Tutorial 5

Ayank Gupta
Section B

Ans 1

| BFS | DFS |
|---|---|
| ① uses queue data structure. | ① uses stack data structure. |
| ② can be used to find shortest single source path in an unweighted graph and we reach a vertex with min. no. of edges from a source vertex. | ② we might traverse through more edges to reach a destination vertex from a source. |
| ③ siblings are visited before the children. | ③ children are visited before siblings. |
| ⊙ Applications ⇒ | ⊙ Applications ⇒ |
| ① shortest path and min. spanning tree. | ① Detecting cycle in graph |
| ② peer to peer networks | ② path finding |
| ③ social networking sites | ③ topological sorting. |
| ④ GPS navigation systems. | ④ solving puzzles with only one soln. |

Ans2 ⇒ In BFS we use Queue data structure as queue is used when things dont have to be processed immediately, but have to be processed in FIFO manner like BFS.

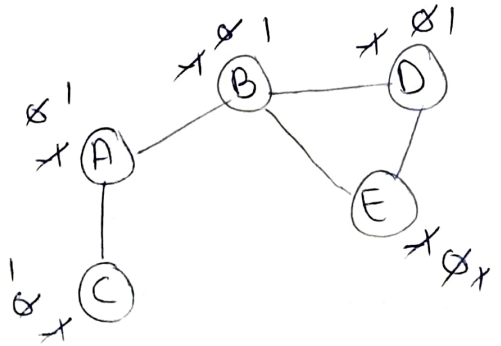
⇒ In DFS stack is used as DFS uses backtracking. For DFS, we retrieve it from root to the farthest node as much as possible. that is same idea as LIFO.

Ans3 ⇒ Dense graph is a graph in which no. of edges is more to the maximal no. of edges.

⇒ sparse graph is a graph in which no. of edges is close to to the minimal no. of edges. It can be a disconnected graph.

- For adjacency lists ⇒ sparse graphs
- For adj. matrix ⇒ dense graphs.

Ans4 ⇒ cycle detection in undirected graph (BFS)




-1 = unvisited

0 = into queue

1 = traversed.

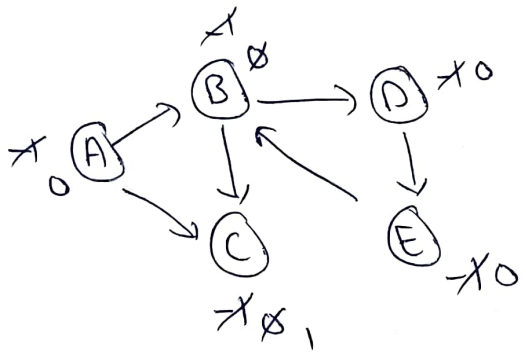
Queue ⇒ | A | B | C | D | E |

visited ⇒ | A | B | C | D |

⇒ when D checks its adj. vertices it finds E with 0.

⇒ if any vertex finds the adj. vertices with 0 ⇒ it contains cycle.

* cycle detection in directed graph ⇒




stack ⇒

| E |
|---|
| D |
| B |
| A |

visited set ⇒ A B C D E

⇒ B → D → E → B

Parent Map

| vertex | Parent |
|---|---|
| A | – |
| B | A |
| C | B |
| D | B |
| E | D |

⇒ these E finds B (adj. of E) with 0

⇒ it contains a cycle

Ans 5 ⇒ The disjoint data structure is also known as union find data structure as merge find set.

It is a DS that contains the collision of disjoint sets.

The disjoint means that the set is partitioned into disjoint subsets, various operation can be performed on it.

* Operations on disjoint Sets ⇒

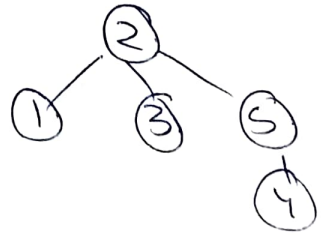
① union ⇒ (a) If S1 and S2 are 2 disjoint sets ⇒ their union S1US2 is a set of all elements x such that x is in S1 or S2.

(b) As the sets should be disjoint S1US2 replaces S1 & S2 which no longer exists.

(c) union is achieved by simply making one of the trees as the subtree of other. i.e. to set parent of one of the roots of the tree to other set.

(i) Eg. ⇒


S1 U S2 ⇒




[merge the sets into 1]

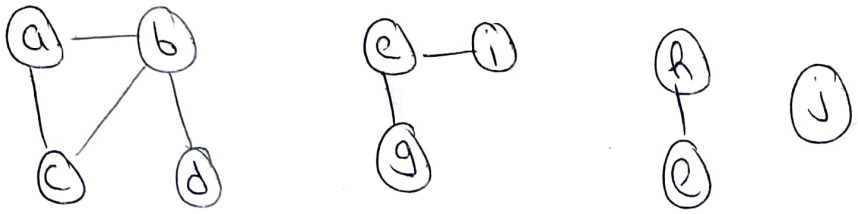
② find ⇒ given an element x, to find the set containing it ⇒


find(3) = S1

find(5) = S2

* make_set (x) ⇒ create a set containing x.

Ans 7 ⇒




$V = \{a, b, c, d, e, j, h, i, j, e\}$

$E = \{(a,b)(a,c)(b,c)(b,d)(e,i)(e,g)(h,e)(j)\}$

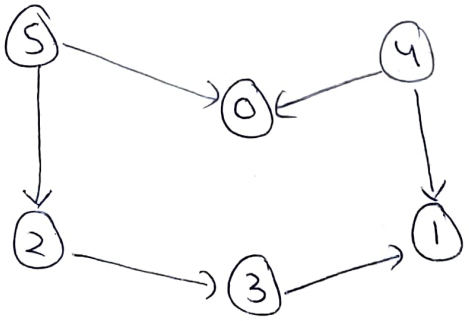
* we have now ⇒ {a, b, c, d}

{e, i, g}

{h, e}

{j}

Ans 8 ⇒




0

1

2 → 3

3 → 1

4 → 0 → 1

5 → 2 → 0

① Algo ⇒ go to node 0, it has no outgoing edges so push 0 into stack and mark visited.

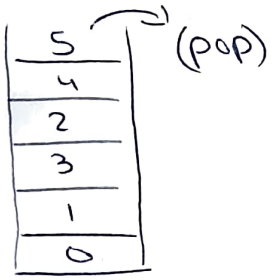
② go to node 1, again it has no outgoing edges ⇒ so push node 1 into stack and mark visited.

(3) go to node 2 ⇒ process all adj. nodes and mark node 2 visited.

(4) node 3 is already visited to continue with next node.

(5) go to node 4 ⇒ all its adj. nodes are already visited. so push node 4 into stack and mark visited.

(6) go to node 5, all its adj. nodes are already visited so push node 5 into stack and mark visited.




5 4 2 3 1 0 ⇒ o/p.

**Ans9** ⇒ Heap is generally preferred for priority Queue implementation as Heaps provide better performance compared to arrays | Linked Lists.

⇒ Algos where priority Queue is used ⇒

1. Dijsktra's Algorithm
2. Prim's Algorithm.

**Ans10** ⇒

| MinHeap | MaxHeap |
|---|---|
| (1) Parent has always lower value than child. | (1) parent has always higher value than child. |
| (2) Root node has smallest value. | (2) Root has highest value |