//Created By Ritwik Chandra Pandey
//On 7th Nov 2021
//Topological Sort

/*Topological sorting for Directed Acyclic Graph (DAG) is a linear ordering of vertices such that for every directed edge u → v, vertex u comes before v in the ordering. Topological sorting for a graph is not possible if the graph is not a DAG. Topological sorting is not possible if the DAG contains a cycle.Topological sorting of a directed acyclic graph is possible only if it has at least one **sink** vertex. **Sink vertex** is a vertex that has no outgoing edge. The first vertex in topological sorting is always a vertex with in-degree as 0 (a vertex with no incoming edges). Recollect that the number of edges pointing to a node is called the in-degree of the node.*/

```
#include <stdio.h>
int main() {
    int i,j,s,d,k,E,N,graph[10][10],indeg[10],flag[10],count=0;
    printf("Enter the number of vertices : ");
        scanf("%d",&N);
        printf("Enter the number of edges : ");
        scanf("%d",&E);
        for (i = 0 ; i<N; i++) {
                for (j = 0; j <N; j++) {
                        graph[i][j] = 0;
                }
        }
        for(i=1;i<=E;i++) {
                printf("Enter source : ");
                scanf("%d",&s);
                printf("Enter destination : ");
                scanf("%d",&d);
                if(s > N || d > N || s<=0 || d<=0) {
                        printf("Invalid index. Try again.\n");
                        i--;
                        continue;
                } else {
                        graph[s][d] = 1;
                }
        }
        //Write the code to print the topological order to match the output.
        for(i=0;i<N;i++){
                indeg[i] = 0;
```

```
            flag[i] = 0;
        }
        for(i=0;i<N;i++){
            for(j=0;j<N;j++){
                indeg[i] = indeg[i] + graph[j][i];
            }
        }
        printf("The topological order is:");
        while(count<N){
            for(int k=0;k<N;k++){
                if(indeg[k]==0 && flag[k]==0){
                    printf("%d ",(k+1));
                    flag[k] = 1;
                }
                for(i=0;i<N;i++){
                    if(graph[i][k] == 1){
                        indeg[k]--;
                    }
                }

            }
            count++;
        }
        printf("\n");
}
```