8. Write the python program to implement DFS.

**AIM :** program to implement DFS.

**ALGORITHM :**

1. Start at the initial vertex.
2. Mark the initial vertex as visited.
3. Print the value of the visited vertex.
4. For each neighbor of the current vertex that has not been visited, recursively visit that neighbor.
5. Repeat steps 3 and 4 until all reachable vertices have been visited.

**PROGRAM :**

class Graph:

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

self.vertices = vertices

self.adjacency\_list = {}

for vertex in range(vertices):

self.adjacency\_list[vertex] = []

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

self.adjacency\_list[u].append(v)

self.adjacency\_list[v].append(u)

def dfs(self, start\_vertex):

visited = [False] \* self.vertices

self.\_dfs\_recursive(start\_vertex, visited)

def \_dfs\_recursive(self, vertex, visited):

visited[vertex] = True

print(vertex, end=' ')

for neighbor in self.adjacency\_list[vertex]:

if not visited[neighbor]:

self.\_dfs\_recursive(neighbor, visited)

g = Graph(7)

g.add\_edge(0, 1)

g.add\_edge(0, 2)

g.add\_edge(1, 3)

g.add\_edge(1, 4)

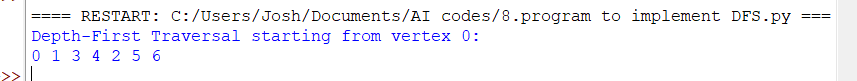
g.add\_edge(2, 5)

g.add\_edge(2, 6)

print("Depth-First Traversal starting from vertex 0:")

g.dfs(0)

**OUT PUT :**

****