**EXPERIMENT NO: 07**

#include<iostream>

#include<stdlib.h>

#include<string.h>

using namespace std;

struct node

{ string vertex;

int time;

node \*next;

};

class adjmatlist

{ int m[10][10],n,i,j; char ch; string v[20]; node \*head[20]; node \*temp=NULL;

public:

adjmatlist()

{ for(i=0;i<20;i++)

{ head[i]=NULL; }

}

void getgraph();

void adjlist();

void displaym();

void displaya();

};

void adjmatlist::getgraph()

{

cout<<"\n enter no. of cities(max. 20)";

cin>>n;

cout<<"\n enter name of cities";

for(i=0;i<n;i++)

cin>>v[i];

for(i=0;i<n;i++)

{

for(j=0;j<n;j++)

{ cout<<"\n if path is present between city "<<v[i]<<" and "<<v[j]<<" then press enter y otherwise n";

cin>>ch;

if(ch=='y')

{

cout<<"\n enter time required to reach city "<<v[j]<<" from "<<v[i]<<" in minutes";

cin>>m[i][j];

}

else if(ch=='n')

{ m[i][j]=0; }

else

{ cout<<"\n unknown entry"; }

}

}

adjlist();

}

void adjmatlist::adjlist()

{ cout<<"\n \*\*\*\*";

for(i=0;i<n;i++)

{ node \*p=new(struct node);

p->next=NULL;

p->vertex=v[i];

head[i]=p; cout<<"\n"<<head[i]->vertex;

}

for(i=0;i<n;i++)

{ for(j=0;j<n;j++)

{

if(m[i][j]!=0)

{

node \*p=new(struct node);

p->vertex=v[j];

p->time=m[i][j];

p->next=NULL;

if(head[i]->next==NULL)

{ head[i]->next=p; }

else

{ temp=head[i];

while(temp->next!=NULL)

{ temp=temp->next; }

temp->next=p;

}

}

}

}

}

void adjmatlist::displaym()

{ cout<<"\n";

for(j=0;j<n;j++)

{ cout<<"\t"<<v[j]; }

for(i=0;i<n;i++)

{ cout<<"\n "<<v[i];

for(j=0;j<n;j++)

{ cout<<"\t"<<m[i][j];

}

cout<<"\n";

}

}

void adjmatlist::displaya()

{

cout<<"\n adjacency list is";

for(i=0;i<n;i++)

{

if(head[i]==NULL)

{ cout<<"\n adjacency list not present"; break; }

else

{

cout<<"\n"<<head[i]->vertex;

temp=head[i]->next;

while(temp!=NULL)

{ cout<<"-> "<<temp->vertex;

temp=temp->next; }

}

}

cout<<"\n path and time required to reach cities is";

for(i=0;i<n;i++)

{

if(head[i]==NULL)

{ cout<<"\n adjacency list not present"; break; }

else

{

temp=head[i]->next;

while(temp!=NULL)

{ cout<<"\n"<<head[i]->vertex;

cout<<"-> "<<temp->vertex<<"\n [time required: "<<temp->time<<" min ]";

temp=temp->next; }

} }

}

int main()

{ int m;

adjmatlist a;

while(1)

{

cout<<"\n\n enter the choice";

cout<<"\n 1.enter graph";

cout<<"\n 2.display adjacency matrix for cities";

cout<<"\n 3.display adjacency list for cities";

cout<<"\n 4.exit";

cin>>m;

switch(m)

{ case 1: a.getgraph();

break;

case 2: a.displaym();

break;

case 3: a.displaya();

break;

case 4: exit(0);

default: cout<<"\n unknown choice";

}

}

return 0;

}

**OUTPUT:**

enter the choice

1.enter graph

2.display adjacency matrix for cities

3.display adjacency list for cities

4.exit

1

enter no. of cities(max. 20):3

enter name of cities: mumbai pune nashik

if path is present between city mumbai and mumbai then press enter y otherwise n: n

if path is present between city mumbai and pune then press enter y otherwise n: y

enter time required to reach city pune from mumbai in minutes:120

if path is present between city mumbai and nashik then press enter y otherwise n: y

enter time required to reach city nashik from mumbai in minutes: 130

if path is present between city pune and mumbai then press enter y otherwise n: y

enter time required to reach city mumbai from pune in minutes: 140

if path is present between city pune and pune then press enter y otherwise n: n

if path is present between city pune and nashik then press enter y otherwise n: y

enter time required to reach city nashik from pune in minutes: 150

if path is present between city nashik and mumbai then press enter y otherwise n: y

enter time required to reach city mumbai from nashik in minutes: 160

if path is present between city nashik and pune then press enter y otherwise n: n

if path is present between city nashik and nashik then press enter y otherwise n: n

\*\*\*\*

mumbai

pune

nashik

enter the choice

1.enter graph

2.display adjacency matrix for cities

3.display adjacency list for cities

4.exit

2

mumbai pune nashik

mumbai 0 120 130

pune 140 0 150

nashik 160 0 0

enter the choice

1.enter graph

2.display adjacency matrix for cities

3.display adjacency list for cities

4.exit

3

adjacency list is

mumbai-> pune-> nashik

pune-> mumbai-> nashik

nashik-> mumbai

path and time required to reach cities is

mumbai-> pune

[time required: 120 min ]

mumbai-> nashik

[time required: 130 min ]

pune-> mumbai

[time required: 140 min ]

pune-> nashik

[time required: 150 min ]

nashik-> mumbai

[time required: 160 min ]

enter the choice

1.enter graph

2.display adjacency matrix for cities

3.display adjacency list for cities

4.exit