**2.Solve any problem using depth first search**

**Aim:** Problem using depth first search

**Code:**

class Graph:

def \_\_init\_\_(self, vertices):

self.V = vertices

self.graph = {}

def add\_edge(self, u, v):

if u not in self.graph:

self.graph[u] = []

self.graph[u].append(v)

def dfs\_util(self, v, visited):

visited.add(v)

print(v, end=' ')

if v in self.graph:

for neighbor in self.graph[v]:

if neighbor not in visited:

self.dfs\_util(neighbor, visited)

def dfs(self, start\_node):

visited = set()

self.dfs\_util(start\_node, visited)

g = Graph(4)

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("Following is Depth First Traversal (starting from vertex 2):")

g.dfs(2)

print()

**OUT PUT:**

**Following is Depth First Traversal (starting from vertex 2):**

**2 0 1 3**