#include<stdio.h>

#include<conio.h>

#define MAX 100

int a[20][20],q[20],visited[20],n,i,j,f=0,r=-1,indegree[20];

int adj[MAX][MAX],reach[20];

int queue[MAX],front=-1,rear=-1;

void insert\_queue(int v);

int delete\_queue();

int isEmpty\_queue();

void main()

{

int ch,v,j,G[MAX][MAX],count=0;

clrscr();

do

{

printf("\n1.BFS\n2.DFS\n3.Topological Sort\n4.Exit\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1: printf("Enter the number of vertices:");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

q[i]=0;

visited[i]=0;

}

printf("\nEnter graph data in matirx form:\n");

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

printf("\nEnter the starting vertex:");

scanf("%d",&v);

visited[v]=1;

printf("%d\t",v);

bfs(v);

break;

case 2: printf("\nEnter the number of vertices:");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

reach[i]=0;

for(j=1;j<=n;j++)

a[i][j]=0;

}

printf("\nEnter the adjacency matrix:\n");

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

dfs(1);

printf("\n");

for(i=1;i<=n;i++)

{

if(reach[i])

count++;

}

if(count==n)

printf("\nGraph is connected");

else

printf("\nGraph is not connected");

break;

case 3: printf("\nEnter the number of vertices:");

scanf("%d",&n);

printf("\nEnter the adjacency matrix:\n");

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

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

scanf("%d",&a[i][j]);

topology();

getch();

break;

case 4: exit(0);

default:printf("Invalid input");

}

}while(ch!=5);

getch();

}

int bfs(int v)

{

for(i=1;i<=n;i++)

if(a[v][i] && !visited[i])

q[++r]=i;

if(f<=r)

{

visited[q[f]]=1;

printf("%d\t",q[f]);

bfs(q[f++]);

}

return 0;

}

int dfs(int v)

{

int i;

reach[v]=1;

for(i=1;i<=n;i++)

if(a[v][i] && !reach[i])

{

printf("\n%d->%d",v,i);

dfs(i);

printf("\n%d",v);

}

return 0;

}

void insert\_queue(int vertex)

{

if(rear==MAX-1)

printf("\nQueue Overflow\n");

else

{

if(front==-1)

front=0;

rear=rear+1;

queue[rear]=vertex;

}

}

int isEmpty\_queue()

{

if((front==-1)||(front>rear))

return 1;

else

return 0;

}

int delete\_queue()

{

int del\_item;

if((front==-1)||(front>rear))

{

printf("\nQueue Underflow\n");

exit(1);

}

else

{

del\_item=queue[front];

front=front+1;

return del\_item;

}

return 0;

}

void find\_indegree()

{

int i,j,sum;

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

{

sum=0;

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

sum+=a[i][j];

indegree[j]=sum;

}

}

int topology()

{

int i,u,v,t[10],s[10],top=-1,k=0;

delay(1000);

find\_indegree();

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

{

if(indegree[i]==0)

s[++top]=i;

}

while(top!=-1)

{

u=s[top--];

t[k++]=u;

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

{

if(a[u][v]==1)

{

indegree[v]--;

if(indegree[v]==0)

s[++top]=v;

}

}

}

printf("\nThe Topological Sequence is: ");

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

printf("\t%d",t[i]+1);

return 0;

}