ASSIGNMENT 5

① DFS

```
#include<iostream>
#include<stack>
using namespace std;
void read_matrix(int **graph, int n){
    cout<<" Matrix = "<<endl;
    for(int i=0; i<n; i++){
        graph[i] = new int[n];
        for(int j=0; j<n; j++)
            cin>>graph[i][j];
    }
}
void dfs(int **graph, int s_node, int n, int d_node){
    bool visited[n];
    for(int i=0; i<n; i++)
        visited[i]=false;
    stack<int> stack;
    stack.push(s_node);
    cout<<" DFS = "<<endl;
    while(!stack.empty()){
        int current_node = stack.top();
        stack.pop();
        if(!visited[current node]){
            cout<<current_node<<" ";
            visited[current_node]=true;
            for(int i=0; i<n; i++){
                if(graph[current node][i]==1 && !visited[i])
                    stack.push(i);
            }
            if(current_node==d_node)
            exit(0);
    }cout<<endl;
}
```

```
int main() {
    int nodes;
    cout<< " Entn no of nodes "<<endl;
    cin >> nodes;
    int **graph = new int *[nodes];
    read_matrix (graph, nodes);
    int starting_node, destination_node;
    cout<< " Entn starting & destination nodes "<< endl;
    cin >> starting_node >> destination_node;
    dfs (graph, starting_node, nodes, destination_node);
    return 0;
}
```

Output

Enter no of nodes

8

Matrix

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

Enter starting and distination nodes

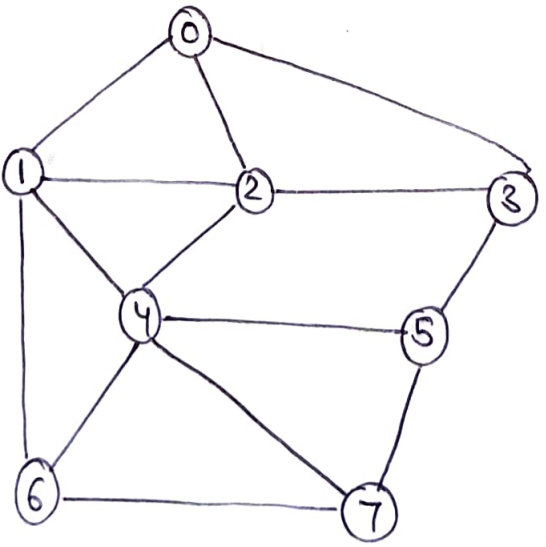
0

7

DFS

0 3 5 7

Given graph is:



BFS

```
#include <iostream>
#include <queue>
using namespace std;
void read_matrix(int **graph, int n){
    cout<<" Matrix"<<endl;
    for(int i=0; i<n; i++){
        graph[i] = new int[n];
        for(int j=0; j<n; j++){
            cin>>graph[i][j];
    }
}
void bfs(int **graph, int s_node, int d_node, int n){
    bool visited[n];
    int parent[n];
    for(int i=0; i<n; i++)
        visited[i] = false, parent[i] = -1;
    queue<int> q;
    visited[s_node] = true;
    q.push(s_node);
    while(!q.empty()){
```

```
int current_node = q.front();
q.pop();
for (int neigh = 0; neigh < n; neigh++) {
  if (graph[current_node][neighbour] == 1 && !visited[neighbour]) {
    visited[neighbour] = true;
    parent[neighbour] = current_node;
    q.push(neigh);
  }
}
}
if (!visited[d_node])
  cout << "No path";
else {
  cout << "BFS";
  int shortest_path[n], current = d_node; path_length = 0;
  while (current != -1) {
    shortest_path[path_length++] = current;
    current = parent[current];
  } for (int i = path_length - 1; i >= 0; i--) {
    cout << shortest_path[i];
    if (i != 0)
      cout << "->";
    cout << endl;
} }
int main() {
int nodes;
cin >> nodes;
int** graph = new int*[nodes];
read_matrix(graph, nodes);
int s_node, d_node;
cin >> s_node >> d_node;
bfs(graph, s_node, d_node, nodes);
return 0;
}
```

Output

Enter no of nodes

8

Matrix

```
0 1 1 1 0 0 0 0
1 0 1 0 1 0 1 0
1 1 0 1 1 0 0 0
1 0 1 0 0 1 0 0
0 1 1 0 0 1 1 1
0 0 0 1 1 0 0 1
0 1 0 0 1 0 0 1
0 0 0 0 1 1 1 0
```

Enter starting & destination nodes

0

7

BFS

0 → 1 → 4 → 7

PEAS parameters of the roadmap agent is

① Performance
- its performance can be measured by its accuracy of directions
- how well the agent optimizes routes to minimize travel time or distance.
- also it should tell traffic conditions.

② Environment
The agent must also provide real time information of roads, highways, streets, landmarks, and also weather conditions.

③ Actuators
The agent uses GPS, voice instructions, display screens. Generates directions, updates routes and responds to user commands.

④ Sensors
Traffic sensor, GPS sensors, cameras, microphones, internet.

Designing an intelligent tourist recommender —
An intelligent tourist recommender must recommend suitable destinations, attractions, and accomodations for tourists based on their preferences, budget, time, etc.

① Performance - The satisfaction and the overall feedback of tourists, their revenue and the total profit of the recommendation system and the quality of recommendations, etc.

(2) Environment - the database of tourists information, their online websites and online platforms, the user ratings and user reviews and also the user interface.

(3) Actuator - the processor, the recommendation algo, the display screen, the speaker, the keyboard, etc.

(4) Sensor - these include microphone, camera, etc.