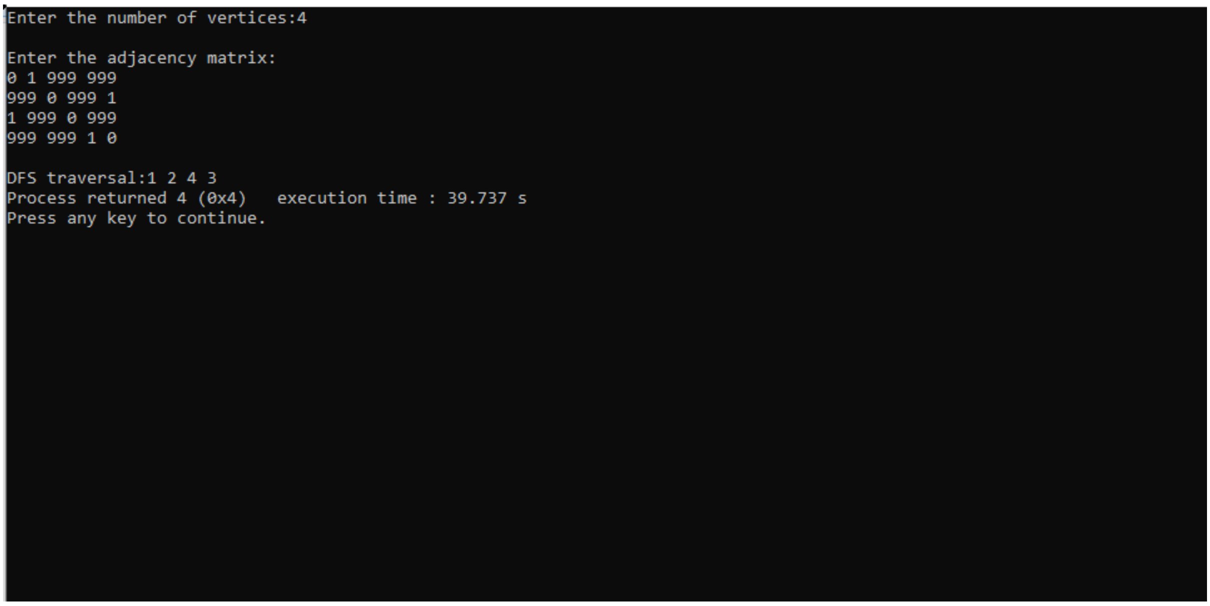
ADA LAB-1,2

1.DFS:

CODE:

```
#include <stdio.h>
int vis[10],a[10][10],n;
void dfs(int);

void main()
{
   int i,j;
   printf("Enter the number of vertices:");
   scanf("%d",&n);
   printf("\nEnter the adjacency matrix:\n");
   for(i=1;i<=n;i++)
   {
      for(j=1;j<=n;j++)
         scanf("%d",&a[i][j]);
   }

   printf("\nDFS traversal:");

   for(int i=1;i<=n;i++)
   {
      if(vis[i] == 0)
         dfs(i);
   }
}

void dfs(int v)
{
   int i;
   vis[v] = 1;
   printf("%d ",v);
   for(int i=1;i<=n;i++)
   {
      if(a[v][i]==1 && vis[i]==0)
         dfs(i);
   }
}
```

OUTPUT:



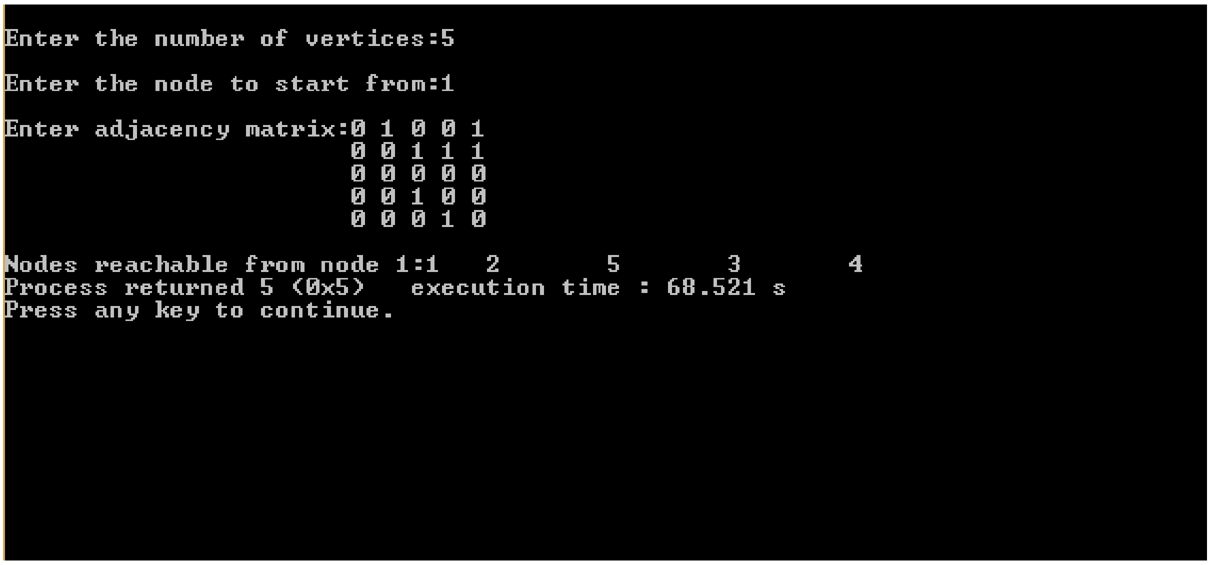
```
Enter the number of vertices:4

Enter the adjacency matrix:
0 1 999 999
999 0 999 1
1 999 0 999
999 999 1 0

DFS traversal:1 2 4 3
Process returned 4 (0x4)   execution time : 39.737 s
Press any key to continue.
```

2.BFS:

CODE:

```
#include<stdio.h>

void bfs(int);

int a[10][10],vis[10],n;// a is adjescency matrix ,n is no vertices

void main()

{

 int i,j,start;

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

 scanf("%d",&n);

 printf("\nEnter the node to start from:");

 scanf("%d",&start);

 printf("\nEnter adjacency matrix:");

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

 {

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

   {
    scanf("%d",&a[i][j]);
   }
 }
 //initialise all vertices to 0-not visited initially
 for(i=1;i<=n;i++)
 {
   vis[i]=0;
 }

 bfs(start);//call function bfs

}

void bfs(int v)
{
 int q[10],f=0,r=0,u,i;
 vis[v]=1;
 q[r]=v;
 printf("\nNodes reachable from node %d:",v);
 while(f<=r)
 {
    u=q[f];
    printf("%d\t",u);
   for(i=1;i<=n;i++)
   {
     if(a[u][i]==1&&vis[i]==0)
     {
       r=r+1;
       q[r]=i;
       vis[i]=1;
     }
    }

    f=f+1;

 }

}
```

OUTPUT:



```
Enter the number of vertices:5

Enter the node to start from:1

Enter adjacency matrix:0 1 0 0 1
                       0 0 1 1 1
                       0 0 0 0 0
                       0 0 1 0 0
                       0 0 0 1 0

Nodes reachable from node 1:1   2       5       3       4
Process returned 5 (0x5)   execution time : 68.521 s
Press any key to continue.
```

3.TOPOLOGICAL SORTING:

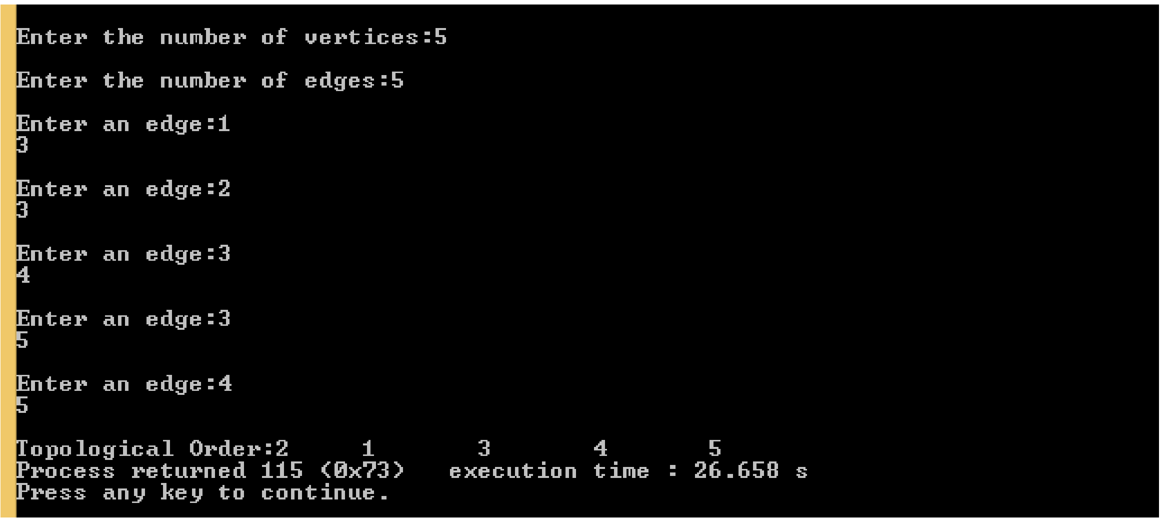
CODE:

```
#include<stdio.h>

#include<conio.h>


int a[10][10],n,exp[10],vis[10],J=0;

void dfs(int);


void main()

{

 int m,u,v,i,j;

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

 scanf("%d",&n);

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

```
{
 for(j=1;j<=n;j++)
  {
    a[i][j]=0;
  }
}
printf("\nEnter the number of edges:");
scanf("%d",&m);
for(i=1;i<=m;i++)
{
  printf("\nEnter an edge:");
  scanf("%d%d",&u,&v);
  a[u][v]=1;
}
for(i=1;i<=n;i++)
{
 vis[i]=0;
}
for(i=1;i<=n;i++)
{
 if(vis[i]==0)
 {
   dfs(i);
 }
}
printf("\nTopological Order:");
for(i=n-1;i>=0;i--)
{
  printf("%d\t",exp[i]);
}
getch();
```

```
}

void dfs(int v)
{
 int i;
 vis[v]=1;
 for(i=1;i<=n;i++)
 {
  if(a[v][i]==1&&vis[i]==0)
  {
    dfs(i);
  }
 }
exp[J++]=v;
}//dfs function
```

OUTPUT:



```
Enter the number of vertices:5

Enter the number of edges:5

Enter an edge:1
3

Enter an edge:2
3

Enter an edge:3
4

Enter an edge:3
5

Enter an edge:4
5

Topological Order:2      1       3       4       5
Process returned 115 (0x73)   execution time : 26.658 s
Press any key to continue.
```

BY,

SHANA DIYA SUJIT

1BM21CS196

SECTION:4D