Tutorial-5

Name → Janvi Sah
Section → G
Roll No. → 62

Ans 1.

BFS	DFS
→ BFS stands for Breadth First Search.	DFS stands for Depth First Search.
→ BFS uses queue data structure	DFS uses stack data structure.
→ BFS can be used to find single source shortest path in an unweighted graph, because in BFS, we reach a vertex with minimum no. of edges from a source vertex.	In DFS, we might Traverse through more edges to reach a destination vertex from a source.
→ BFS is more suitable for searching vertex which are closer to the given source.	DFS is more suitable when there are solutions away from source.
→ Here, siblings are visited before the children.	Here, children are visited before the siblings.
→ It requires more memory.	It requires less memory.
Applications	Applications
i) Shortest path and minimum spanning tree for unweighted graph.	i) Detecting cycle in a graph
ii) Peer to peer networks	ii) Path finding
iii) GPS navigation system	iii) Topological Sorting
iv) To test if graph is bipartite.	iv) To test if graph is bipartite
	v) Solving puzzles with only one solution.

Ans2. The data structure used in BFS is a queue and a graph. The algorithm makes sure that every node is visited not more than once.

DFS algorithm Traverse a graph in a depthword motion and uses a stack to remember to get the next vertex to start a search when a dead end occurs in any iteration.

Ans 3.

Sparse graph is a graph in which the number of edges is close to minimal number of edges. Sparse graph can be a disconnected graph.

graph, $G = (V, E)$ in which $|E| = O(|V|)$

If a graph is sparse, we should store it as a list of edges.

Dense graph is a graph in which the number of edges is close to maximal number of edges.

graph, $G = (V, E)$ in which $|E| = O(|V^2|)$

If a graph is dense, we should store it as an adjacency matrix.

Ans 4.

Detect cycles in a directed graph using BFS.

Step 1. Compute in-degree (no. of incoming edges) for each of the vertex present in the graph and initialize the count of visited nodes as 0.

Step 2. Pick all the vertices with in-degree as 0 and add them in queue.

Step 3. Remove vertex from queue and then,

1. Increment count of visited nodes as 1.
2. Decrease in-degree by 1 for all its neighbouring nodes.
3. If in-degree is reduced to 0 then add it to queue.

Step 4. Repeat step 3: until queue is empty.

Step 5. If count of visited nodes is not equal to the no. of nodes in the graph has cycle otherwise not.

In

B

```
function Find(x) is
    if x.parent ≠ x then
        x.parent = Find(x.parent)
        return x.parent
    else
        return x
    end if
end function
```

iii) Merging two sets

The operation Union(x,y) replaces the set containing y with their union.

```
function Union(x,y) is
    x = Find(x)
    y = Find(y)
    if x = y then
        return
    end if
    if x.size < y.size then
        (x,y) = (y,x)
    end if
end function
```

Depth First Search (DFS) can be used to detect cycle in Graph. DFS for a connected graph produces tree. For disconnected graph it produces forest.

Ans. Disjoint set data structure

It 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 partioned into the disjoint subsets.
for example → $S_1 = \{1, 2, 3, 4\}$ and $S_2 = \{5, 6, 7, 8\}$

No elements in common.

Operations

(i) **Making new sets**

The MakeSet operation adds a new element into a new set containing only the new element, and the new set is added to the data structure.

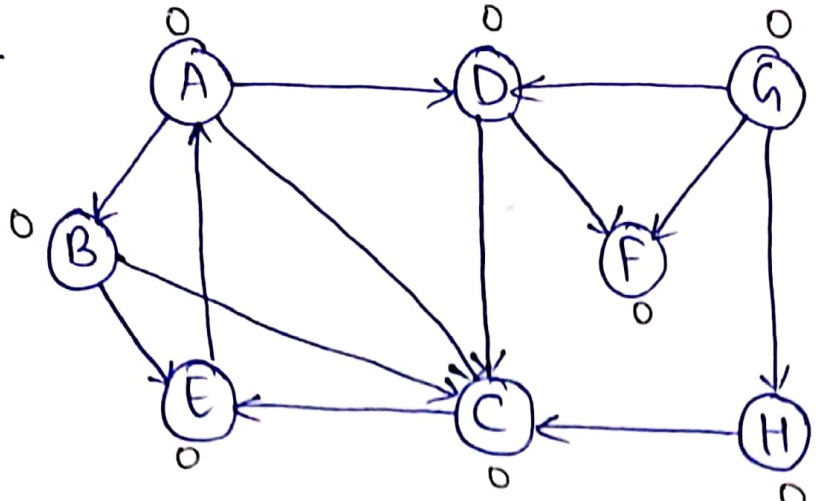
```
function MakeSet(x) is
    if x is not already in the forest then
        x.parent = x
        x.size = 1  // if node store size
        x.rank = 0  // if node store rank
    end if
end function
```

(ii) **Finding set representive**

The find operation follows the chain of parent pointers from a specified query node x until it reaches a root element.

Ans 6.

BFS



← source node

(i) Mark all visited as 0.

queue

node	G	D	F	H	C	E	A	B
parent	X	G	G	G	D	C	E	A

dequeue

G	D	F	H	C	E	A	

Traversal → GDFHCEAB

DFS

(i) Visited | G | | | | | | |

Stack | D | F | H | | | | |

(ii) Visited | G | D | | | | | |

Stack | F | H | C |

(iii) Visited | G | D | F | | | | |

Stack H C

(iv) Visited | G | D | F | H | | | |

Stack C

v) Visited | G | D | F | H | C | | |

Stack E

vi) Visited | G | D | F | H | C | E | |

Stack A

vii) Visited | G | D | ~~F~~ | H | C | E | A |

Stack B

viii) Visited | G | D | F | H | C | E | A | B |

Stack Empty.