Marwadi University	Marwadi University Faculty of Technology Department of Information and Communication Technology

Subject: DAA (01CT0512)	**AIM: Depth First Search**	
Experiment No: 21	**Date: 3/10/2023**	**Enrolment No: 92100133020**

Depth First Search:

DFS explores all the vertices of a graph by going as deep as possible along each branch before backtracking.

Algorithm:

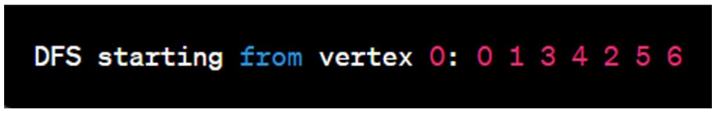
1. Start from the source vertex, mark it as visited, and explore as deep as possible along each branch.
2. When a dead end is reached, backtrack to the nearest unvisited vertex and continue exploration.

Code:

```
#include <iostream>
#include <vector>
using namespace std;

class Graph {
  int V;
  vector<int> *adj;

  void DFSUtil(int v, bool visited[]) {
    visited[v] = true;
    cout << v << " ";
    for (int i : adj[v]) {
      if (!visited[i]) {
        DFSUtil(i, visited);
      }
    }
  }

public:
  Graph(int V) {
    this->V = V;
    adj = new vector<int>[V];
  }

  void addEdge(int v, int w) {
    adj[v].push_back(w);
  }

  void DFS(int v) {
    bool *visited = new bool[V];
    fill(visited, visited + V, false);
```

```
        cout << "DFS starting from vertex " << v << ": ";
        DFSUtil(v, visited);
    }
};

int main() {
    Graph g(7);
    g.addEdge(0, 1);
    g.addEdge(0, 2);
    g.addEdge(1, 3);
    g.addEdge(1, 4);
    g.addEdge(2, 5);
    g.addEdge(2, 6);
    g.DFS(0);
    return 0;
}
```

Output:



```
DFS starting from vertex 0: 0 1 3 4 2 5 6
```

Space complexity: __________

Justification:____
____

Time complexity:

Best case time complexity: __________

Justification:____
____

Worst case time complexity: __________

Justification:____