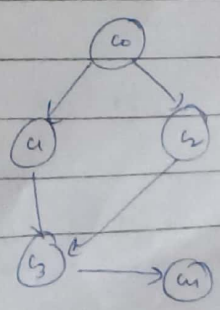
① Topological Sort Using Source Removal Method

```
#include <stdio.h>
void toposort (int a[10][10], int n)
{
    int i, j, t[100], u, indegree[100]={0}, k=0,
    v, s[100], top = -1;

    for (j=0; j<n; j++)
    {
        for (i=0; i<n; i++)
        {
            indegree[j] += a[i][j];
        }
    }

    while (top != -1)
    {
        u = s[top];
        top--;
        t[k++] = u;
        for (j=0; j<n; j++)
        {
            if (a[u][j])
            {
                v=j;
                indegree[v]--;
                if (indegree[v]==0)
                {
                    top++;
                    s[top] = v;
                }
            }
        }
    }
```

```
printf("Topological Order : \n");
for(i=0; i<n; i++)
{
    printf("%d -> ", T[i]);
}
}
int main()
{
    int a[10][10], n, i, j;
    printf("Enter no. of vertices");
    scanf("%d", &n);
    printf("Enter adjacency matrix\n");
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
        {
            scanf("%d", &a[i][j]);
        }
    }
    topsort(a, n);
    return 0;
}
```

→ output

```
Enter no. of vertices: 5
Enter adjacency matrix
0 1 1 0 0
0 0 0 1 0
0 0 0 1 0
0 0 0 0 1
0 0 0 0 0
```



Topological order

0 → 2 → 1 → 3 → 4

(2) Topological Sort Using DFS

```
#include <stdio.h>
int a[20][20], n, res[20], visited[20], j = 0;
void DFS (int v)
{
    visited[v] = 1
    for (int i = 0; i < n; i++)
    {
        if (a[i][v] == 1 && visited[i] == 0)
        DFS (i);
    }
    res[j++] = v;
}
void main()
{
    int i, j, v;
    printf ("Enter no. of vertices");
    scanf ("%d", &n);
    printf ("Enter adjacency matrix:\n");
    for (int i=0; i<n; i++)
    {
        for (j=0; j<n; j++)
            scanf ("%d", &a[i][j]);
    }
    for (int i = 0; i < n; i++)
        visited[i] = 0;
    for (v=0; v<n-1; v++)
```

```
}
    if (visited[v] == 0)
{
        DFS(v);
for (i = 0; i < n; i++)
        printf("%d ", res[i]);
}
```