20MCA135

DATA STRUCTURES LAB

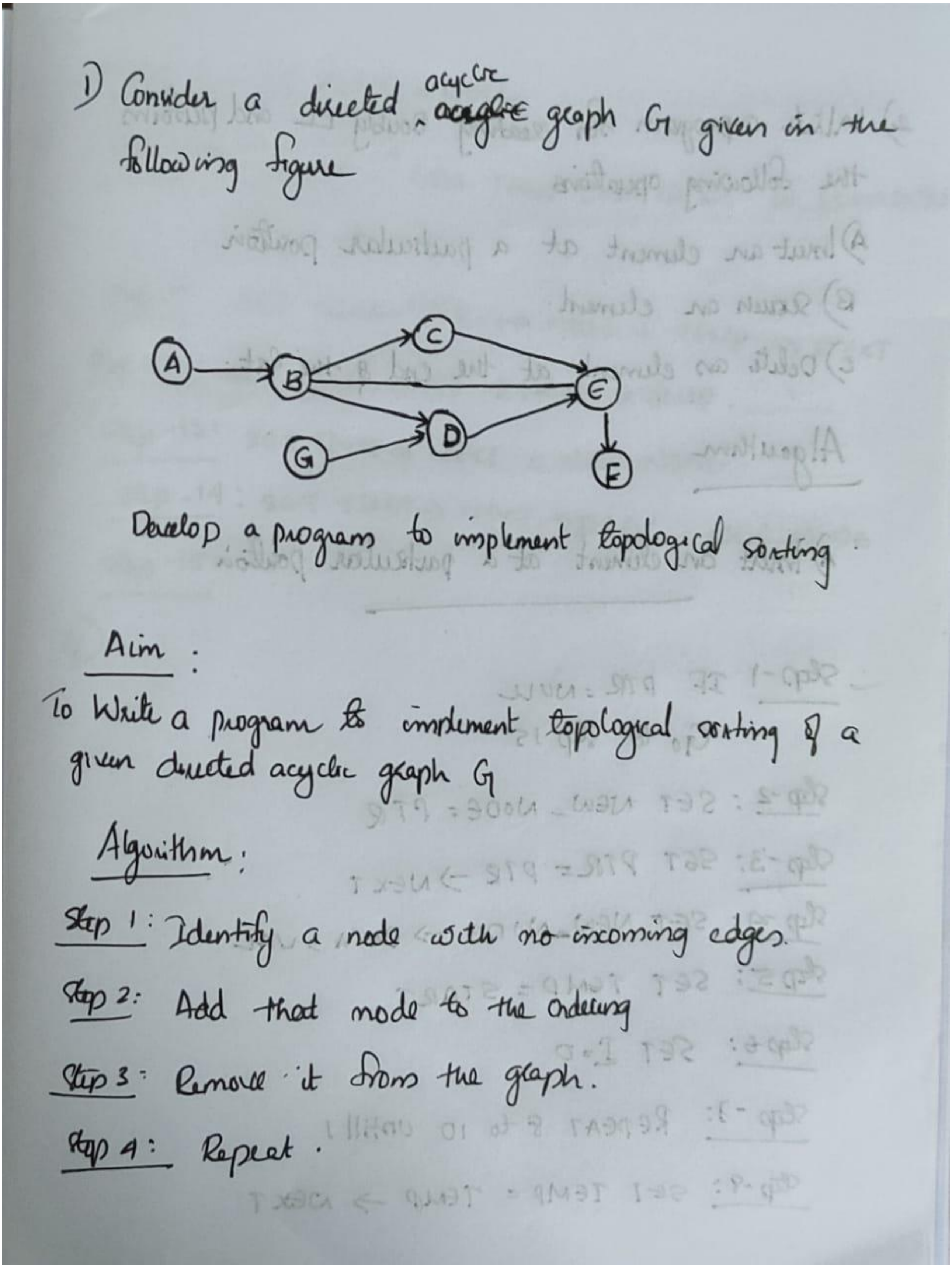
HARI KRISHNAN S R

TKM20MCA-2019

Roll no 219

TKMCE

GITHUB: https://github.com/harikrsr/Data-Structures



1) Consider a directed acyclic ~~acyglic~~ graph G given in the following figure

Develop a program to implement topological sorting .

Aim :

To Write a program to implement topological sorting of a given directed acyclic graph G

Algorithm:

Step 1: Identify a node with no incoming edges.

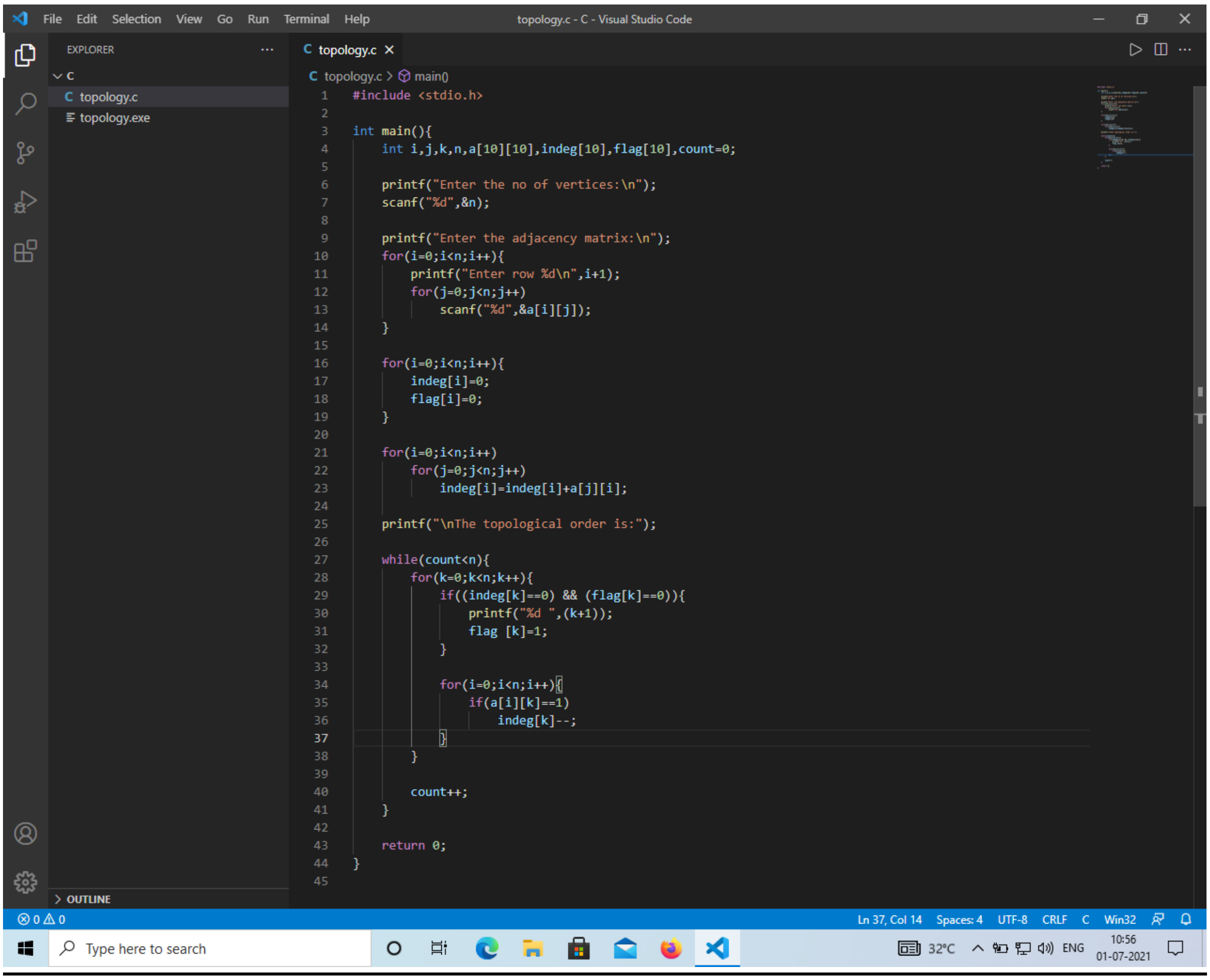
Step 2: Add that node to the ordering

Step 3: Remove it from the graph.

Step 4: Repeat .

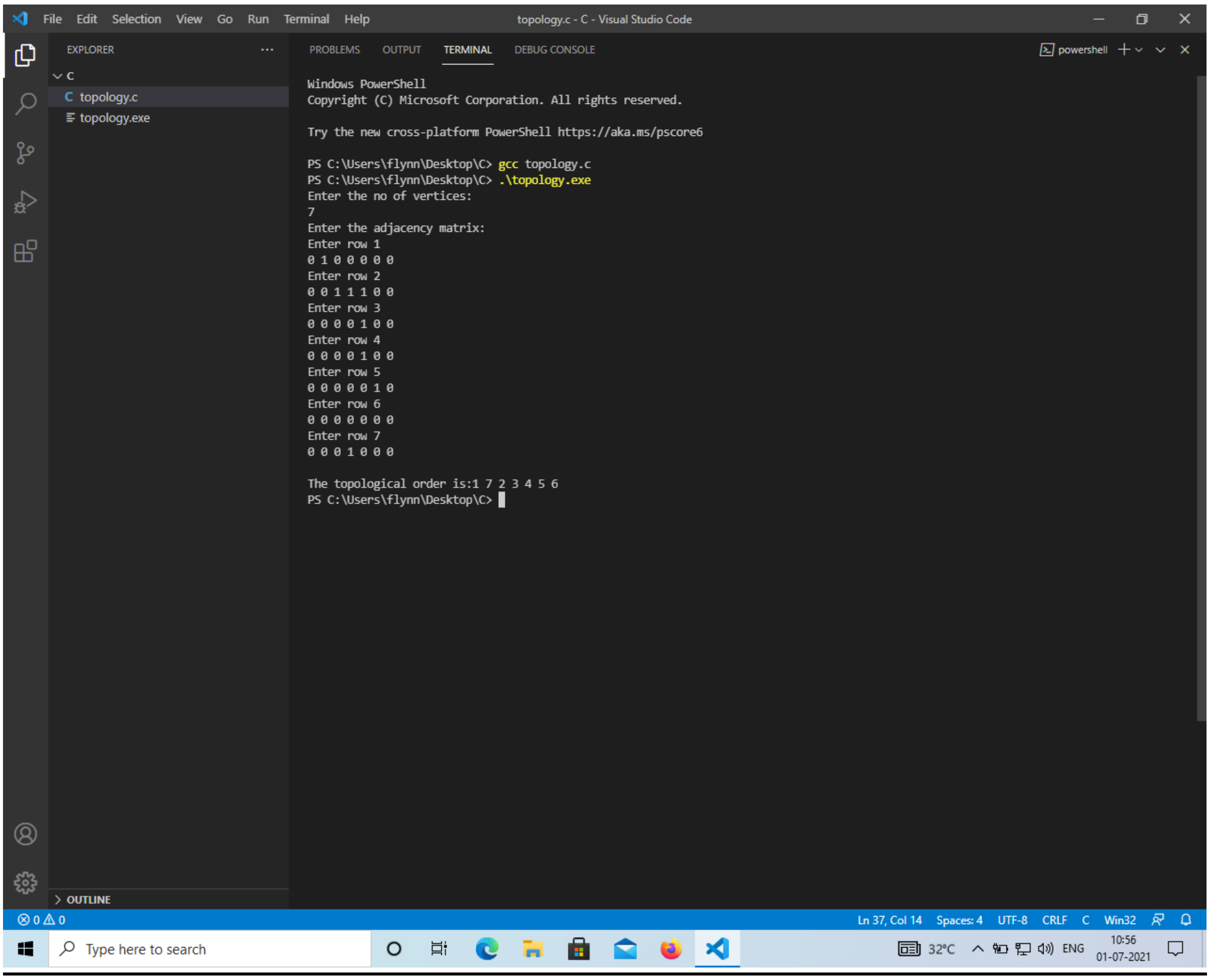
PROGRAM CODE

```
#include <stdio.h>


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


          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 %d\n",i+1);
                    for(j=0;j<n;j++)
                              scanf("%d",&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("%d ",(k+1));
                flag [k]=1;
            }

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

        count++;
    }

    return 0;
}
```



```c
#include <stdio.h>

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

    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 %d\n",i+1);
        for(j=0;j<n;j++)
            scanf("%d",&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("%d ",(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\flynn\Desktop\C> gcc topology.c
PS C:\Users\flynn\Desktop\C> .\topology.exe
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 1 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\flynn\Desktop\C>
```

2) Write a program for creating Doubly LL and perform the following operations

A) Insert an element at a particular position

B) Search an element.

c) Delete an element at the end of the list.

Algorithm

A) Insert an element at a particular position

```
Step-1  IF PTR = NULL
            Go to Step 15
Step-2 : SET NEW_NODE = PTR
Step-3 : SET PTR = PTR -> NEXT
Step-4 : SET NEW_NODE -> DATA = VAL
Step 5 : SET TEMP = START
Step 6 : SET I = 0
Step -7 : REPEAT 8 to 10 untill I
Step-8 : SET TEMP = TEMP -> NEXT
```

```
step-9 : IF TEMP = NULL
step-10:     WRITE "LESS THAN DESIRED No. OF ELEMENTS"
             Goto step 15
step-11 : SET NEW_NODE -> NEXT = TEMP -> NEXT
step-12 : SET NEW_NODE -> PREV = TEMP.
step-13 : SET TEMP -> NEXT = NEW NODE.
step-14 : SET TEMP -> NEXT -> PREV = NEW NODE
step-15 : EXIT
```

B) Search an element

```
Step-1   Input the item which we want to search
         as data
Step-2 : P = Start.
Step-3 : Repeat while P<> NULL
             IF (P => info = data)
                 Print the location of node.
```

PROGRAM CODE

```
#include<stdio.h>

#include<stdlib.h>

struct node

{

   struct node *prev;

   struct node *next;

   int data;

};

struct node *head;

void insert_at_beginning();

void insert_at_specified();

void deletion_at_last();

void display();

void search();

void main ()

{

int choice =0;

   while(choice != 9)

   {

      printf("\n");

      printf("\nChoose one option from the following list");

      printf("\n1.Insert in beginning  2.Insert at a particular position 3.Delete from last
4.Search 5.Show 9.Exit");

      printf("\nEnter your choice? = ");

      scanf("%d",&choice);

      switch(choice)

      {

         case 1:
```

```
        insert_at_beginning();
        break;
        case 2:
        insert_at_specified();
        break;
        case 3:
        deletion_at_last();
        break;
        case 4:
        search();
        break;
        case 5:
        display();
        break;
        case 6:
        exit(0);
        break;
        default:
        printf("Please enter valid choice in the menu");
      }
   }
}
void insert_at_beginning()
{
  struct node *ptr;
  int item;
  ptr = (struct node *)malloc(sizeof(struct node));
  if(ptr == NULL)
  {
     printf("\nOVERFLOW");
  }
  else
  {
   printf("Enter Item value to insert at beginnning = ");
   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 successfully");
}

}

void insert_at_specified()
{
```

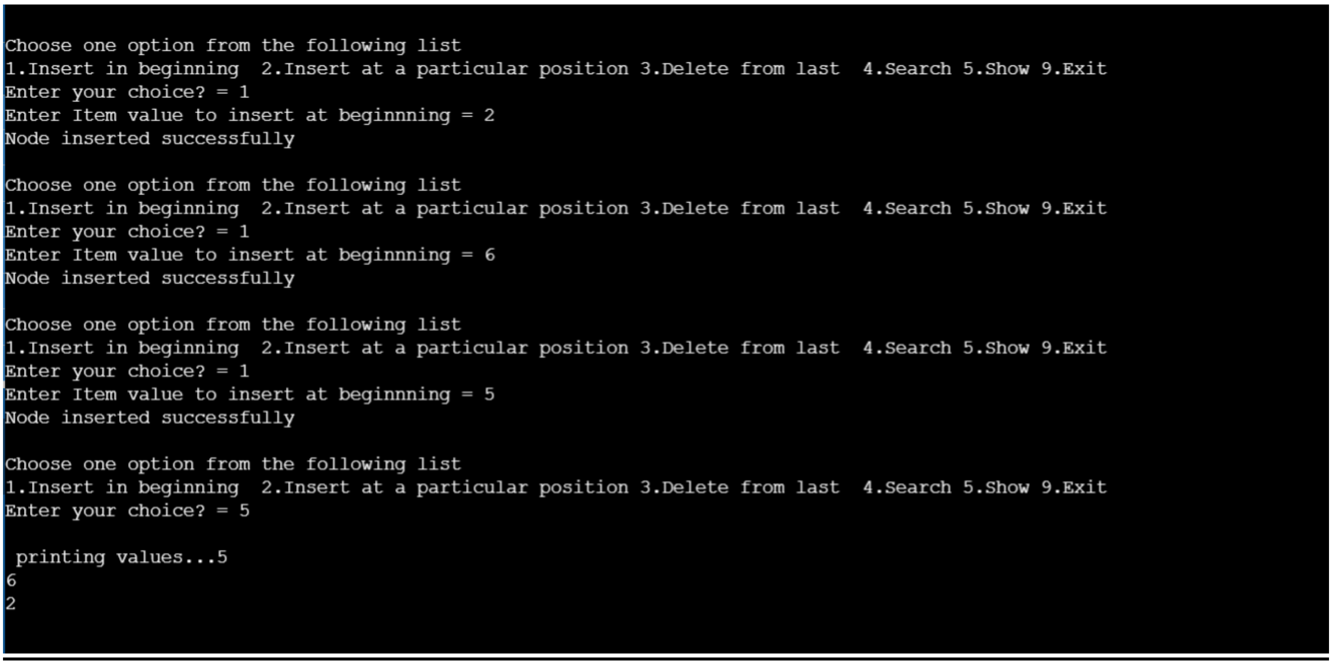
```
struct node *ptr,*temp;
int item,loc,i;
ptr = (struct node *)malloc(sizeof(struct node));
if(ptr == NULL)
{
   printf("\n OVERFLOW");
}
else
{
   temp=head;
   printf("Enter the location  = ");
   scanf("%d",&loc);
   for(i=0;i<loc;i++)
   {
      temp = temp->next;
      if(temp == NULL)
      {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

```
  }
}

void deletion_at_last()
{
   struct node *ptr;
   if(head == NULL)
   {
      printf("\n UNDERFLOW");
   }
   else if(head->next == NULL)
   {
      head = NULL;
      free(head);
      printf("\nnode deleted successfully");
   }
   else
   {
      ptr = head;
      while(ptr->next != NULL)
      {
         ptr = ptr -> next;
      }
      ptr -> prev -> next = NULL;
      free(ptr);
      printf("\nnode deleted successfully");
   }
}
```

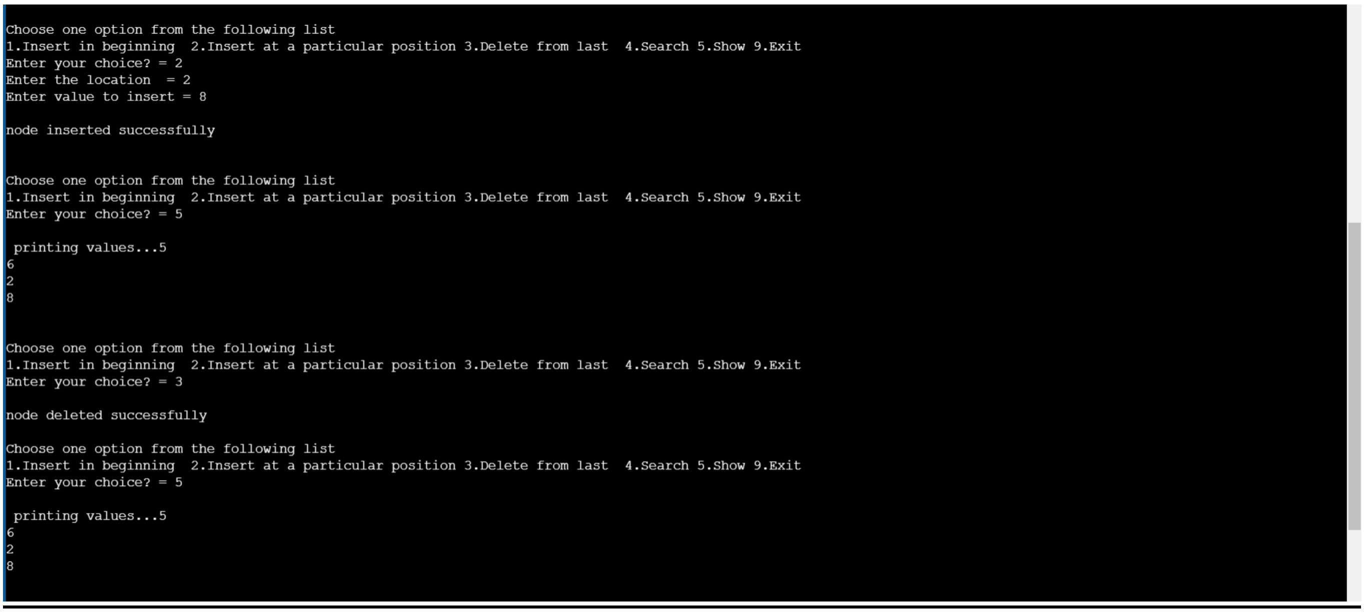
```
void display()
{
   struct node *ptr;
   printf("\n printing values...");
   ptr = head;
   while(ptr != NULL)
   {
      printf("%d\n",ptr->data);
      ptr=ptr->next;
   }
}
void search()
{
   struct node *ptr;
   int item,i=0,flag;
   ptr = head;
   if(ptr == NULL)
   {
      printf("\nEmpty List");
   }
   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");
        }
    }

}
```

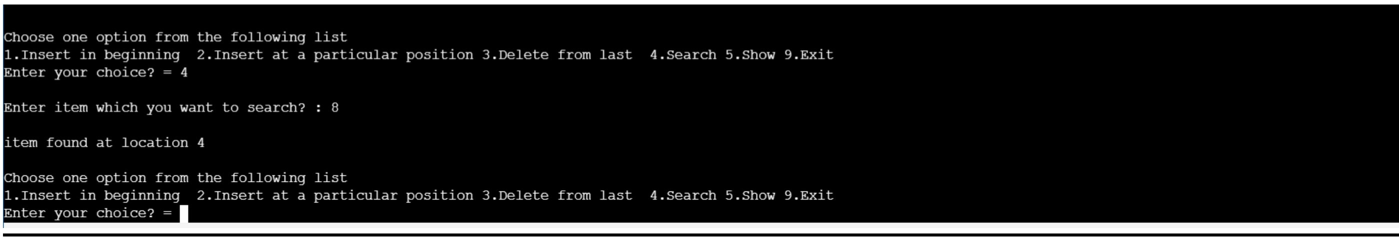
OUTPUT



```
Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 1
Enter Item value to insert at beginnning = 2
Node inserted successfully

Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 1
Enter Item value to insert at beginnning = 6
Node inserted successfully

Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 1
Enter Item value to insert at beginnning = 5
Node inserted successfully

Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 5

 printing values...5
6
2
```



```
Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 2
Enter the location  = 2
Enter value to insert = 8

node inserted successfully


Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 5

 printing values...5
6
2
8


Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 3

node deleted successfully

Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 5

 printing values...5
6
2
8
```



```
Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? = 4

Enter item which you want to search? : 8

item found at location 4

Choose one option from the following list
1.Insert in beginning  2.Insert at a particular position 3.Delete from last  4.Search 5.Show 9.Exit
Enter your choice? =
```