Assignment - 6

Date

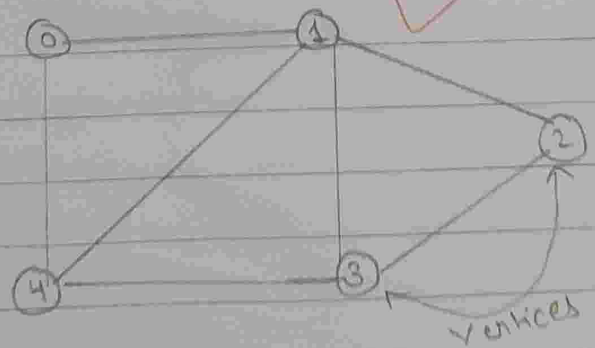
Title: Represent a given graph using adjacency matrix list to perform DFS and using adjacency list to perform BFS. Use the map of the area around the college as the graph. Identify the prominent land marks as nodes and perform DFS and BFS on that.

Objective:-

1. To identify directed & undirected graph
2. To represent graph using adjacency matrix & list
3. To transverse program to the graph.

Theory:

A graph is a non-linear data structure consisting of nodes edges. The nodes are sometimes also referred to a vertices and the edges are lines or areas that connect any two nodes in graph consists of a finite set of vertices (or nodes) and set of Edges which connect a pair of nodes.



A graph is a data structure that consists of a following two components:

1. A finite set of vertices also called as nodes.

2. A finite set of ordered pair of the form (u, v) called as edge. The pair is ordered because (u, v) is not the same as (v, u) in case of directed graph (di-graph). The pair of the form (u, v) indicates that there is an edge from vertex u to vertex v.

The edges may contain weight / value / cost

The following two are the most commonly used representation of a graph

1. Adjacency matrix
2. Adjacency list

There are other representations also like, Incidence matrix and incidence list. The choice of graph representation situation specific. It totally depends on the type of operations to be performed and ease of use.

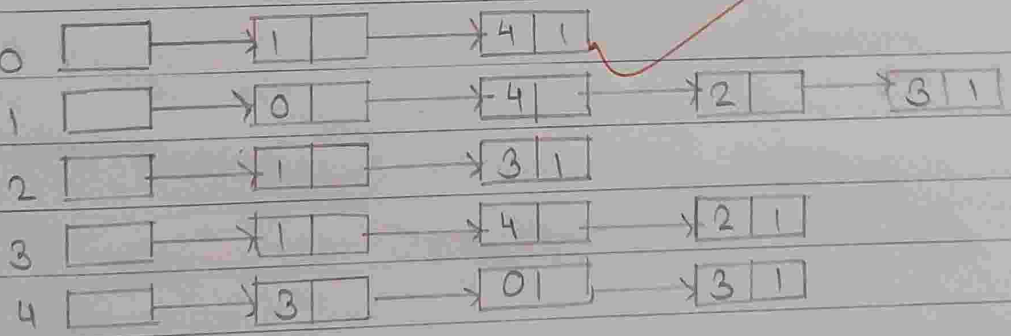
Adjacency Matrix:

Adjacency matrix is a 2D array of a size V x V where V is the number of vertices in a graph. Let the 2D array be adj[][], a slot adj[i][j]=1 indicates that there is an edge from vertex i to the vertex j. Adjacency matrix for undirected graph is always symmetric. Adjacency matrix is also used to represent weighted graphs. If adj[i][j]=W, then there is an edge from vertex i to vertex j with weight W.

	0	1	2	3	4
0	0	1	0	0	1
1	1	0	1	1	1
2	0	1	0	1	0
3	0	1	1	0	1
4	1	1	0	1	0

Adjacency list:

An array of lists is used. The size of the array is equal to number of vertices. Let the array be an array[]. An entry array[i] represents the list of vertices adjacent to the ith vertex. This representation can also be used to represent a weighted graph. The weights of edges can be represented as lists of pairs. Following is the adjacency list representation of the above graph.



Breadth first search or BFS is a graph transversal algorithm.

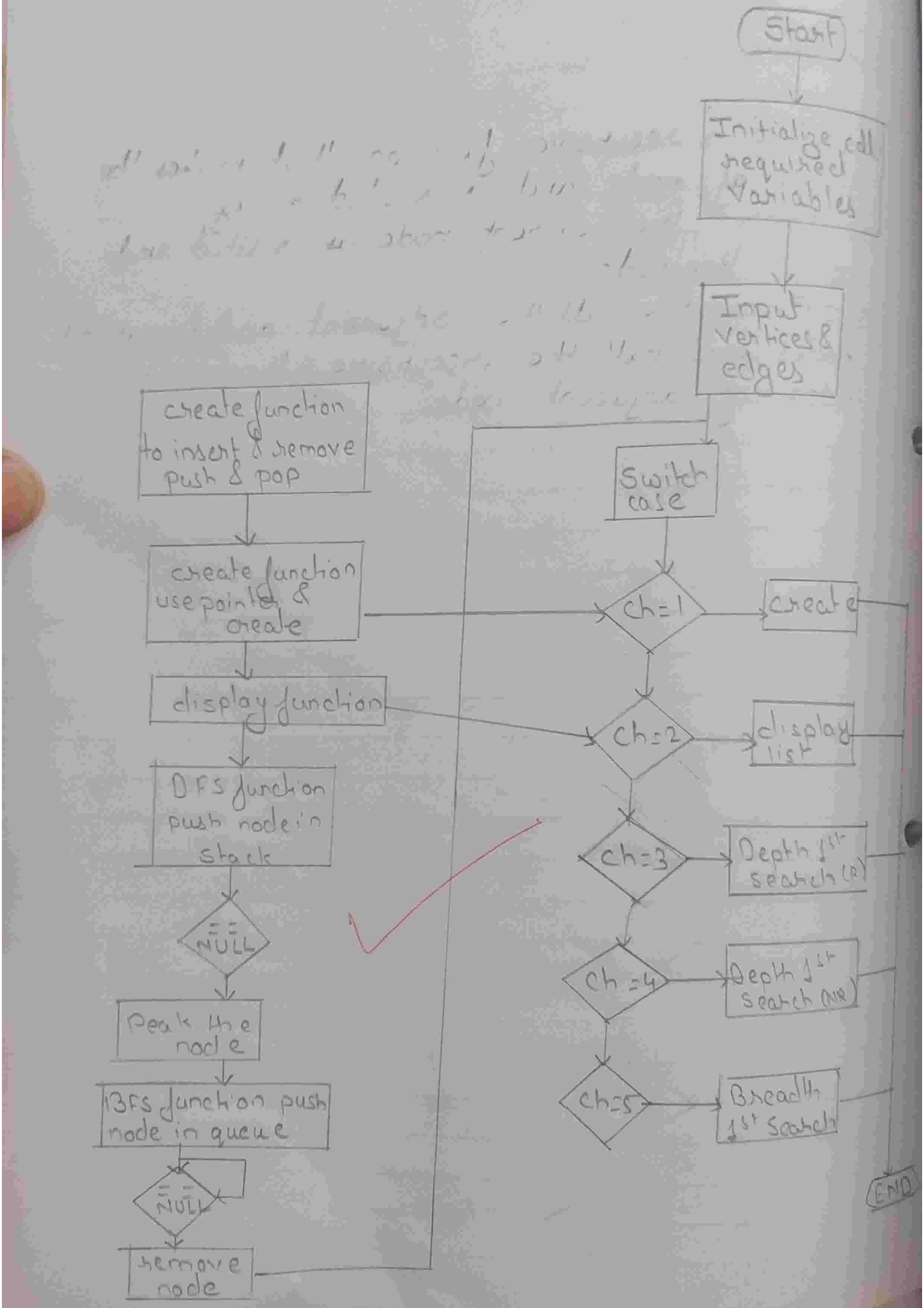
- It is used for transversing or searching a graph in a stematic fashion.
- BFS uses a strategy that searching a graphin

breadth first search.

Breadth first transversal (or search) for a graph is similar to bredth first transversal of a tree. Graphs may contain cycles, so we may come to the same node again. To avoid processing a node more than once, we use a boolean visited array. For simplicity, it is assumed that all vertices are rechable from the strong vertex.

Algorithm

1. Create a recursive function that makes the index of node and a visited array.
2. Mark the current node as visited and print the node.
3. Transverse all the adjacent and unmarked nodes and call the recursive function with index of adjacent node.


Start
Initialize all required Variables
Input vertices & edges
Switch case
ch=1
create
ch=2
display list
ch=3
Depth 1st search (R)
ch=4
Depth 1st search (NR)
ch=5
Breadth 1st Search
END
create function to insert & remove push & pop
create function use pointer & create
display function
DFS function push node in stack
== NULL
Peak the node
BFS function push node in queue
== NULL
remove node