

**NAME: Haresh Kumar N L (192425009)**

**COURSE NAME: DATA STRUCTURES FOR MODERN COMPUTING SYSTEMS**

**COURSE CODE: CSA0302**

Experiment 38: Topological Sorting

Code:

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

#define MAX 100

int adj[MAX][MAX]; // adjacency matrix
int visited[MAX];
int stack[MAX];
int top = -1;
int n; // number of vertices

void push(int v) {
    stack[++top] = v;
}

int pop() {
    if(top == -1) return -1;
    return stack[top--];
}

void dfs(int v) {
    visited[v] = 1;
    for(int i = 0; i < n; i++) {
        if(adj[v][i] && !visited[i])
            dfs(i);
    }
}
```

```
    }

    push(v); // push after visiting all neighbors

}
```

```
void topologicalSort() {

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

    for(int i = 0; i < n; i++)
        if(!visited[i])
            dfs(i);

    printf("Topological Order: ");

    while(top != -1)
        printf("%d ", pop());
        printf("\n");

}
```

```
int main() {

    int e, u, v;

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

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

    // initialize adjacency matrix

    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            adj[i][j] = 0;
```

```
printf("Enter edges (source destination):\n");
for(int i = 0; i < e; i++) {
    scanf("%d %d", &u, &v);
    adj[u][v] = 1; // directed edge u -> v
}

topologicalSort();

return 0;
}
```

Output:

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

```
==== Code Execution Successful ===
```