Lab - 3

Topological Sort

23/5/24.

1. Source Removal Method.

```
#include <stdio.h>
#define v 100
int top=-1;
void indegree(int amatrix[v][v],int n, int in[v])
{
   for(int i=0; i<n; i++){
      for(int j=0; j<n; j++){
         if(amatrix[i][j]){
            in[j]++;
         }}}
}

void toposort(int amatrix[v][v] , int n)
{
   int in[v]={0};
   int topo[v];
   int k=0;
   int s[v]={0};
   indegree(amatrix,n,in);
   for(int i=0; i<n; i++)
   {
      if(in[i]==0){
         top++;
         s[top]=i;
      }}
   while(top!=-1)
   {
      int vertex=s[top];
      top--;
      topo[k++]=vertex;
```

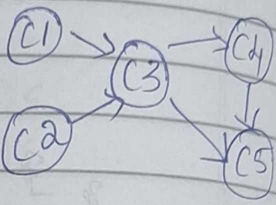
```
        for (int i=0; i<n; i++)
        {
            if (amatrix[vertex] [i])
            {
                in[i]--;
                if (in[i] = = 0)
                {
                    top++;
                    s[top] = i;
                }
            }
        }
    }
    if (k!=n)
    { printf("cycle exists"); }
    else{
     printf("The topological sort:");
    for (int i = 0; i<n; i++)
    {
        printf("%d", topo[i]+1);
    }
    }
}
int main()
{
    int amatrix[v] [v];
    int n;
    printf("enter the no of vertices:");
    scanf("%d", &n);
    printf("enter the adjacency matrix:\n");
    for(int i=0; i<n; i++)
    { for(int j=0; j<n ; j++)
        {
            scanf("%d", &amatrix[i] [j]);
    }   }
```

[illegible] 30/5/24

```
    toposort(a_matrix, n);
    return 0;
}
```

Output

```
enter no of vertices : 5
enter the adjacency matrix:
0 0 1 0 0
0 0 1 0 0
0 0 0 1 1
0 0 0 0 1
0 0 0 0 0
the topological sort: 2 1 3 4 5
```



2. using DFS

```
#include <stdio.h>
#define v 100
int j=0;
void dfs(int amatrix[v][v], int n, int
        visited[], int start, int res[])
{
   visited[start]=1;
   for(int i=0; i<n; i++)
   {
     if(amatrix[start][i]==1 && visited[i]==0)
     {
       dfs(amatrix, n, visited, i, res);
     }
   }
   res[j++]=start;
}

void toposort(int amatrix, int n)
{
   int visited[v]={0};
   int res[v];
   j=0;
   for(int i=0; i<n; i++)
   {
      if(visited[i]==0)
      {
        dfs(amatrix, n, visited, i, res);
      }
   }
   printf("the topological sort:");
```

```
    for (int i=n-1; i>=0; i--)
    {
        printf("%d", res[i]);
    }
};

[illegible] 20/5/24

int main()
{
    int amatrix[V][V];
    int n;
    printf("enter the no of vertices:");
    scanf("%d", &n);
    printf("enter the adjacency matrix:\n");
    for(int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            scanf("%d", &amatrix[i][j]);
        }
    }
    toposort(amatrix, n);
    return 0;
}
```

```
enter the no of vertices: 4
enter the adjacency matrix:
0 0 0 0
1 0 0 0
1 0 0 1
0 1 0 0
the topological sort: 2 3 1 0
```

$$\begin{tikzcd}
2 \arrow[r] \arrow[dr] & 0 & 1 \arrow[l] \\
 & 3 \arrow[ur] &
\end{tikzcd}$$