Tutorial-5

Ans1]

BFS

- •) It stands for Breadth First Search.
- •) It uses Queue data structure.
- •) It is more suitable for searching vertices which are closer to given source.
- •) BFS considers all neighbours first of therefore not suitable for decision
- •) Here siblings are visited before children.
- •) There is no concept of bracktraching.

DFS

- •) It stands for Depth First search.
- •) It uses stack data structure.
- •) It is more suitable when there are solutions away from source.
- •)DFS is more suitable for game puzzle problems. We make a decision, then explore all paths through this decision. And if decision leads to win situation we stop.
- •) Here children are visited before siblings.
- •) It is a recursive algorith that uses backtraching.

Applications:-

•) BFS → Bipartite graph and shortest path, peer to peer networking, circular in search engine of GPs navigation system.

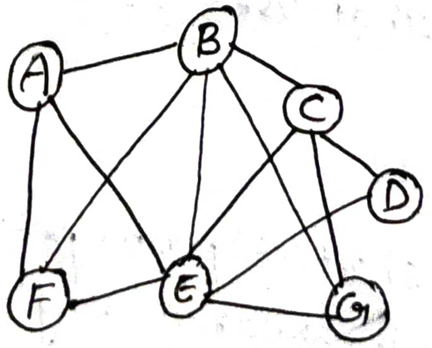
•) DFS → acyclic graph, topological order, scheduling problems, sundahu puzzle.

Ans2] For implementing BFS we need a queue data structure for finding shortest path between any node. BFS searches for nodes levelwise, i.e search nodes w.r.t their distance from root (source). For this queue in letter to use in BFS.

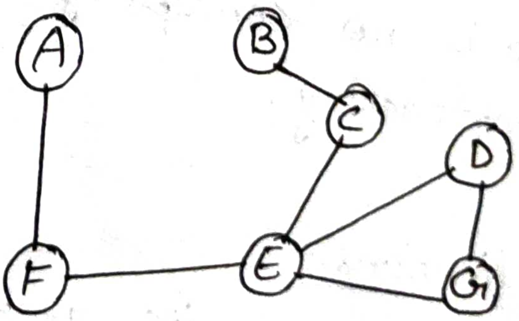
For implementing DFS we need a stack data structure as it traverse a graph in depth motion and uses stack to remenless to get the next vertex to start a search, when a dead end occurs in any iteration.

Ans3] Dense graph is graph in which no. of edges is close to moximal no. of edges.

Space graph is graph in which no. of edges is less.



Dense Graph
(many edges b/w nodes)



Sparse graph
(few edges b/w nodes)

Ans 4]

For detecting cycles in a grafph using BFS we need to use kahn's algorithm for Topological sorting:-

The steps involved are:-

(1) Compute in-degree (no. of incoming edges)

(2) Pick all vertices with-in degree as 0 and add them in queue.

(3) Rename a vertex from queue and then

- increment count of visited nodes by 1
- Decrease in-degree by 1 for all its neighbouring nodes.
- If in-degree of neigbouring nodes is reduced to zero then add to queue.

(4) Repeat 3) until queue is empty.

5) If count is visited nodes is not equal to no. of nodes in grafph, has cycle, otherwise not.

The detecting cycle in grafph using DFS we need to do following:

DFS for a connected grafph produces a tree. There is cycle in graph produces a tree. There is a cycle in graph if there is back edge present in the grafph. To detect cycle, we check for a cycle in individual tree by checking back edges. To detect a back edge, keep track of vertices.

Ans5] A disjoint set is a data structure that keeps track of set of elements partioned into several disjoint subsets. In other words, a disjoint set is a group of sets where no item can be in more than one set.

3 operations:-

•) Find ⟶ can be implemented by recursively traversing the parent array until we hit a node who is parent to itself.

```
int find (int i)
    if (parent[i] == i) {
        return i;

    }
   else {
   return find (parent[i])
   }
}
```

•) Union → It takes 2 elements as input and find representatives of this sets using the find operation and finally puts either one of the trees under node of other tree, effectively merging the trees and sets.

eg-

```
void union (int i, int j){
   int irep = this.find (i);
   int jrep = this.find(j);
   this.parent [irep] = jrep;
   }
```

·) Union by Rank →

We need a new array rank [ ].

If we are visiting 2 trees, we call them left and right, then it all depends on rank of left and right.

```
void union (int i, int j) {
        int irep = this.Find(i);
         int jrep = this.find(j);
         if (irep == jrep) return;
            irank = Rank[irep];
         jrank.Rank[jrep]
           if (irank < jrank)
           this. parent[irep] = jrep
         else if (jrank < irank)
            this. parent [jrep] = irep;
         else
             this.parent[irep] = jrep;
                Rank [jrep]++;
              }
         }
```

Ans6)

BFS

child	G	H	D	F	C	E	A	B
Parent		G	G	G	H	C	E	A

Path → G → H → C → E → A → B

DFS

~~G~~
D
H
~~F~~
~~C~~
~~E~~
~~A~~
B
Nodes Visited

G
F
C
E
A
B
STACK

Path → G → F → C → E → A → B

Ans7) V = {a} {b} {c} {d} {e} {f} {g} {h} {i} {j}

E = {a, b}, {a, c}, {b, c}, {b, d}, {e, f}, {e, g}, {h, i}, {j}

(a,b)	{a, b} {c} {d} {e} {f} {g} {h} {i} {j}
(a,c)	{a, b, c} {d} {e} {f} {g} {h} {i} {j}
(b,c)	{a, b, c} {d} {e} {f} {g} {h} {i} {j}
(b,d)	{a, b, c, d} {e} {f} {g} {h} {i} {j}
(e,f)	{a, b, c, d} {e, f} {g} {h} {i} {j}
(e,g)	{a, b, c, d} {e, f, g} {h} {j}
(h,i)	{a b c d} {e, f, g} {h, i} {j}

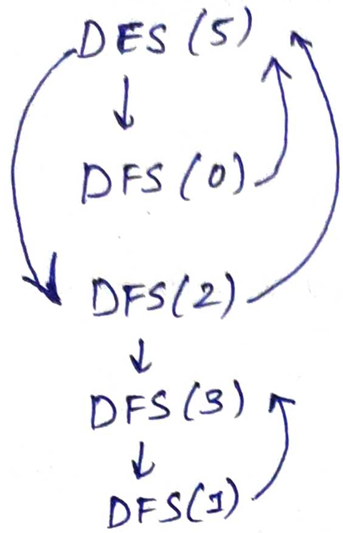
No. of connected components = 3.

Ans//

Ans8)

We take source node as 5

Applying Topological Sort



DFS (4)
↓
Not possible

q: 5/4, Pop 5 of decrement indegree of it by 1

q: 4/2; Pop 4 of decrement indegree push 0.

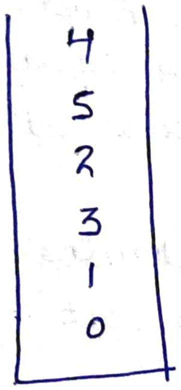
q: 2/0 Pop 2 of decrement indegree push 3

q: 0/3 Pop 0, Pop 3
Push 1

q: 1, Pop 1

Answer: 5 4 2 0 3 1,
Topological Sort

DFS



Stack 4 → 5 → 2 → 3 → 1 → 0

Ans9

Yes, heap data structure can be used to implement priority queue. It will take O(logN) time to insert and delete each element in priority queue. Based on heap structure, priority queue has two types max-priority queue based on max-heap and min-priority queue based on min-heap.

Prim's Minimum spanning Tree use Priority Queue:-

- Dijkstra's Algorithm -

When graph is stored in form of adjacency list or matrix, priority queue is used to extract minimum efficiently.

- Prim's Algorithm -

used to store keys of nodes and extract minimum key node at every step.

Ans10)

Min-Heap	Max-Heap
1) In min-heap, key present at root node must be less than or equal to among keys present at all of its children.	In max-heap the key present at root must be greater than or equal to among keys present at all of its children.
2) Minimum key element is present at the root.	Maximum key element is present at the root.
3) Uses ascending priority.	uses descending priority.
4) The smallest element has priority while construction of min-heap.	The largest element has priority, while construction of max-heap.
5) The smallest element is the first to be popped from the heap.	The largest element is the first to be popped from the heap.