Shashwat Shah
CST-SPL-1 39
2017552

Assignment-3

PAGE NO.:
DATE: / /

(1.) DFS	BFS
DFS stands for Depth First Search is an edge based technique.	BFS stands for Breadth First Search is vertex based technique for finding shortest path in graph.
It uses Stack data data Structure.	It uses queue data Structure.
=	
BFS consider all neighbours first and therefore not suitable for decision making trees used in game or puzzles.	DFS is more suitable ~~for~~ ~~decision~~ ~~making~~ ~~trees~~ ~~used~~ ~~in~~ ~~game~~ ~~or~~ ~~puzzles~~ for game or puzzle problems.
Here, Siblings are visited befor the children.	Here, Children are visited before the Siblings.
In BFS, there is no concept of backtracking	DFS algorithm is recursive algorithm that uses the idea of backtracking.
Requires more memory	Require less memory.
Application → Bipartite & Shortest path	→ Acyclic & Topological Order.

PAGE NO.:
DATE: / /

2. Breadth first Search uses queue data structure because. ito remembel ito get itne next vertex ito stait a seaich, when a dead end occurs in any iteration.

Depth first Search use Stack data Structure because ito ~~remb~~ remember ito get ithe next vertex ito stait a seaich, when a dead end occurs in any iteration.

3. In a dense graph, every pair of vertices is connected by one edge.

The Spause graph is completely ithe opposite if a graph has only a few edges (the number of edges is close ito the maximum number of edges), then it is a Spause graph.

4. Steps involved in detecting cycle in a directed graph using BFS:

Step1: Compute in-degree (number of incoming edges) for each of the vertex present in the graph and initialize the count of visited nodes as 0.

Step2: Pick all vertices with in-degree as 0 and add them into a queue.

PAGE NO.:
DATE: / /

Step3: Remove a vertex from the queue and then

(1) Increment count of visited nodes by 1.

(2) Decrease in degree by 1 for all its neighbouring nodes.

(3) If in-degree of a neighbouring nodes is reduced to zero, then add it to the queue.

Step4: Repeat Step 3 until the queue is empty.

Step5: If count of visited nodes is not equal to the number of nodes in the graph has cycle, otherwise not.

Using DFS

(1) Create the graph using the given number of edges and vertices.

(2) Create a recursive function that initialize the current index or vertex, visited, & recursion stack.

(3) Mark the current node as visited and also mark the index in recursion stack

PAGE NO.:
DATE: / /

(4) Find all the vertices which are not visited and are adjacent to the current node. Recursively call the function for those vertices, if the recursive function returns true, return true.

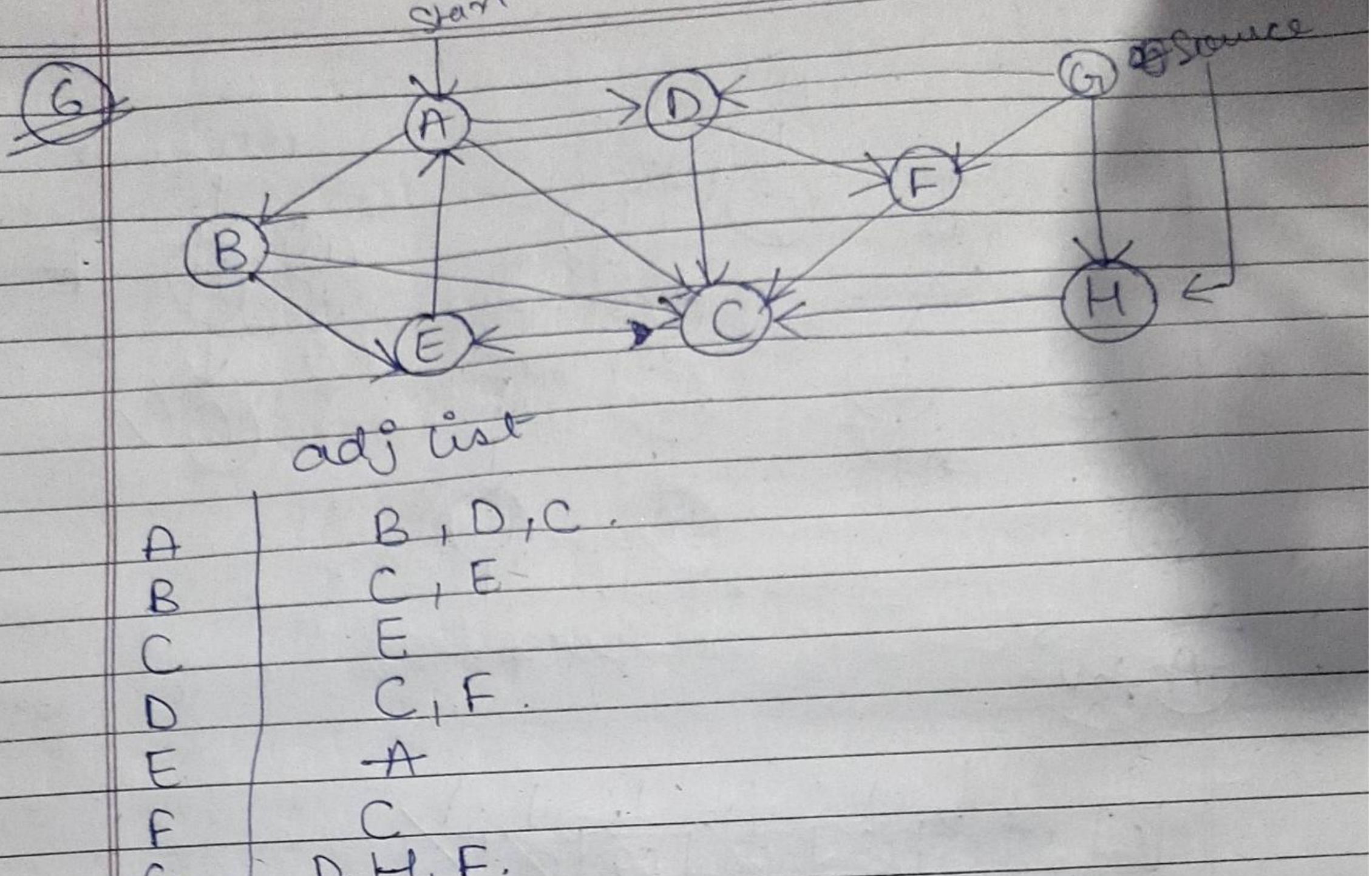
(5) If the adjacent vertices are already marked in the recursion stack then return true, Return true.

(6) Create a wrapper class, that calls the recursive function for all the vertices and if any function returns true return true else if for all vertices the function return false - Return false.

5. The disjoint set data-structure is also known as union-find data structure and merge-find set. It is a data structure that contains a collection of disjoint or non-overlapping sets.

The disjoint set means that when the set is partitioned into the disjoint subsets.

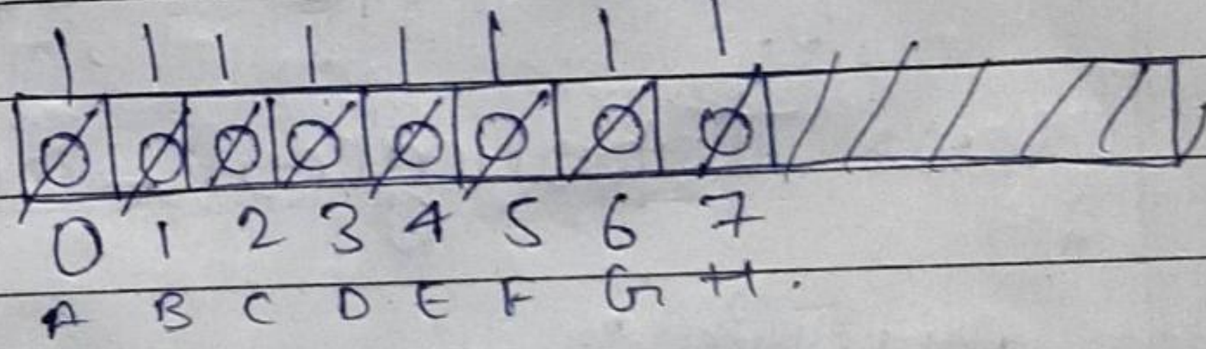
PAGE NO. :
DATE : / /

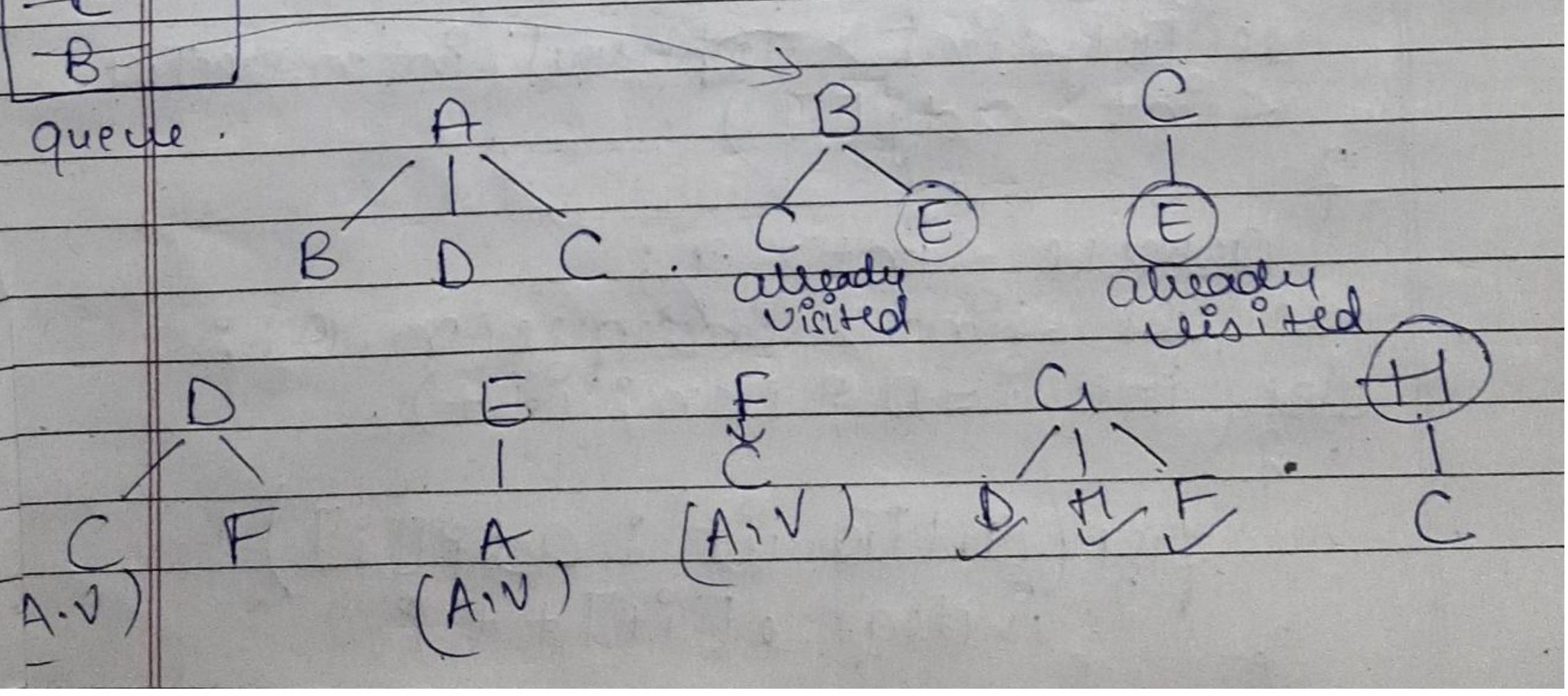


adj list

A	B, D, C.
B	C, E
C	E
D	C, F.
E	A
F	C
G	D, H, F.
H	C

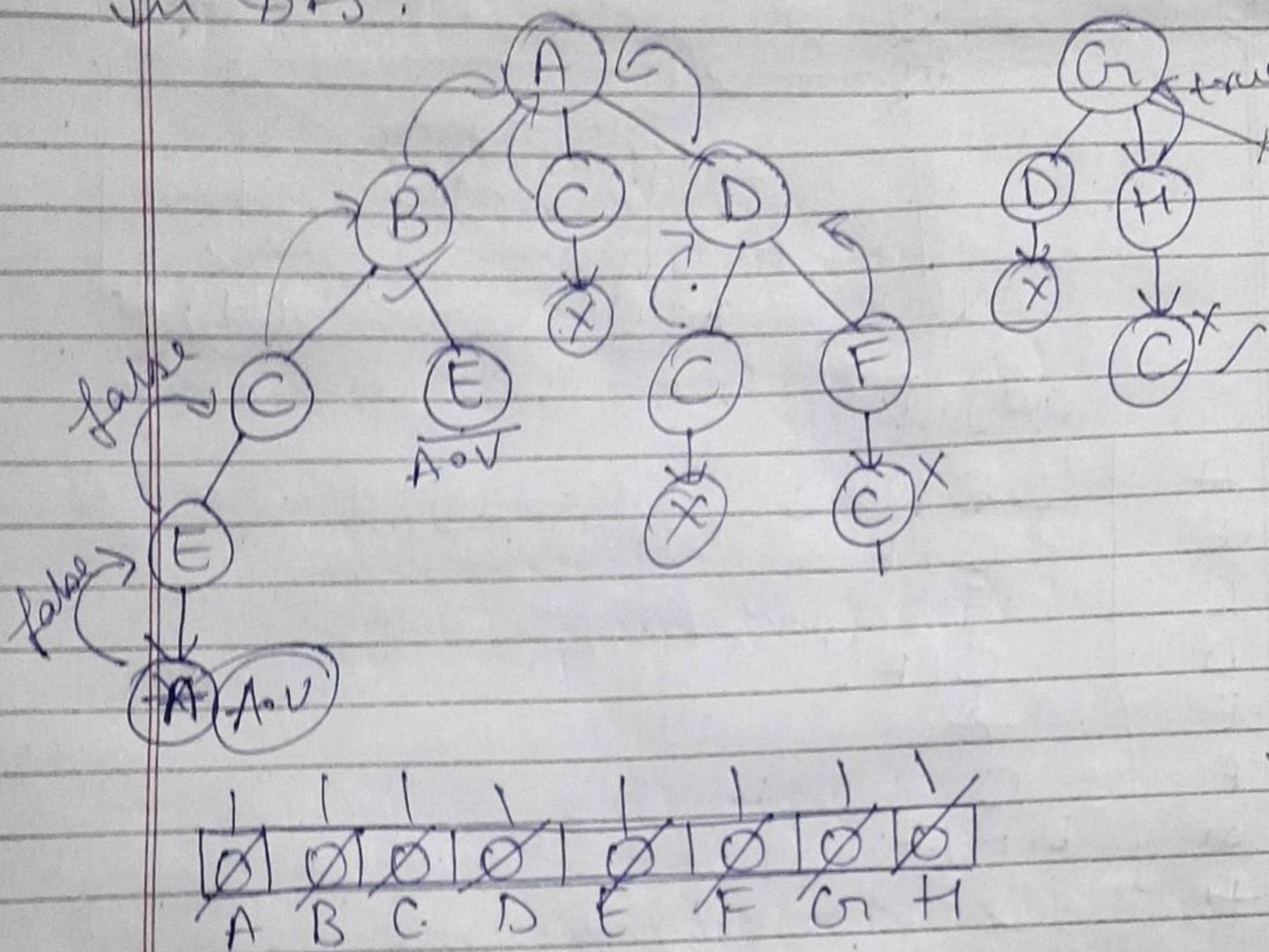
BFS., src = A dest = H





PAGE NO.: _____
DATE: / /

Jm DFS.



(8.)
```
#include <bits/stdc++.h>
using namespace std;

class Topological_Sort
{
  public:
  vector<int> toposort(int n, vector
    <int> adj[])
  {
    queue<int> q;
    vector<int> indegree(n, 0);
    for(int i=0; i<n; i++)
    {
      for(auto it : adj[i])
      {
        indegree[it]++;
```

```
for(int i=0; i<n; i++)
{
    cout << indegree[i];
}
cout << endl;

for(int i=0; i<n; i++)
{
    if(indegree[i]==0)
    {
        q.push(i);
    }
}

vector<int> topo;
while(!q.empty())
{
    int node = q.front();
    q.pop();
    topo.push_back(node);

    for(auto it : adj[node])
    {
        indegree[it]--;
        if(indegree[it]==0)
        {
            q.push(it);
        }
    }
}
return topo; } };
```

```
int main()
{
  int nodes, edges;
  cin >> nodes >> edges;
  vector <int> adj[nodes+1];

for(int i=0; i<edges; i++).
  {
    int u, v;
    cin >> u >> v.
    adj[u].push_back(v);
  }

Topological_Sort tps;
vector <int> ans;
ans = tps.toposort(nodes+1, adj);

for(int i=0; i<ans.size(); i++)
  {
    cout << ans[i];
  }
return 0;
}
```

9. ① Every item has priority associated with it.
② An element with high priority is dequeued before an element with low priority.
③ If two elements have same priority, they are served according to their order in the queue.

Application : Dijkstra's Shortest path Algorithm using priority queue. When the graph is stored in the form of adjlist or matrix, priority queue can be used to extract minimum efficiently when implementing Dijkstra's algorithm.

Prims Algorithms used priority queue.

10.

Min-heap	Max-heap
In a min-heap the key present at the root node must be less than or equal to among the keys present at all of its children.	In a max-heap, the key present at the root node must be greater than or equal to among the keys at present at all of its children.
In a min-heap, the minimum key element present at the root	In a max-heap, the maximum key element present at root.

PAGE NO.: _____
DATE: / /

3.	A min-heap uses the ascending priority	A max heap uses the descending priority.
4.	In the construction of a Min-heap, the smallest element has priority.	In the construction of a Max-heap, the greatest element has priority.
5.	In a Min-heap, the smallest element is the first ito be popped from the heap.	In a Max-heap, the greatest element is the first ito be popped from the heap.