```
#include <stdio.h>
#include <stdbool.h>
#define MAX_VERTICES 100
int graph[MAX_VERTICES][MAX_VERTICES];
int visited[MAX_VERTICES];
int stack[MAX_VERTICES];
int top = -1;
void push(int vertex) {
    stack[++top] = vertex;
}
int pop() {
    return stack[top--];
}
void dfs(int vertex, int numVertices) {
    visited[vertex] = 1;
    for (int i = 0; i < numVertices; ++i) {
        if (graph[vertex][i] && !visited[i]) {
            dfs(i, numVertices);
        }
    }
    push(vertex);
}
void topologicalSort(int numVertices) {
    for (int i = 0; i < numVertices; ++i) {
        visited[i] = 0;
    }
    for (int i = 0; i < numVertices; ++i) {
        if (!visited[i]) {
            dfs(i, numVertices);
        }
    }
```

```
    printf("Topological Sort: ");
    while (top >= 0) {
        printf("%d ", pop());
    }   printf("\n");
}
int main() {
    int numVertices, numEdges;
    printf("Enter the number of vertices: ");
    scanf("%d", &numVertices);
    for (int i = 0; i < numVertices; ++i) {
        for (int j = 0; j < numVertices; ++j) {
            graph[i][j] = 0;       }
    }
    printf("Enter the number of edges: ");
    scanf("%d", &numEdges);
    for (int i = 0; i < numEdges; ++i) {
        int src, dest;
        printf("Enter edge %d (source destination): ", i + 1);
        scanf("%d %d", &src, &dest);
        graph[src][dest] = 1;
    }
    topologicalSort(numVertices);
    return 0;
}
```

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