Lab: (26/02/24)

__/__/____

1) Write a program to complete breath first search in trees.

```
#include <stdio.h>
#include <stdlib.h>
#define Max_Nodes 100

Struct Node {
    int data;
    Struct Node* next;
};
Struct Graph {
    int numNodes;
    Struct Node* adjLists[Max_Nodes];
    int visited[Max_Nodes];
};
Struct Node* createNode(int data) {
    Struct Node* newNode = (Struct Node*)malloc(sizeof
                                    (Struct Node));
    NewNode->data = data;
    newNode->next = NULL;
    return newNode;
}
Struct Graph* createGraph(int n) {
    struct Graph* graph= (struct Graph*)malloc(sizof(
                                    struct Graph));
    graph->numNodes = n;
    for (int i = 0; i < n; i++) {
        graph->adjList[i] = NULL;
        graph->visited[i] = 0;
    }
    return graph;
}
```

```
void addEgge(struct Graph* graph, int src, int dest) {
Struct Node* newNode = creatiNode(dest);
    newNode → next = graph → adjList[src];
    graph → adjLists[src] = newnode;

    newNode = creatiNode(src);
    newNode → next = graph → adjLists[dest];
    graph → adjList[dest] = newNode;
}

void BFS(Struct Graph* graph, int startNode) {
    int queue[Max_Nodes];
    int front=0, rear=0;
    graph → visited[startNode] = 1
    queue[rear++] = startNode;
    while(front < rear) {
        int current = queue[front++];
        printf("%d", current);
    struct node *temp = graph → adjLists[current];
    while(temp) {
        int adjNode = temp → data;
    if(!graph → visited[adjNode]) {
        graph → visited[adjNode]) = 1;
        queue[rear++] = adjNode;
    }
    temp = temp → next;
    }
  }
}
```

```
int main (){
    int numNodes;
    printf ("Enter the no of nodes:" );
    scanf ("%d", &numNodes);
    struct Graph* graph = creatGraph (numNodes);
    int numEgdes;
    printf("Enter the number of edges : ");
    scanf ("%d ", &numEdges);
    for(int i=0 ; i < numEdges; i++) {
    int src , dest ;
    printf("Enter edge %d (source): ", i+1 );
    Scanf ("%d %d , &src , &dest );
    addedge ( graph , src , dest );
    int startNode ;
    printf("Enter the starting node for BFS traversal: ");
    scanf("%d ", &startNode );
    printf("BFS traversal starting from node %d : ", start
    BFS (graph startNode);
    return 0 ;
}
```

NL
26/2/24

2) Program to show depth first search in a tree graph.

```
#include <stdio.h>
#include <stdlib.h>
#define Max_Nodes 100

Struct Node {
        int data;
        struct Node* next;
};
Struct Graph {
        int NumNodes;
        struct Node* adjLists[Max_Nodes];
        int visited[Max_nodes];
};
Struct Node* CreateNode (int data) {
        struct Node* newNode = (Struct Node*) malloc
        newNode → data = data;                    (sizeof(Struct Node));
        new node → next = NULL;
        return newnode;
}
Struct Graph* gCreateGraph (int n) {
        struct Graph* graph = (Struct Graph*) malloc(
                                        sizeof(struct Graph));
        graph → numNodes = n;
        for (int i = 0; i < n; i++){
                graph → adjList[i] = NULL;
                graph → visited[i] = 0;
        }
    return graph;
}
```

__/__/____

```
Void addEdge (struct Graph* graph, int src, int dest) {
    struct Node* newNode = createNode (dest);
    newNode → node = graph → adjLists [src];
    graph → adjList [src] = newNode;
    newNode = createNode (src);
    newNode → next = graph → adjList[dest];
    graph → adjLists[dest] = newNode;
}
void DFS (struct Graph * graph, int startNode ){
    graph → visited [ startNode ] = 1 ;
    printf ( "%d ", startNode );
    Struct Node* temp = graph → adjLists[startNode];
    while (temp) {
        int adjNode = temp → data ;
        if (! graph → visited [ adjNode ]) {
            DFS (graph, adjNode );
        }
        temp = temp → next ;
    }
}
int main ( ) {
    int numNodes ;
    printf ("Enter number of nodes : ");
    scanf ("%d ", & numNodes);
    struct Graph = createGraph (numNodes );
    int Edges;
    printf ("Enter the number of edges: " );
    scanf ("%d ", & numEdges);
```

```
    for (int i =0; i< numEdges; i++){
    int src, dest;
    printf ("Enter edge %d( source, destination): ",
                                        i+1);
    scanF("%d, %d", &src, &dest);
}
int startNode;
printf ("Enter the starting node for DFS traversal:");
scanf ("%d", &startNode);
printf ("DFS traversal starting from node %d:", startNode);
DFS (graph, startNode);
return 0;
}.
```