# LAB 3:TOPOLOGICAL SORT

```
#include <stdio.h>
#include <stdlib.h>

#define MAX_NODES 100

void topologicalSort(int n, int adjMatrix[MAX_NODES][MAX_NODES]) {
    int inDegree[MAX_NODES] = {0};
    int sorted[MAX_NODES];
    int index = 0;

    // Compute in-degree of each node
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (adjMatrix[i][j] == 1) {
                inDegree[j]++;
            }
        }
    }

    // Find all nodes with no incoming edges
    int stack[MAX_NODES];
    int top = -1;
    for (int i = 0; i < n; i++) {
        if (inDegree[i] == 0) {
            stack[++top] = i;
        }
    }

    // Perform topological sort
    while (top >= 0) {
        int u = stack[top--];
        sorted[index++] = u;

        // Reduce the in-degree of adjacent nodes
        for (int v = 0; v < n; v++) {
            if (adjMatrix[u][v] == 1) {
                inDegree[v]--;
                if (inDegree[v] == 0) {
                    stack[++top] = v;
                }
            }
        }
    }

    // Check if there was a cycle in the graph
    if (index != n) {
        printf("The graph has a cycle!\n");
```

```
        return;
    }

    // Print the topologically sorted order
    printf("Topological Sort: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", sorted[i]);
    }
    printf("\n");
}

int main() {
    int n, e;
    int adjMatrix[MAX_NODES][MAX_NODES] = {0};

    printf("Enter the number of nodes: ");
    scanf("%d", &n);

    printf("Enter the number of edges: ");
    scanf("%d", &e);

    printf("Enter the edges (source destination):\n");
    for (int i = 0; i < e; i++) {
        int src, dest;
        scanf("%d %d", &src, &dest);
        adjMatrix[src][dest] = 1;
    }

    topologicalSort(n, adjMatrix);

    return 0;
}
```

## OUTPUT:

```
Enter the number of nodes: 6
Enter the number of edges: 6
Enter the edges (source destination):
5 0
5 2
2 3
4 0
4 1
3 1
Topological Sort: 5 2 3 4 1 0
```