AI-1

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 dfs\_util(self, vertex, visited):

visited.add(vertex)

print(vertex, end=' ')

for neighbor in self.graph.get(vertex, []):

if neighbor not in visited:

self.dfs\_util(neighbor, visited)

def dfs(self, start):

visited = set()

self.dfs\_util(start, visited)

def bfs(self, start):

visited = set()

queue = [start]

while queue:

vertex = queue.pop(0)

if vertex not in visited:

print(vertex, end=' ')

visited.add(vertex)

for neighbor in self.graph.get(vertex, []):

if neighbor not in visited:

queue.append(neighbor)

# 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("DFS:")

g.dfs(2)

print("\nBFS:")

g.bfs(2)

O/P

DFS:

2 0 1 3

BFS:

2 0 3 1