**Assignment No 5**

**Title:**

Represent a given graph using adjacency list /matrix to perform DFS and using adjacency list to perform BFS. Use the map of the area around the college as the graph. Identify the prominent landmarks as nodes and perform DFS and BFS on that.

**Objectives:**

1. To understand concept of Graph traversal

2. To understand concept Breadth first search and Depth first search..

**Theory:**

Graphs are the most general data structure. They are also commonly used data structures.

A non-linear data structure consisting of nodes and links between nodes.

**Representing Graphs using Adjacency Lists**

**Definition**:

* A directed graph with *n* vertices can be represented by *n* different linked lists.
* List number *i* provides the connections for vertex *i*.
* For each entry *j* in list number *i*, there is an edge from *i* to *j*.

**Breadth First Search:**BFS stands for [Breadth First Search](https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/) is a vertex-based technique for finding the shortest path in the graph. It uses a [Queue data structure](https://www.geeksforgeeks.org/queue-data-structure/) that follows first in first out. In BFS, one vertex is selected at a time when it is visited and marked then its adjacent are visited and stored in the queue. It is slower than DFS.   
Example:

* **Input:**

A

/ \

B C

/ / \

D E F

* **Output:**

A, B, C, D, E, F

**Depth First Search:**  
DFS stands for [Depth First Search](https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/) is an edge-based technique. It uses the [Stack data structure](https://www.geeksforgeeks.org/stack-data-structure/) and performs two stages, first visited vertices are pushed into the stack, and second if there are no vertices then visited vertices are popped.   
Example:

**Input:**

A

/ \

B C

/ / \

D E F

**Output:**

A, B, D, C, E, F

**Algorithms:**

BFS Algorithm:

1. Start by putting any one of the graph's vertices at the back of a queue.
2. Take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the back of the queue.
4. Keep repeating steps 2 and 3 until the queue is empty.

The DFS algorithm

1. Start by putting any one of the graph's vertices on top of a stack.
2. Take the top item of the stack and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the top of the stack.
4. Keep repeating steps 2 and 3 until the stack is empty.

**Conclusion:** This program gives us the knowledge about using stack and queue data structures for graph traversals.