Name: Faraz Pathan

Rollno: COTB70

class Graph:

    def \_\_init\_\_(self):

        self.graph = {}

    def add\_edge(self, u, v):

        if u in self.graph:

            self.graph[u].append(v)

        else:

            self.graph[u] = [v]

    def bfs(self, start):

        visited = {}

        for vertex in self.graph:

            visited[vertex] = False

        queue = []

        queue.append(start)

        visited[start] = True

        while queue:

            start = queue.pop(0)

            print(start, end=" ")

            for i in self.graph.get(start, []):

                if not visited[i]:

                    queue.append(i)

                    visited[i] = True

    def dfs\_util(self, v, visited):

        visited[v] = True

        print(v, end=" ")

        for i in self.graph.get(v, []):

            if not visited[i]:

                self.dfs\_util(i, visited)

    def dfs(self, start):

        visited = {}

        for vertex in self.graph:

            visited[vertex] = False

        self.dfs\_util(start, visited)

# Example usage:

g = Graph()

g.add\_edge(0, 1)

g.add\_edge(0, 2)

g.add\_edge(1, 2)

g.add\_edge(2, 0)

g.add\_edge(2, 3)

g.add\_edge(3, 3)

print("BFS starting from vertex 2:")

g.bfs(2)

print("\nDFS starting from vertex 2:")

g.dfs(2)

**OUTPUT:**

BFS starting from vertex 2:

2 0 3 1

DFS starting from vertex 2:

2 0 1 3