28/02/24

Week : 9

BFS.

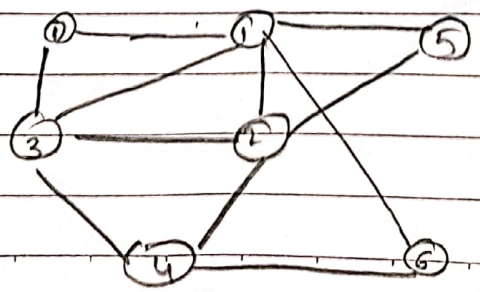
```
#include <stdio.h>
#define MAX_VERTICES 10
int n, i, j, visited[MAX_VERTICES],
    queue, [MAX_VERTICES], front = 0, rear = 0
int adj [MAX_VERTICES] [MAX_VERTICES];

Void bfs (int v) {
    visited [v] = 1;
    queue [rear++] = v;
  while (front < rear) {
      int current = queue [front++];
      Printf ("%d\t", current);

for (int i = 0; i < n; i++) {
    if (adj [current] [i] && !visited [i])
        visited [i] = 1;
        queue [rear++] = i;
    }
  }
}

int main () {
    int v;
    Printf ("Enter number of vertices:");
    Scanf ("%d", &n);
    for (i = 0; i < n; i++) {
        visited [i] = 0;
    }
```

```
Printf ("Enter the graph data in matrix form");
        for (i=0; i<n; i++)
            for (j=0; j<n; j++)
                Scanf ("%d", &adj[i][j]);
Printf ("Enter the Starting vertex");
Scanf ("%d", &v);
bfs(v);
for(i=0; i<n; i++)
    if(!visited[i])
Printf ("In BFS it is not Possible. Not all nodes
        are reachable. \n");
return 0;
}
```

Output: Enter the number of vertices: 7
Enter graph data in matrix form:

0 1 0 1 0 0 0
0 1 0 0 1 0 0
0 0 1 0 0 0 1
0 1 1 0 0 0 0
0 1 0 0 1 0 0

Enter the Starting vertex: 4
4 2 3 6 1 5 0



Visited 4, 2, 3, 6
4 → 2, 3, 6
2 → 1, 3, 4, 5, 6
1 → 0, 2, 3, 5, 6

DFS:

```
#include <stdio.h>
#include <stdbool.h>
#define MAX_VERTICES 10

void dfs(int graph[MAX_VERTICES][MAX_VERTICES], int num_vertices, bool visited[MAX_VERTICES], int vertex) {
    visited[vertex] = true;
    int i;
    for (i = 0; i < num_vertices; i++) {
        if (graph[vertex][i] == 1 && !visited[i])
            dfs(graph, num_vertices, visited, i);
    }
}

bool isConnected(int graph[MAX_VERTICES][MAX_VERTICES], int num_vertices) {
    bool visited[MAX_VERTICES] = {false};

    dfs(graph, num_vertices, visited, 0);

    for (int i = 0; i < num_vertices; i++) {
        if (!visited[i]) {
            return false;
        }
    }
    return true;
}
```

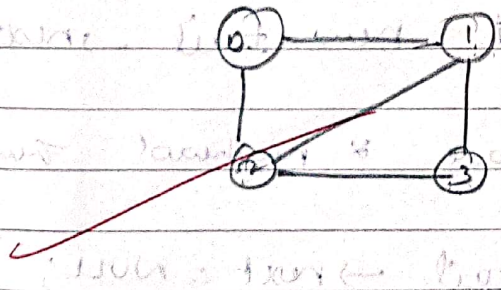
```
int main() {
    int num_vertices;
    Printf ("Enter number of vertices");
    Scanf ("%d", &num_vertices);
    int i,j;
    int graph [MAX_VERTICES][MAX_VERTICES];
    Printf ("Enter the adjacency matrix:\n");
    for (i=0; j < num_vertices ; ++j) {
        Scanf ("%d", &graph[i][j]);
    }
}
```

Output: Enter the number of vertices : 4

Enter the adjacency matrix:

0 1 1 0
1 0 0 1
1 0 0 0
0 1 0 0

The graph is Connected.



Method - 4

- Rotate list

```
Struct ListNode * rotateRight (Struct ListNode* head, int k)
{
    If (head == NULL || head -> next == NULL || k == 0)
    return head;
    int length = 1;
    Struct ListNode *tail = head;
    While (tail -> next)
    {
        length ++;
        tail = tail -> next;
    }
    k = k % length;
    if (k == 0)
    return head;
    Struct ListNode * new_tail = head;
    for (int i = 0; i < length
    {
        new_tail = new_tail -> next;
    }
    Struct ListNode * newhead = new_tail ->
    next;
    new_tail -> next = NULL;
    tail -> next = head;
    return new_head;
}
```

output

Case 1

head = [1, 2, 3, 4, 5]

k = 2

Case 2 :

head = [0, 1, 2]

k = 4

Sol:1

22/7/24.