LAB-1

1. Write program to obtain the Topological ordering of vertices in a given digraph.

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

#define MAX 100

typedef struct {
    int items[MAX];
    int front, rear;
} Queue;

void initQueue(Queue *q) {
    q->front = -1;
    q->rear = -1;
}

int isEmpty(Queue *q) {
    return q->front == -1;
}

void enqueue(Queue *q, int value) {
    if (q->rear == MAX - 1) {
        printf("Queue Overflow\n");
        return;
    }
    if (q->front == -1) q->front = 0;
    q->items[++q->rear] = value;
}

int dequeue(Queue *q) {
    if (isEmpty(q)) {
        printf("Queue Underflow\n");
        return -1;
    }
    int item = q->items[q->front];
    if (q->front >= q->rear) {
        q->front = q->rear = -1;
    } else {
        q->front++;
    }
    return item;
```

```
}

void topologicalSort(int n, int graph[MAX][MAX]) {
    int in_degree[MAX] = {0};
    int i, j;

    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            if (graph[i][j] == 1) {
                in_degree[j]++;
            }
        }
    }

    Queue q;
    initQueue(&q);

    for (i = 0; i < n; i++) {
        if (in_degree[i] == 0) {
            enqueue(&q, i);
        }
    }

    printf("Topological Sort Order: ");
    while (!isEmpty(&q)) {
        int u = dequeue(&q);
        printf("%d ", u);

        for (j = 0; j < n; j++) {
            if (graph[u][j] == 1) {
                in_degree[j]--;
                if (in_degree[j] == 0) {
                    enqueue(&q, j);
                }
            }
        }
    }
    printf("\n");
}

int main() {
    int n, i, j;
    printf("Enter number of vertices: ");
    scanf("%d", &n);

    int graph[MAX][MAX];

    printf("Enter adjacency matrix:\n");
```

```
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            scanf("%d", &graph[i][j]);
        }
    }

    topologicalSort(n, graph);

    return 0;
}
```

OUTPUT: