**Assignment no 6**

**SAI ANANT PATIL**

**COB208**

/\*There are flight paths between cities. If there is a flight between city A and city B

then there isan edge between the cities. The cost of the edge can be the time that flight takes to reach cityB from A, or the amount of fuel used for the journey. Represent this as a graph. The node canbe represented by airport name or name of the city. Use adjacency MATRIX representation of the graph.\*/

|  |
| --- |
|  |
|  |
|  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | #include<iostream> |
|  | #include<queue> |
|  | #include<stack> |
|  | usingnamespacestd; |
|  |  |
|  | classGraph |
|  | { |
|  | string city[10]; |
|  | int a[10][10]; |
|  | int n; |
|  | public: |
|  | voidinput(); |
|  | voiddisplay(); |
|  | voidBFS(); |
|  | voidDFS(); |
|  | }; |
|  |  |
|  | voidGraph::input() |
|  | { |
|  | cout<<"\nEnter no. of cites: "; |
|  | cin>>n; |
|  | cout<<"\nEnter the names of cities: "; |
|  | for(inti=0 ; i<n ; i++) |
|  | cin>> city[i]; |
|  |  |
|  | cout<<"\nEnter the distances: "; |
|  | for(inti=0 ; i<n ; i++) |
|  | for(int j=i ; j<n ; j++) |
|  | { |
|  | if(i==j) |
|  | { |
|  | a[i][j] = 0; |
|  | continue; |
|  | } |
|  |  |
|  | cout<<"\nEnter the distance between "<< city[i] <<" and "<< city[j]<<" : "; |
|  | cin>> a[i][j]; |
|  | a[j][i] = a[i][j]; |
|  | } |
|  | } |
|  |  |
|  | voidGraph::display() |
|  | { |
|  |  |
|  | for(inti=0 ; i<n ; i++) |
|  | { |
|  | cout<<"\n"; |
|  | for(int j=0 ; j<n ; j++) |
|  | { |
|  | cout<<a[i][j] <<"\t"; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | voidGraph::BFS() |
|  | { |
|  | cout<<"\n\nBFS Traversal: "; |
|  | queue<int> q; |
|  | int visit[n]; |
|  | for(inti=0 ; i<n ; i++) |
|  | visit[i] = 0; |
|  | string start; |
|  | intindex; |
|  | cout<<"\nEnter starting city: "; |
|  | cin>>start; |
|  | for(inti=0 ; i<n ; i++) |
|  | if(start == city[i]) |
|  | index =i; |
|  |  |
|  | visit[index] = 1; |
|  | cout<<city[index]<<" -> "; |
|  | int current = index; |
|  | while(1) |
|  | { |
|  | for(inti=0 ; i<n ; i++) |
|  | { |
|  | if(a[current][i]!=0&& visit[i] == 0) |
|  | { |
|  | visit[i] = 1; |
|  | q.push(i); |
|  | cout<<city[i]<<" -> "; |
|  | } |
|  |  |
|  | } |
|  |  |
|  | if(q.empty()!=0) |
|  | break; |
|  |  |
|  | else |
|  | { |
|  | current = q.front(); |
|  | q.pop(); |
|  | } |
|  | } |
|  | } |
|  |  |
|  | voidGraph::DFS() |
|  | { |
|  | cout<<"\n\nDFS Traversal: "; |
|  | stack<int> s; |
|  | int visit[n]; |
|  | for(inti=0 ; i<n ; i++) |
|  | visit[i] = 0; |
|  | string start; |
|  | intindex; |
|  | cout<<"\nEnter starting city: "; |
|  | cin>>start; |
|  | for(inti=0 ; i<n ; i++) |
|  | if(start == city[i]) |
|  | index =i; |
|  | s.push(index); |
|  | visit[index] = 1; |
|  | int current = index; |
|  | cout<< city[index]<<" -> "; |
|  | while(1) |
|  | { |
|  | for(inti=0 ; i<n ; i++) |
|  | { |
|  | if(a[current][i]!=0&& visit[i]==0) |
|  | { |
|  | s.push(i); |
|  | cout<<city[i]<<" -> "; |
|  | visit[i] = 1; |
|  | current = i; |
|  | i=0; |
|  | } |
|  | } |
|  |  |
|  | if(s.empty()!=0) |
|  | break; |
|  |  |
|  | else |
|  | { |
|  | current = s.top(); |
|  | s.pop(); |
|  | } |
|  | } |
|  | } |
|  | intmain() |
|  | { |
|  | Graph g1; |
|  | int choice; |
|  | MENU: |
|  | cout<<"\n\nGRAPH TRAVERSAL"; |
|  | cout<<"\n1. Input data"; |
|  | cout<<"\n2. Display data"; |
|  | cout<<"\n3. DFS Traversal"; |
|  | cout<<"\n4. BFS Traversal"; |
|  | cout<<"\n5. Exit"; |
|  | cout<<"\nEnter your choice: "; |
|  | cin>> choice; |
|  | switch(choice) |
|  | { |
|  | case1: |
|  | g1.input(); |
|  | break; |
|  | case2: |
|  | g1.display(); |
|  | break; |
|  | case3: |
|  | g1.DFS(); |
|  | break; |
|  | case4: |
|  | g1.BFS(); |
|  | break; |
|  | case5: |
|  | return0; |
|  | default: |
|  | cout<<"\nInvalidchoice.Try again!"; |
|  | } |
|  | if(choice != 5) |
|  | goto MENU; |
|  | return0; |
|  | }  Output: |

