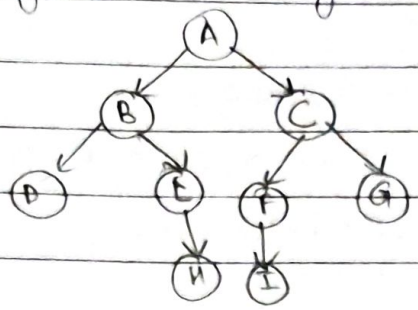
(ii) (a) write an algorithm for BFS

(b) Using the algo. find the BFS of the graph starting from vertex A.



- (a)

```
bfs (int v)
{
    visited[v] = true;
    initialize Queue (q)
    addTo Queue (q, v)
    while not isQueueEmpty (q) do
    {
        removefrom Queue (q, v)
        print(v)
        for all vertices w adjacent to v, do
        {
            if not visited[w], then
            {
                addToQueue (q, w)
                visited[w] = true;
            }
        }
    }
```

(b) visited: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
A B C D E F G H I

① bfs(A)
Visited[A] = true
q [A]

② q not empty
remove A from q.
Unvisited verts adj to A: B, C. ⇒ add to q.
Res: A
q [B | C]

③ q not empty
remove B from q.
Result: A → B.
Unvisited nodes adj to B: D, E → add to q
q [C | D | E]

④ Remove C from q
Res: A → B → C.
Unvisited nodes adj to C: F, G.
q [D | E | F | G]

⑤ Remove D from q
Result: A → B → C → D
Unvisited adj to D: —
q [E | F | G]

⑥ Remove E from q.
Res: A → B → C → D → E
Unvisited adj to E: H.
q [F | G | H]

⑦ Remove F from q
Res: A → B → C → D → E → F
Unvisited node adj to F: I.
q [G | H | I]

⑧ Remove G from q
Res: A → B → C → D → E → F → G
No unvisited adj node.
[H | I]

⑨ Remove H from q.
Res: A → B → C → D → E → F → G → H
q [I]

⑩ Remove I from q.
Res: A → B → C → D → E → F → G → H → I
q [ ]

⑪ Queue empty
∴ Stop.

RESULT: A → B → C → D → E → F → G → H → I