Program 4

- DFS METHOD

```
#include<stdio.h>
int count=0;
int stack[10];
int output[10];
int top = -1;
int wow = 0;
void dfs(int a[10][10], int n, int visited[10], int current)
{
      int j,m;
    stack[++top] = current;
      visited[current]=1;
      for (j=0;j<n;j++)
      {
            if (a[current][j]==1 && visited[j]==0)
            {
                  dfs(a,n,visited,j);
            }
      }
    m = stack[top--];
    printf("%d ", m);
    output[wow++] = current;
}


int DFS(int a[10][10], int n)
{
  int visited[10],comp=0,i;

      for (i=0;i<n;i++)
      {
          visited[i] = 0;
      }


      printf("Pop order:\n");

      for (i=0;i<n;i++)
      {
              if (visited[i] == 0)
            {
```

```
                        dfs(a,n,visited,i);
                        comp++;
                }


        }

          if (comp > 1)
        {
                printf("\nThe graph is disconnected \n");
                printf("\nThe no. of components are:%d\n",comp);
          }
        else
        {
              printf("\nGraph is connected.\n");
        }

}

int main()
{
        int a[10][10],n,i,j;
        printf("Enter the no. of vertices:");
        scanf("%d",&n);


        printf("Enter the adjacency matrix:\n");
        for (i=0;i<n;i++)
        {
              for (j=0;j<n;j++)
              {
                    scanf("%d",&a[i][j]);
              }
        }



        DFS(a,n);

      printf("Topological sort : ");
        while(wow > 0){
              printf("%d ",output[--wow]);
        }
      printf("\n");
```

```
	return 0;
}
```

- Vertex Removal method

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
	int n;
	int a[10][10];
	int i,j,k,node;
	int in[10]={0};
	int v[10]={0};
	printf("Enter the value of n:\n");
	scanf("%d", &n);
	printf("Enter adj matrix:\n");
	for(i=1;i<=n;i++)
	{
		for(j=1;j<=n;j++)
		{
			scanf("%d", &a[i][j]);
			if(a[i][j]==1)
				in[j]++;
		}
	}
	printf("Topological order:\n");
	for(k=1;k<=n;k++)
	{
		for(i=1;i<=n;i++)
		{
			if(in[i] == 0 && v[i] == 0)
			{
				node=i;
				printf("%5d", node);
				v[node]=1;
				break;
			}
```

```
            }

            for(i=1;i<=n;i++)
            {
                  if(a[node][i]==1)
                  in[i]--;
            }

      }
}
```