = int(input("Enter the number of vertices: "))

    for i in range(1, numVertices + 1):
        for j in range(1, numVertices + 1):
            adjacencyMatrix[i][j] = int(input(f"Enter 1 if 'Node {i}' has an edge with 'Node {j}',
else enter 0: "))

    print("Adjacency Matrix:")
    for i in range(1, numVertices + 1):
```

```
        for j in range(1, numVertices + 1):
            print(adjacencyMatrix[i][j], end=" ")
        print()

    while True:
        for i in range(1, numVertices + 1):
            visited[i] = 0

        print("\nMENU")
        print("1. Depth First Search (DFS)")
        print("2. Breadth First Search (BFS)")
        choice = int(input("Enter your choice: "))
        startVertex = int(input("Enter the Source Vertex: "))

        if choice == 1:
            depthFirstSearchRecursive(startVertex, numVertices)
        elif choice == 2:
            vertexQueue = deque([startVertex])
            visited[startVertex] = 1
            breadthFirstSearchRecursive(vertexQueue, numVertices)

        cont = input("\nDO YOU WANT TO CONTINUE (Y/N)? ").strip().lower()
        if cont != 'y':
            break

if __name__ == "__main__":
    main()
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS F:\5th Sem\AI\Practicales> & "C:/Program Files/Python311/python.exe" "f:/5th Sem/AI/Practicales/AI.py"
Enter the number of vertices: 4
Enter 1 if 'Node 1' has an edge with 'Node 1', else enter 0: 1
Enter 1 if 'Node 1' has an edge with 'Node 2', else enter 0: 4
Enter 1 if 'Node 1' has an edge with 'Node 3', else enter 0: 6
Enter 1 if 'Node 1' has an edge with 'Node 4', else enter 0: 2
Enter 1 if 'Node 2' has an edge with 'Node 1', else enter 0: 2
Enter 1 if 'Node 2' has an edge with 'Node 2', else enter 0: 15
Enter 1 if 'Node 2' has an edge with 'Node 3', else enter 0: 3
Enter 1 if 'Node 2' has an edge with 'Node 4', else enter 0: 1
Enter 1 if 'Node 3' has an edge with 'Node 1', else enter 0: 6
Enter 1 if 'Node 3' has an edge with 'Node 2', else enter 0: 7
Enter 1 if 'Node 3' has an edge with 'Node 3', else enter 0: 8
Enter 1 if 'Node 3' has an edge with 'Node 4', else enter 0: 9
Enter 1 if 'Node 4' has an edge with 'Node 1', else enter 0: 0
Enter 1 if 'Node 4' has an edge with 'Node 2', else enter 0: 4
Enter 1 if 'Node 4' has an edge with 'Node 3', else enter 0: 2
Enter 1 if 'Node 4' has an edge with 'Node 4', else enter 0: 1
Adjacency Matrix:
1 4 6 2
2 15 3 1
6 7 8 9
0 4 2 1

MENU
1. Depth First Search (DFS)
2. Breadth First Search (BFS)
Enter your choice: 1
Enter the Source Vertex: 1
1 2 3 4
DO YOU WANT TO CONTINUE (Y/N)? n
PS F:\5th Sem\AI\Practicales>
```