LAB 5

Aniket sambher

Reg no-190905466

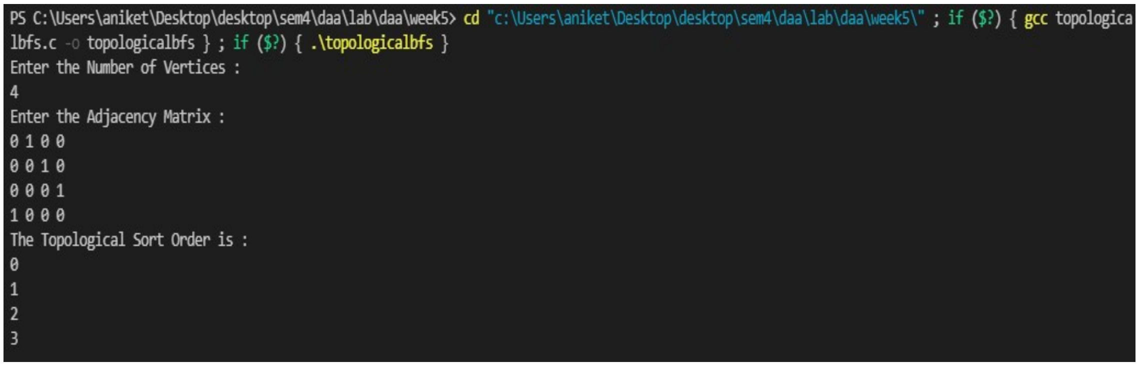
Section A

Roll no-58

Q1.(i)Topological sort using dfs

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

int adj[50][50], visited[50], stack[100],n,t=0;

void dfs(int v)
{
	visited[v]=1;

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

	stack[t++]=v;
}
```

```
void printStack()
{
    for(int i=n-1;i>=0;i--)
    {
        printf("%d\n",stack[i]);
    }

    printf("\n");
}

int main()
{
    printf("Enter the Number of Vertices : \n");
    scanf("%d", &n);

    printf("Enter the Adjacency Matrix : \n");

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

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

                }

        }


        printf("The Topological Sort Order is :\n");

        printStack();



        return 0;

}
```



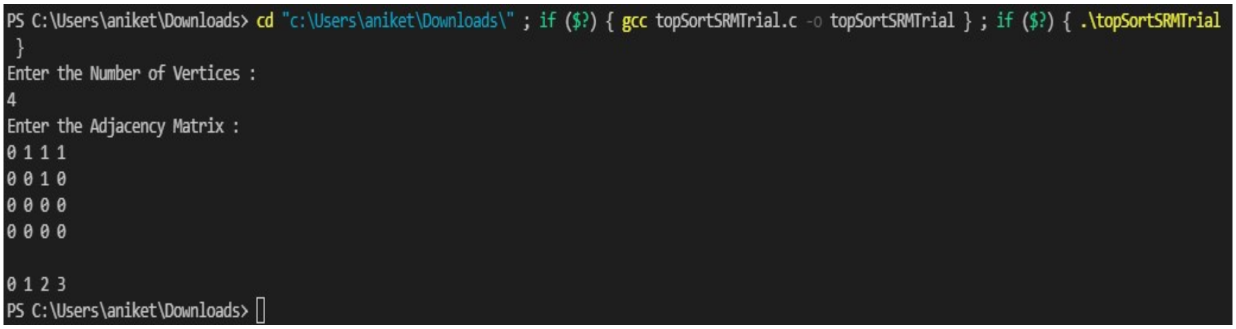
```
PS C:\Users\aniket\Desktop\desktop\sem4\daa\lab\daa\week5> cd "c:\Users\aniket\Desktop\desktop\sem4\daa\lab\daa\week5\" ; if ($?) { gcc topologica
lbfs.c -o topologicalbfs } ; if ($?) { .\topologicalbfs }
Enter the Number of Vertices :
4
Enter the Adjacency Matrix :
0 1 0 0
0 0 1 0
0 0 0 1
1 0 0 0
The Topological Sort Order is :
0
1
2
3
```

PTO

Q1(ii)Topological sort using source removal technique

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

int queue[100], k_1 = 0, k = 0, arr[100][100], n, indegree[100],counter = 0;

int main()
{
  int i,j;
        printf("Enter the Number of Vertices : \n");
        scanf("%d", &n);

        printf("Enter the Adjacency Matrix : \n");

        for(i = 0; i<n; i++)
        {
               for(j = 0; j<n; j++)
               {
               scanf("%d", &arr[i][j]);
               }
        }
  for(i=0; i<n; ++i){
    indegree[i]=0;
  }
  for(i = 0; i < n; ++i){
    for(j = 0; j < n; ++j){
      if(arr[j][i]==1){
        indegree[i]++;
        }
    }
}
while(1){
    for(i =0 ;i<n; ++i){
        if(indegree[i]==0){
            indegree[i]=-1;
            break;
        }
    }
    if(indegree[i]==-1){
        queue[k++] = i;
        for(j = 0; j<n; ++j){
            if(arr[i][j]==1){
                indegree[j]--;
            }
        }
    }
    counter++;
    if(counter >= n){
        break;
    }

}
printf("\n");
for(i=0;i<n;++i){
    printf("%d ",queue[i]);
}
printf("\n");
```

```
    return 0;

}
```



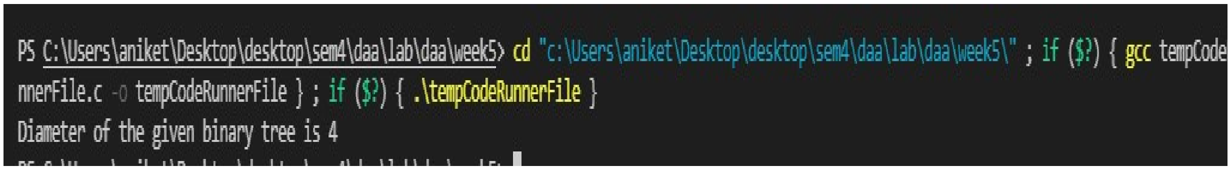
```
PS C:\Users\aniket\Downloads> cd "c:\Users\aniket\Downloads\" ; if ($?) { gcc topSortSRMTrial.c -o topSortSRMTrial } ; if ($?) { .\topSortSRMTrial
 }
Enter the Number of Vertices :
4
Enter the Adjacency Matrix :
0 1 1 1
0 0 1 0
0 0 0 0
0 0 0 0

0 1 2 3
PS C:\Users\aniket\Downloads> 
```

PTO

Q2.Find diameter of a binary tree

```
#include <stdio.h>

#include <stdlib.h>

struct node {

            int val;

            struct node *left, *right;

};

struct node* newNode(int value)

{

            struct node* node

                        = (struct node*)malloc(sizeof(struct node));

            node->val = value;

            node->left = NULL;

            node->right = NULL;

            return (node);

}

int max(int a, int b)

{

   return (a > b) ? a : b;

}

int height(struct node* node)

{

            if (node == NULL)

                        return 0;

            return 1 + max(height(node->left), height(node->right));

}

int diameter(struct node* tree)

{

            if (tree == NULL)
```

```
		return 0;
	int lheight = height(tree->left);
	int rheight = height(tree->right);
	int ldiam = diameter(tree->left);
	int rdiam = diameter(tree->right);
	return max(lheight + rheight + 1, max(ldiam, rdiam));
}
int main()
{
	struct node* root = newNode(1);
	root->left = newNode(2);
	root->left->left = newNode(4);
	root->left->right = newNode(5);
	root->left->right->left= newNode(6);
	root->left->right->right = newNode(7);



	printf("Diameter of the given binary tree is %d\n",
		diameter(root));
	return 0;
}
```



```
PS C:\Users\aniket\Desktop\desktop\sem4\daa\lab\daa\week5> cd "c:\Users\aniket\Desktop\desktop\sem4\daa\lab\daa\week5\" ; if ($?) { gcc tempCode
nnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Diameter of the given binary tree is 4
```