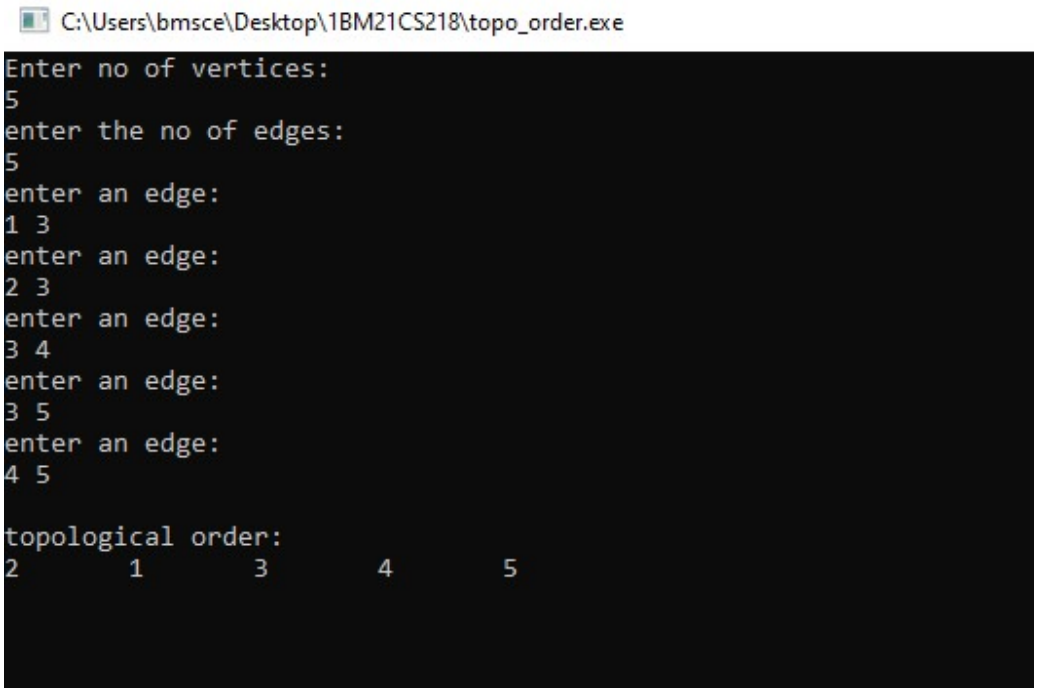
LAB-02

Topological ordering:

```
#include<stdio.h>
#include<conio.h>
void dfs(int);
int a[10][10],vis[10],exp[10],n,i,j=0;
void main()
{
 int i,j,m,u,v;
 printf("Enter no of vertices:\n");
 scanf("%d",&n);
 for(i=1;i<=n;i++)
 {
  for(j=1;j<=1;j++)
  {
   a[i][j]=0;
  }
 }
 printf("enter the no of edges:\n");
 scanf("%d",&m);
 for(i=1;i<=m;i++)
```

```
{
 printf("enter an edge:\n");
 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:\n");
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;
}
```


C:\Users\bmsce\Desktop\1BM21CS218\topo_order.exe
Enter no of vertices:
5
enter the no 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

Nodes from starting vertex using BSF traversal:

```
#include<stdio.h>
#include<conio.h>
void bfs(int);
int a[20][20],vis[20],n;
void main()
{
 int i,j,m,p,q,start;
 printf("Enter no of vertices:\n");
 scanf("%d",&n);
 for(i=1;i<=n;i++)
 {
  for(j=1;j<=n;j++)
  {
   a[i][j]=0;
  }
 }
 printf("enter the no of edges:\n");
 scanf("%d",&m);
 for(i=1;i<=m;i++)
```

```
{
 printf("enter an edge:\n");
 scanf("%d%d",&p,&q);
 a[p][q]=1;
}
for(i=1;i<=n;i++)
{
 vis[i]=0;
}
printf("\nenter the starting vertex:\n");
scanf("%d",&start);
printf("nodes reachable from starting vertex are:");
bfs(start);
getch();
}

void bfs(int v)
{
 int i,q[30],f=0,r=0,u;
 vis[v]=1;
 q[r]=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;
}
}
```

```
Enter no of vertices:
5
enter the no of edges:
6
enter an edge:
1 2
enter an edge:
1 5
enter an edge:
2 5
enter an edge:
2 4
enter an edge:
3 1
enter an edge:
3 4

enter the starting vertex:
1
nodes reachable from starting vertex are:1      2       5       4
```