**Practical – 1**

AIM: Implementation of Graph and Searching (DFS and BFS).  
  
#include <iostream>

#include <vector>

#include <queue>

using namespace std;

class graph

{

public:

int vertices;

vector<bool> visitedListdfs;

vector<vector<int>> adj;

graph(int size)

{

// must assign size ..

adj.resize(size);

visitedListdfs.resize(size);

vertices = size;

}

int BFS(int x);

int DFS(int x);

void addedge(int a, int b);

};

void graph::addedge(int a, int b)

{

adj[a].push\_back(b);

adj[b].push\_back(a);

}

int graph::BFS(int x)

{

vector<bool> visitedList;

visitedList.resize(vertices, false);

queue<int> queue;

// get queue front and then push thier sub child in queue untill queue is not empty..

queue.push(x);

visitedList[x] = true;

while (queue.size() > 0)

{

int x = queue.front();

cout << x << " ";

queue.pop();

for (auto node : adj[x])

{

if (visitedList[node] == false)

{

visitedList[node] = true;

queue.push(node);

}

}

}

cout << endl;

}

int graph::DFS(int x)

{

cout << visitedListdfs[x];

visitedListdfs[x] = true;

cout << x << " ";

for (auto i : adj[x])

{

if (!visitedListdfs[i])

{

DFS(i);

}

}

}

int main()

{

graph obj(8);

cout << "Graph" << endl;

cout << "Graph 1" << endl;

obj.addedge(0, 1);

obj.addedge(0, 2);

obj.addedge(1, 3);

obj.addedge(1, 4);

obj.addedge(2, 5);

obj.addedge(2, 6);

cout << "----------BFS-----------" << endl;

obj.BFS(0);

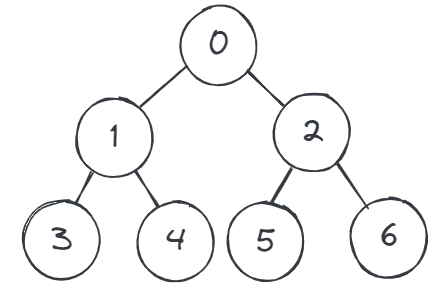
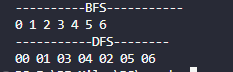
cout << "-----------DFS--------" << endl;

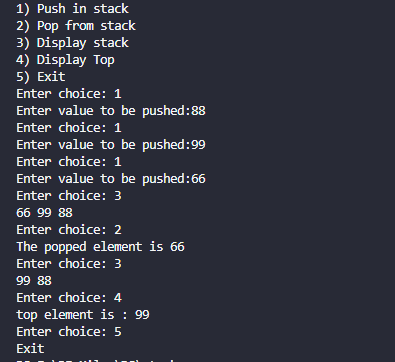
obj.DFS(0);

}

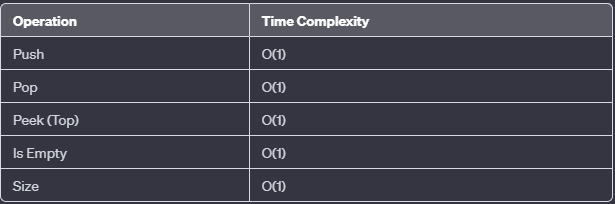
**OUTPUT**

visualize





Time analysis



Applications

* Call center systems.
* Function Call Management
* Expression Evaluation
* Backtracking Algorithms
* Undo Mechanisms in Software
* Memory Management
* Parsing and Syntax Analysis
* Task Scheduling
* in Algorithmic Problems
* The history of a web browser is stored in the form of a stack.
* Call logs, E-mails, and Google photos in any gallery are also stored in form of a stack.
* YouTube downloads and Notifications are also shown in LIFO format(the latest appears first ).
* Allocation of memory by an operating system while executing a process.