23-5-24

Lab - 4

Topological sorting using source removal algorithm

```
#include <stdio.h>
#include <stdlib.h>
void topo (int a[10][10], int n);

int main()
{
    int a[10][10], n;
    printf("Enter the number of vertices: \n");
    scanf("%d", &n);
    printf("Enter the adjacency matrix of the graph: \n");
    for(int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
            scanf("%d", &a[i][j]);
    }
    topo (a, n);
    return 0;
}

void topo (int a[10][10], int n)
{
    int indegree[10], s[10], T[10], top = -1;
    for (int i=0; i<n; i++)
    {
        int sum = 0;
        for (int j=0; j<n; j++)
            sum += a[j][i];
        indegree[i] = sum;
        if (indegree[i] == 0)
        {
            top++;
            s[top] = i;
        }
    }
}
```

```
int idx = 0;
while (top != -1)
{
    int u = s[top];
    top--;
    T[idx++] = u;
    for(int j=0; j<n; j++)
    {
        if (a[u][j] == 1)
        {
            indegree[j]--;
            if (indegree[j] == 0)
                top++;
                s[top] = j;
        }
    }
}
printf("The jobs that need to be executed in order are: \n");
for(int i = 0; i<n; i++)
    printf("%d \t", T[i]);
printf("\n");
}
```

Output:

Enter the number of vertices: 4

Enter the adjacency matrix of the graph:

0 0 0 0

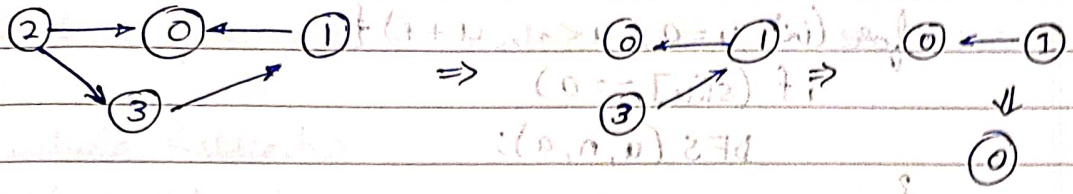
1 0 0 0

1 0 0 1

0 1 0 0

The jobs that need to be executed in order are

2 3 1 0

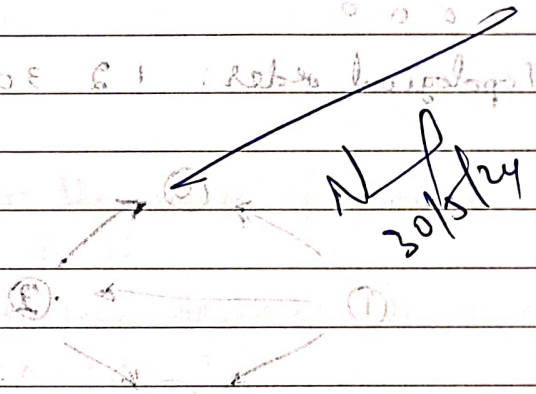


2) Topological sorting using DFS algorithm

```
#include <stdio.h>
#include <stdlib.h>
#define maxv 100

int s[maxv] = {0};
int res[maxv];
int j = 0;
void DFS (int u, int n, int a[maxv][maxv]) {
    s[u] = 1;
    for (int v=0; v<n; v++){
        if (a[u][v] == 1 && s[v] == 0) {
            DFS (v,n, a);
        }
    }
    res[j++] = u;
}

int main()
{
    int n;
    printf("Enter the number of vertices: \n");
    scanf("%d", &n);
    int a[maxv][maxv];
    printf("Enter the adjacency matrix : \n");
    for (int i=0; i<n; i++){
        for (int j = 0; j<n; j++){
            scanf("%d", &a[i][j]);
        }
    }
}
```



```
    for (int u=0; u<n; u++){
        if (s[u]==0)
            DFS(u, n, a);
    }
    printf("Topological order: ");
    for (int i=j-1; i>=0; i--) {
        printf("%d ", res[i]);
    }
    printf("\n");
    return 0;
}
```

Output:

Enter the number of vertices: 4

Enter the adjacency matrix:

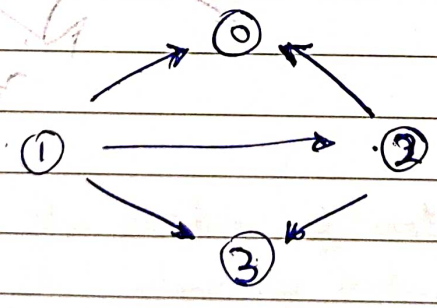
0 0 0 0

1 0 1 1

1 0 0 1

0 0 0 0

Topological order: 1 2 3 0



I(1) (4)

O(2) (1)

2(3) (3)

3(4) (2)

⇒ Popping sequence ⇒ 0 3 2 1

Topological sort ⇒ 1 2 3 0.

(Reverse)