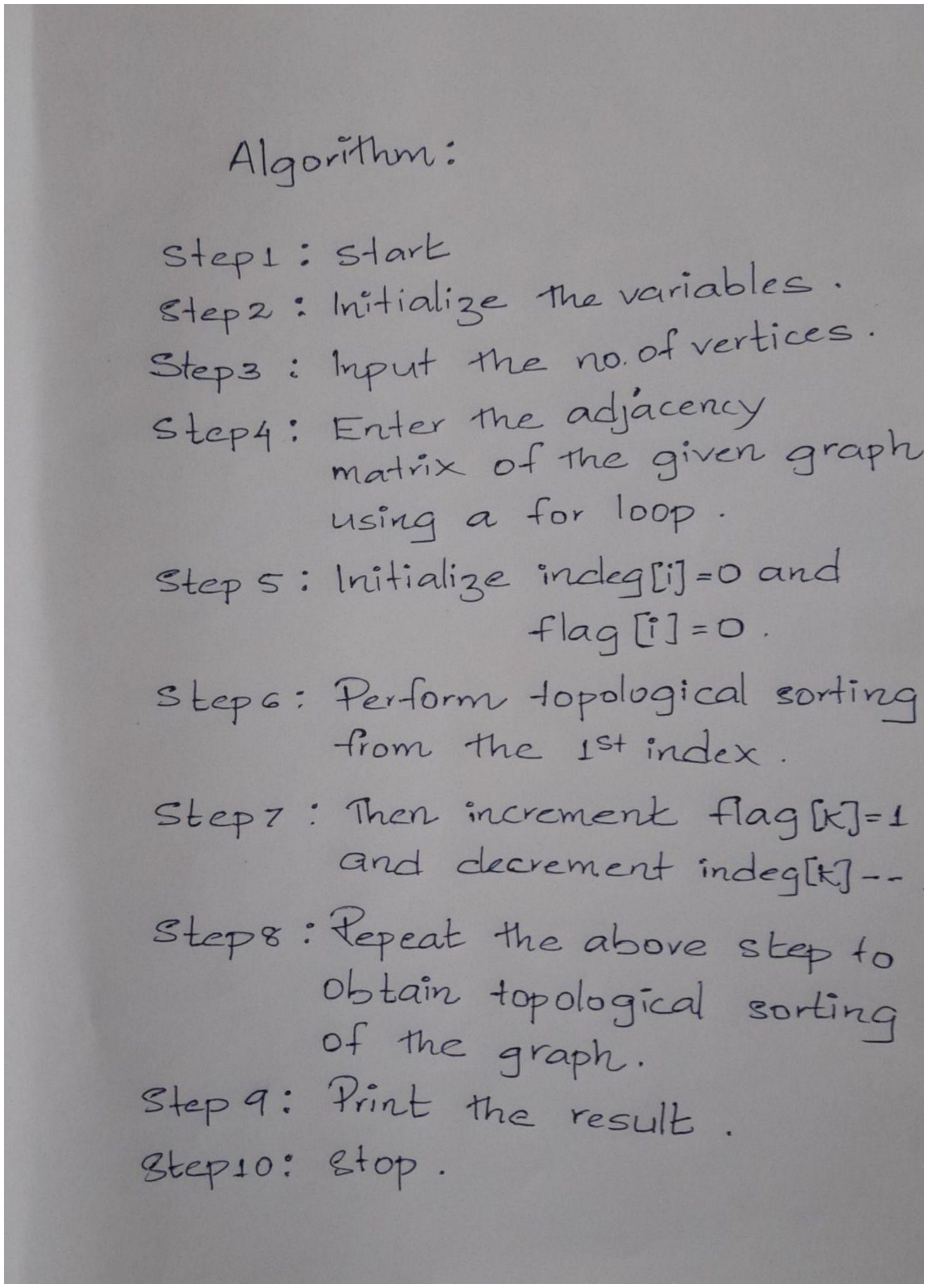
# DATASTRUCTURE LAB

Anilamol Chacko
S1MCA
TKM20MCA-2009

**QUESTION-1**

Consider a directed acyclic graph G
Develop a program to implement topological sorting.

**Algorithm:**



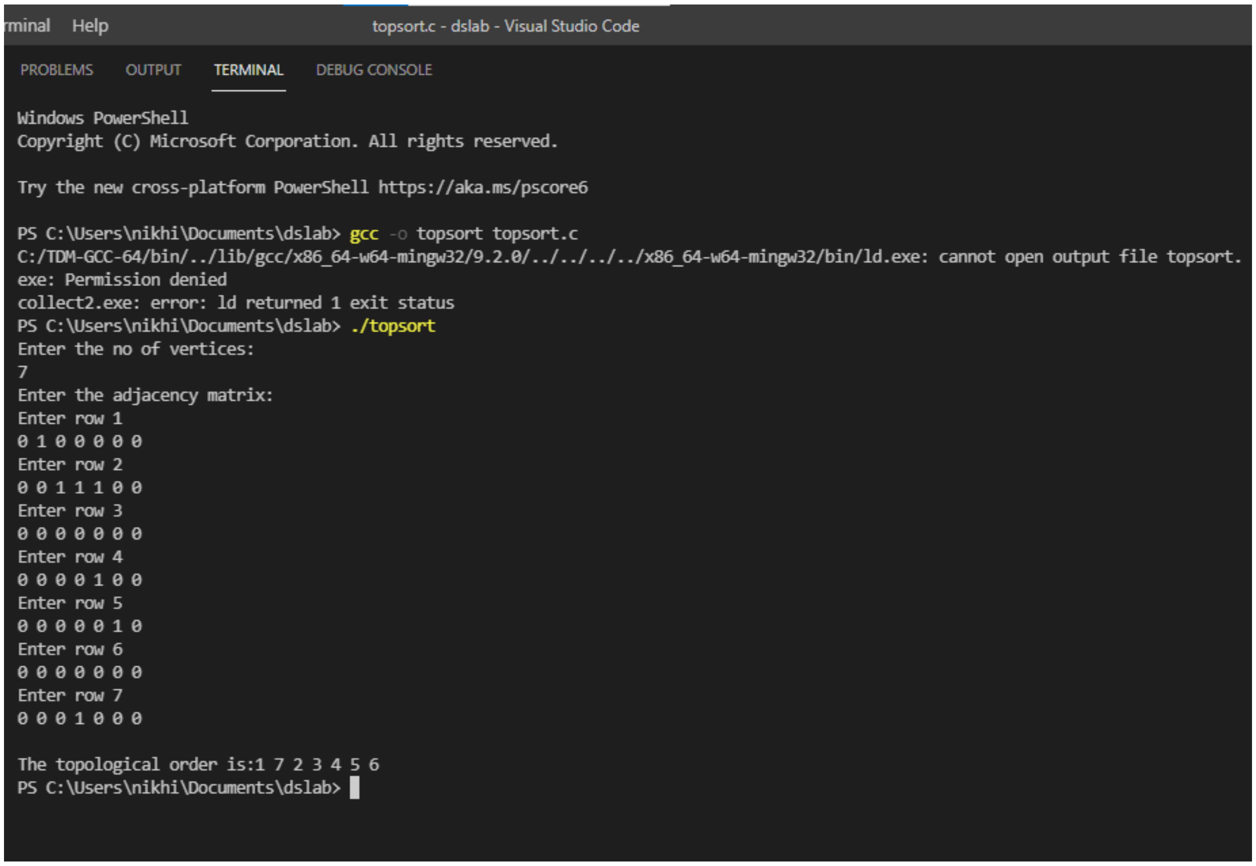
Algorithm:

Step1 : Start
Step2 : Initialize the variables.
Step3 : Input the no. of vertices.
Step4 : Enter the adjacency matrix of the given graph using a for loop.
Step 5 : Initialize indeg[i]=0 and flag[i]=0.
Step6 : Perform topological sorting from the 1st index.
Step7 : Then increment flag[k]=1 and decrement indeg[k]--
Step8 : Repeat the above step to obtain topological sorting of the graph.
Step 9 : Print the result.
Step10 : Stop.

**Program Code**

```
#include <stdio.h>

int main(){
            int n,a[10][10],indeg[10],flag[10],count=0;
    char i,j,k;

            printf("Enter the no of vertices:\n");
            scanf("%d",&n);

            printf("Enter the adjacency matrix:\n");
            for(i=0;i<n;i++){
                        printf("Enter row %c\n",i+1);
                        for(j=0;j<n;j++)
                                    scanf("%c",&a[i][j]);
            }

            for(i=0;i<n;i++){
        indeg[i]=0;
        flag[i]=0;
    }

    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            indeg[i]=indeg[i]+a[j][i];

    printf("\nThe topological order is:");

    while(count<n){
        for(k=0;k<n;k++){
            if((indeg[k]==0) && (flag[k]==0)){
                printf("%c ",(k+1));
                flag [k]=1;
            }

            for(i=0;i<n;i++){
                if(a[i][k]==1)
                    indeg[k]--;
            }
        }

        count++;
    }

    return 0;
}
```

## Output



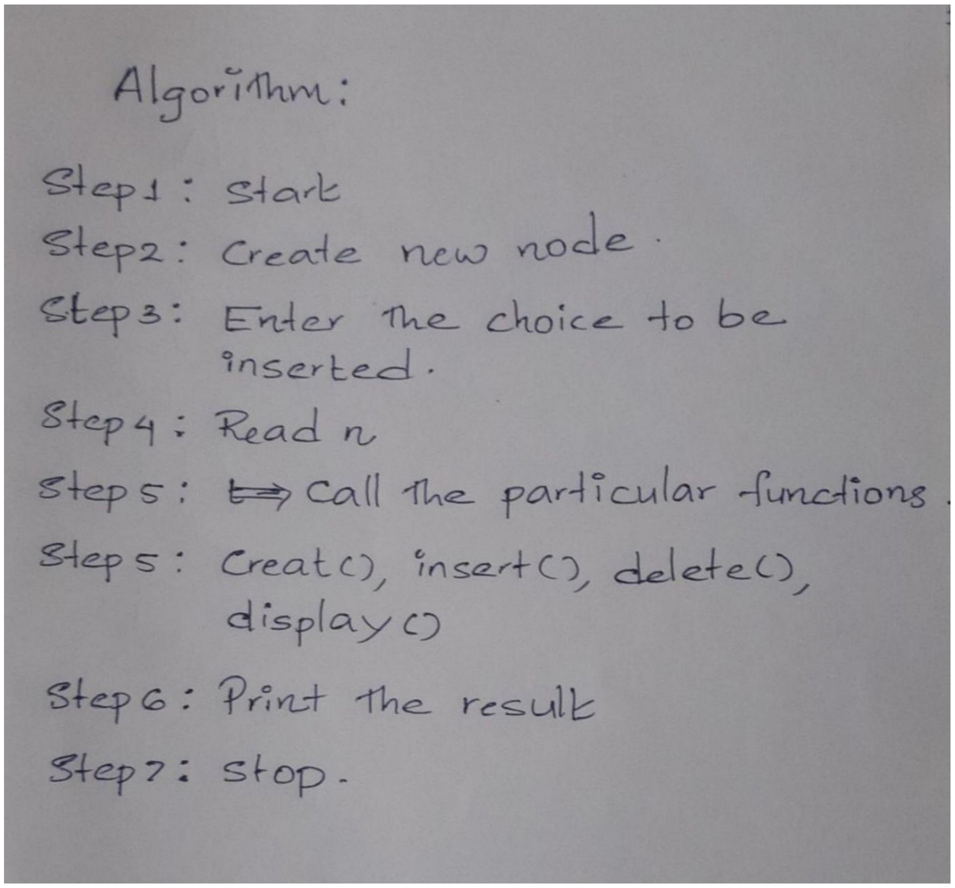
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\nikhi\Documents\dslab> gcc -o topsort topsort.c
C:/TDM-GCC-64/bin/../lib/gcc/x86_64-w64-mingw32/9.2.0/../../../../x86_64-w64-mingw32/bin/ld.exe: cannot open output file topsort.
exe: Permission denied
collect2.exe: error: ld returned 1 exit status
PS C:\Users\nikhi\Documents\dslab> ./topsort
Enter the no of vertices:
7
Enter the adjacency matrix:
Enter row 1
0 1 0 0 0 0 0
Enter row 2
0 0 1 1 1 0 0
Enter row 3
0 0 0 0 0 0 0
Enter row 4
0 0 0 0 1 0 0
Enter row 5
0 0 0 0 0 1 0
Enter row 6
0 0 0 0 0 0 0
Enter row 7
0 0 0 1 0 0 0

The topological order is:1 7 2 3 4 5 6
PS C:\Users\nikhi\Documents\dslab>
```

**QUESTION-2**

Write a program for creating doubly linked list and perform the following operations.

1. Insertion an element at a particular position.
2. Search an element.
3. Delete an element at the end of the list.

**Algorithm**



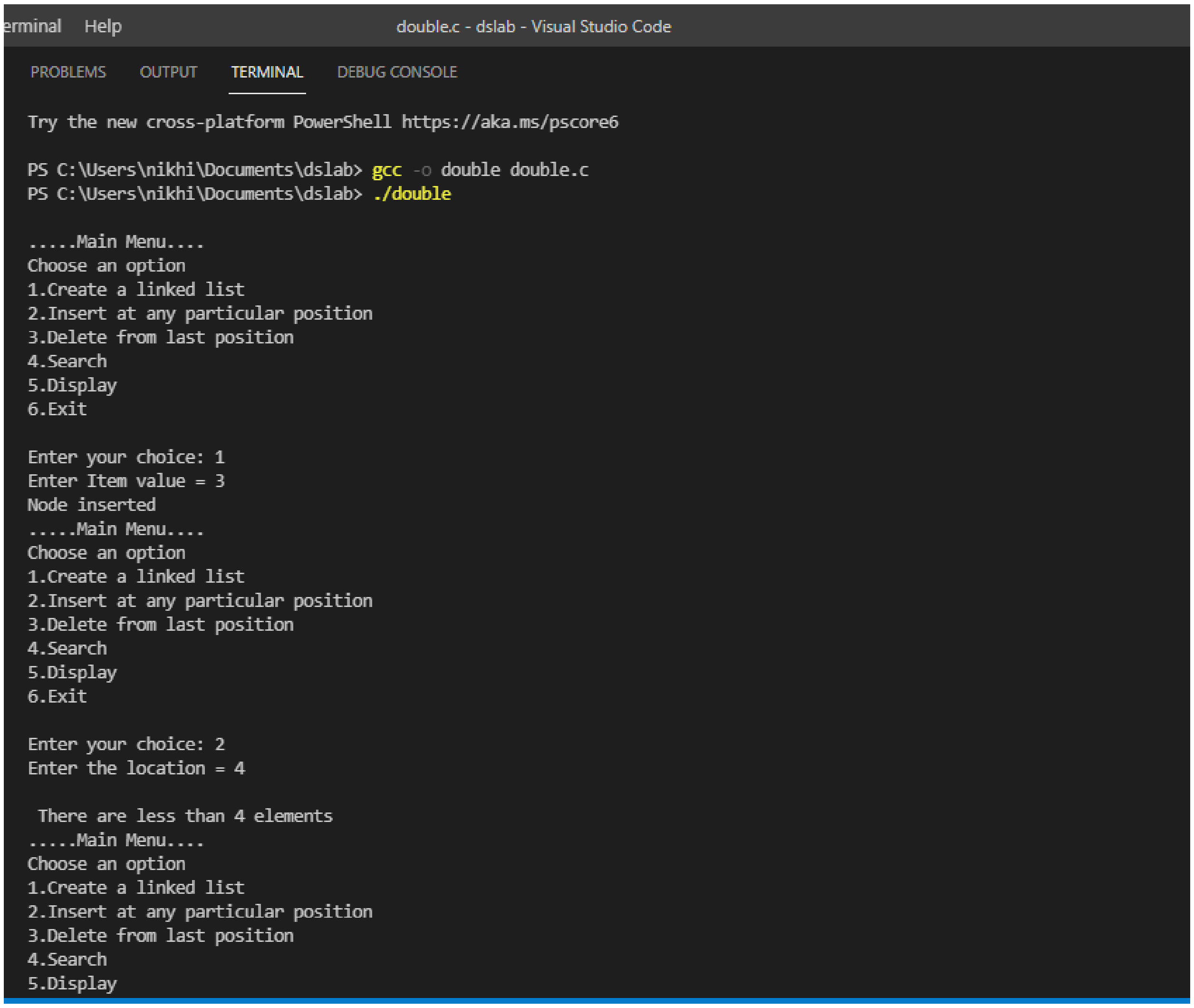
Algorithm:

```
Step1 : Start
Step2 : Create new node.
Step3 : Enter the choice to be
        inserted.
Step4 : Read n
Step5 : ⟹ Call the particular functions.
Step5 : Creat(), insert(), delete(),
        display()
Step6 : Print the result
Step7 : stop.
```

**Program code:**

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
    struct node *prev;
    struct node *next;
    int data;
};
struct node *head;
void create();
void insertion_particular();
void deletion_end();
void display();
void search();
void main()
{
    int choice = 0;
    while (choice != 9)
    {
        printf("\n.....Main Menu....");
        printf("\nChoose an option");
        printf("\n1.Create a linked list");
        printf("\n2.Insert at any particular position");
        printf("\n3.Delete from last position");
        printf("\n4.Search");
        printf("\n5.Display");
        printf("\n6.Exit\n");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            create();
            break;
        case 2:
            insertion_particular();
            break;
        case 3:
            deletion_end();
            break;
        case 4:
            search();
            break;
        case 5:
            display();
            break;
        case 6:
            exit(0);
            break;
        default:
            printf("Please enter valid choice..");
        }
    }
}
void create()
{
    struct node *ptr;
    int item;
    ptr = (struct node *)malloc(sizeof(struct node));
    if (ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("Enter Item value = ");
        scanf("%d", &item);

        if (head == NULL)
        {
            ptr->next = NULL;
            ptr->prev = NULL;
            ptr->data = item;
            head = ptr;
        }
        else
        {
            ptr->data = item;
            ptr->prev = NULL;
            ptr->next = head;
            head->prev = ptr;
            head = ptr;
        }
        printf("Node inserted");
    }
}

void insertion_particular()
{
    struct node *ptr, *temp;
    int item, loc, i;
    ptr = (struct node *)malloc(sizeof(struct node));
    if (ptr == NULL)
    {
        printf("\n OVERFLOW");
    }
    else
    {
        temp = head;
        printf("Enter the location = ");
        scanf("%d", &loc);
        for (i = 0; i < loc-1; i++)
        {
            temp = temp->next;
            if (temp == NULL)
            {
                printf("\n There are less than %d elements", loc);
                return;
            }
        }
        printf("Enter value = ");
        scanf("%d", &item);
        ptr->data = item;
        ptr->next = temp->next;
        ptr->prev = temp;
        temp->next = ptr;
        temp->next->prev = ptr;
        printf("\nnode inserted\n");
    }
}

void deletion_end()
{
    struct node *ptr;
    if (head == NULL)
    {
        printf("\n UNDERFLOW");
    }
    else if (head->next == NULL)
    {
        head = NULL;
        free(head);
        printf("\nnode deleted");
    }
    else
    {
        ptr = head;
        while (ptr->next != NULL)
        {
            ptr = ptr->next;
        }
        ptr->prev->next = NULL;
        free(ptr);
        printf("\nnode deleted");
    }
}
void display()
{
    struct node *ptr;
    printf("\n printing values...");
    ptr = head;
    while (ptr != NULL)
    {
        printf("%d\n", ptr->data);
        ptr = ptr->next;
    }
}
void search()
{
    struct node *ptr;
    int item, i = 0, flag;
    ptr = head;
    if (ptr == NULL)
    {
        printf("\nEmpty List");
    }
    else
    {
        printf("\nEnter item which you want to search?");
        scanf("%d", &item);
        while (ptr != NULL)
        {
            if (ptr->data == item)
            {
                printf("\nitem found at location %d ", i + 1);
                flag = 0;
                break;
            }
            else
            {
                flag = 1;
            }
            i++;
            ptr = ptr->next;
        }
        if (flag == 1)
        {
            printf("\nItem not found");
        }
    }
}
```

**Output:**


erminal Help double.c - dslab - Visual Studio Code
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\nikhi\Documents\dslab> gcc -o double double.c
PS C:\Users\nikhi\Documents\dslab> ./double

.....Main Menu....
Choose an option
1.Create a linked list
2.Insert at any particular position
3.Delete from last position
4.Search
5.Display
6.Exit

Enter your choice: 1
Enter Item value = 3
Node inserted
.....Main Menu....
Choose an option
1.Create a linked list
2.Insert at any particular position
3.Delete from last position
4.Search
5.Display
6.Exit

Enter your choice: 2
Enter the location = 4

 There are less than 4 elements
.....Main Menu....
Choose an option
1.Create a linked list
2.Insert at any particular position
3.Delete from last position
4.Search
5.Display
```

Gitlink:
https://github.com/anilachacko/DSlab