VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

DATA STRUCTURE LAB RECORD (19CS3PCDST)

Submitted by

RAVI SAJJANAR (1BM19CS127)

Under the Guidance of

Prof. SHEETAL VA
Assistant Professor, BMSCE

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2020 to Jan-2021

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the LAB RECORD carried out by **RAVI SAJJANAR (1BM19CS127)** who is the bonafide students of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visveswaraiah Technological University, Belgaum during the year 2020-2021. The lab report has been approved as it satisfies the academic requirements in respect of **DATA STRUCTURE LAB RECORD (19CS3PCDST)** work prescribed for the said degree.

Signature of the Guide	Signature of the HOD
Prof. Prof. Sheelal VA	Dr. Umadevi V
Assistant Professor	Associate Prof.& Head, Dept. of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

External Viva

Name of the Examiner	Signature with date
1.______________________________	_________________________
2. ______________________________	__________________________

INDEX

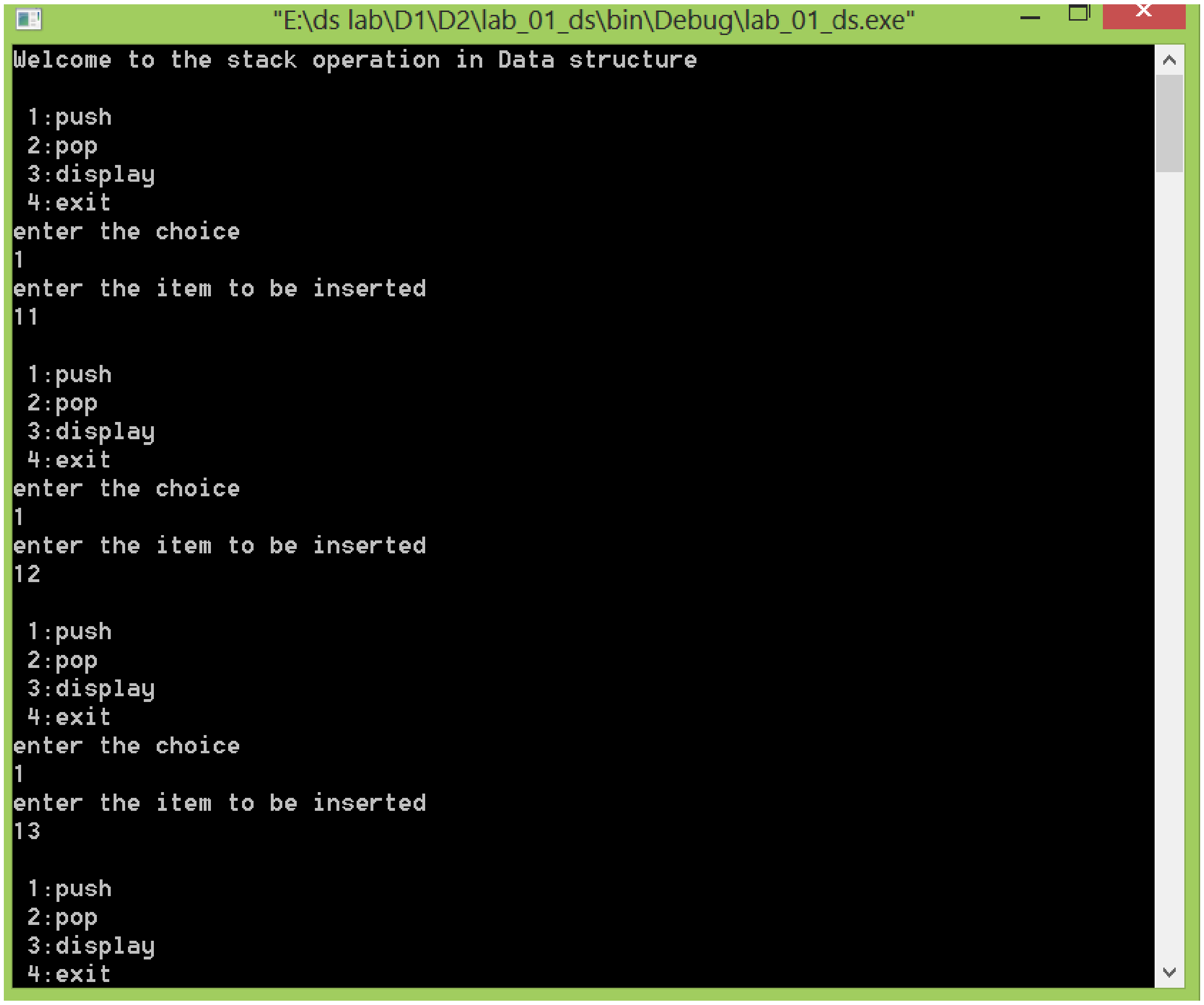
PROGRAM 1:

Write a program to simulate the working of stack using an array with the following:
a) Push
b) Pop
c) Display
The program should print appropriate messages for stack overflow, stack underflow

```
#include <stdio.h>
#include <stdlib.h>
# define STACK_SIZE 5
int top = -1;
int s[10];
int item;

void push()
{
   if (top==STACK_SIZE-1)
   {
      printf("Stack over_flow\n");
      return;
   }
   top=top+1;
   s[top]=item;
}

int pop()
{
   if (top==-1)
      return -1;
   return s[top--];
}

void display()
{
    int i;
    if(top==-1)
    {
        printf("Stack is empty\n");
        return;
    }
    printf("contents of the stack\n");
    for (i=top;i>=0;i--)
    {
        printf("%d\n",s[i]);
    }
}

void main()
{
   int item_deleted;
   int choice;

   for(;;)
   {
       printf("\n 1:push \n 2:pop \n 3:display \n 4:exit\n" );
       printf("enter the choice\n");
       scanf("%d",&choice);

       switch(choice)
       {
           case 1: printf("enter the item to be inserted\n");
                   scanf("%d",&item);
                   push();
                   break;
           case 2: item_deleted=pop();
                   if(item_deleted==-1)
                    printf("Stack is empty\n");
                   else
                    printf("item_deleted is %d\n",item_deleted);
                   break;
           case 3: display();
```

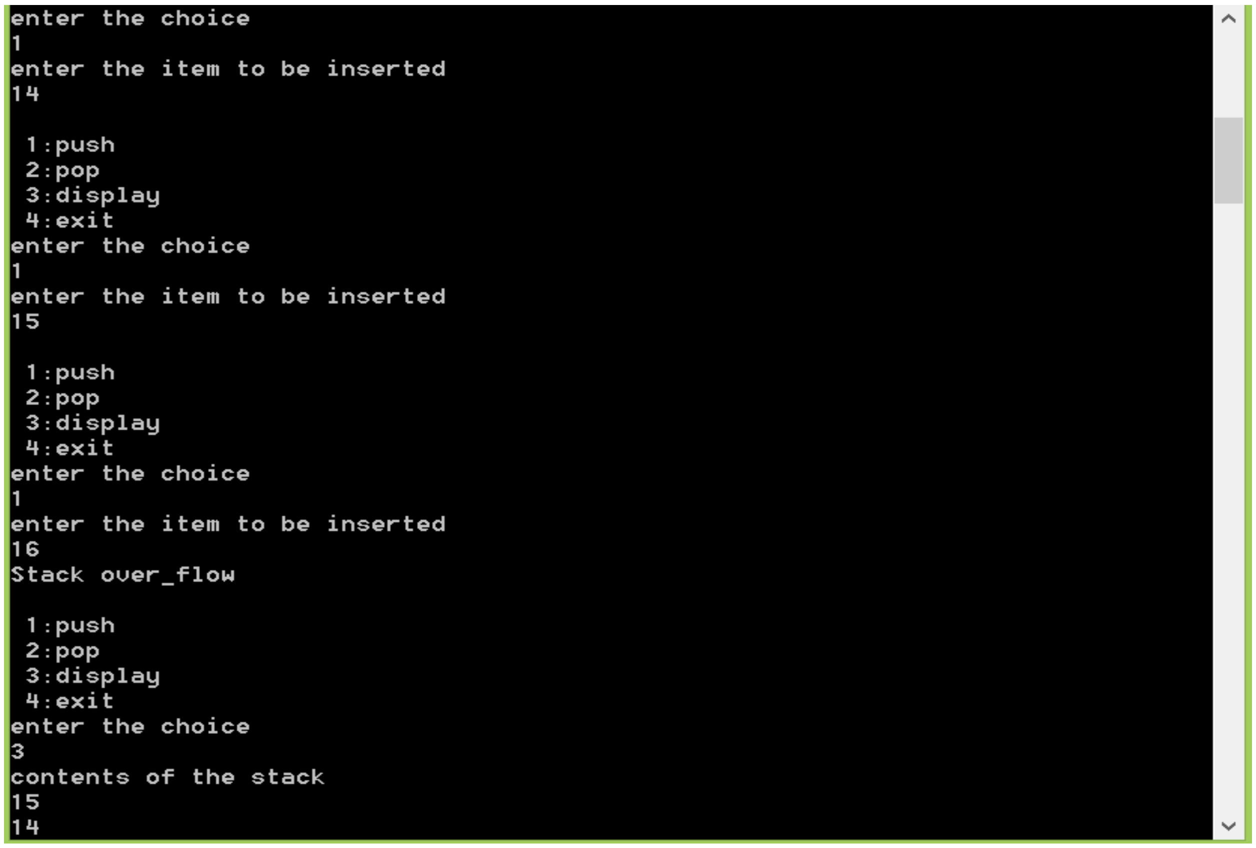
```
                break;
            default : exit(0);

        }
    }
}
```

OUTPUT:

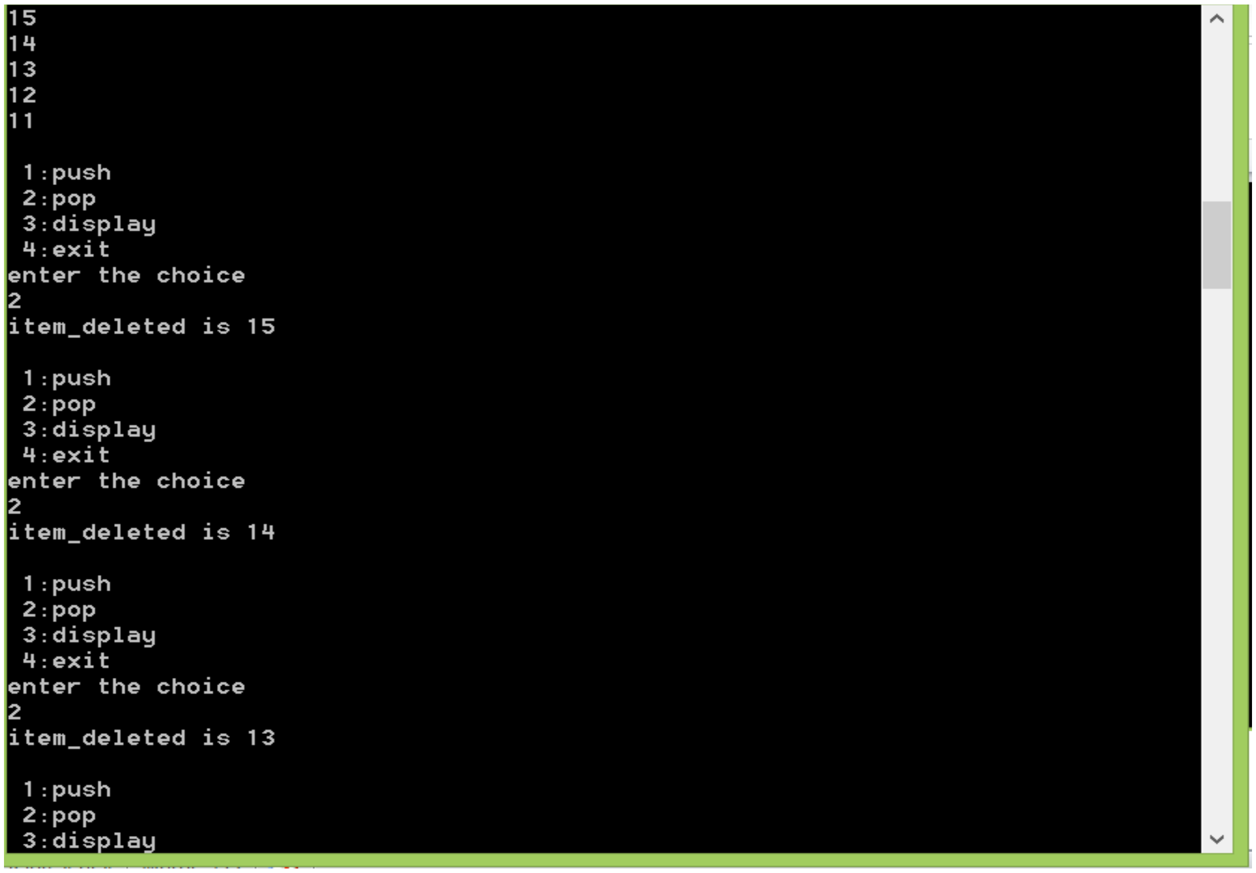


"E:\ds lab\D1\D2\lab_01_ds\bin\Debug\lab_01_ds.exe"

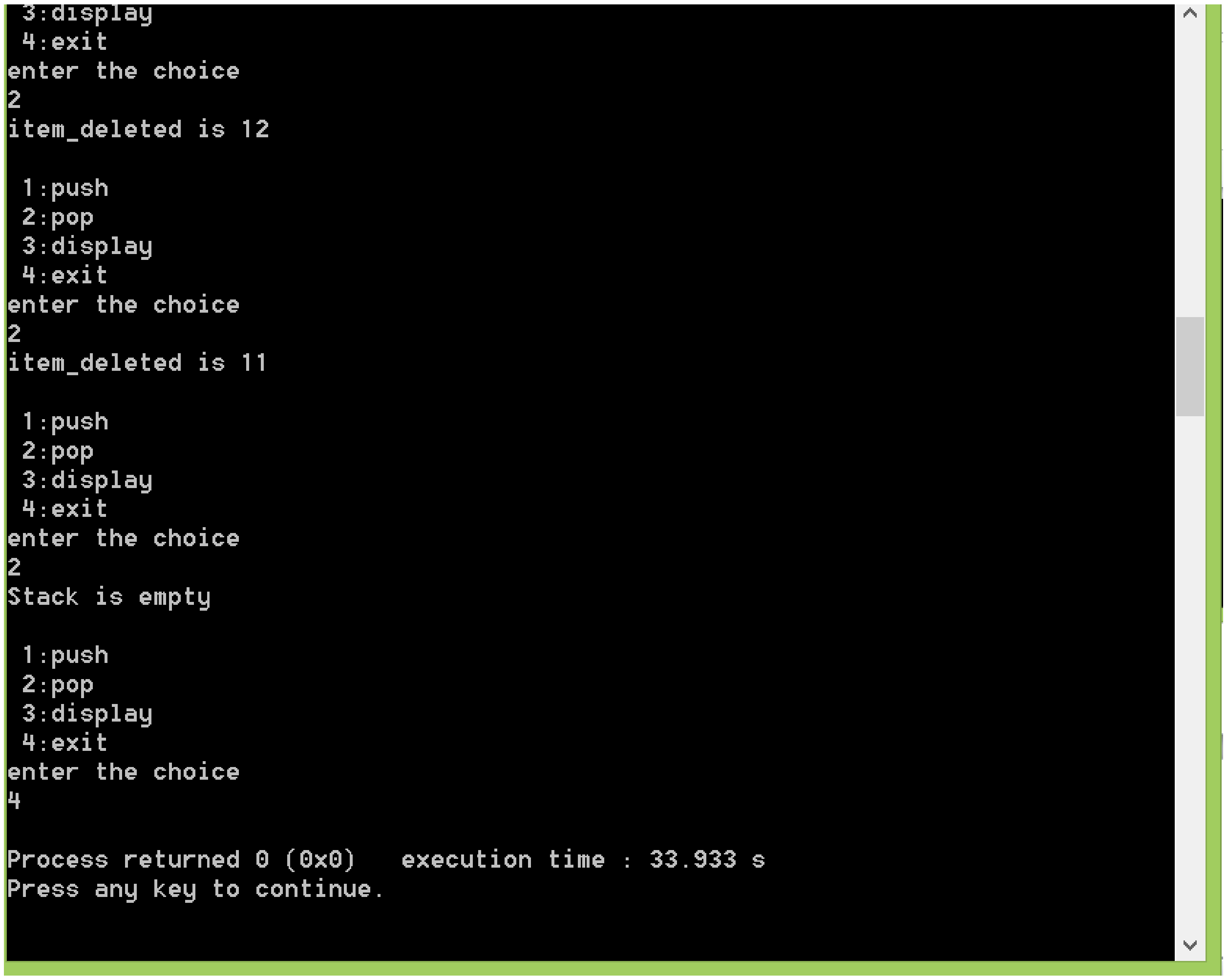
```
Welcome to the stack operation in Data structure

 1:push
 2:pop
 3:display
 4:exit
enter the choice
1
enter the item to be inserted
11

 1:push
 2:pop
 3:display
 4:exit
enter the choice
1
enter the item to be inserted
12

 1:push
 2:pop
 3:display
 4:exit
enter the choice
1
enter the item to be inserted
13

 1:push
 2:pop
 3:display
 4:exit
```



```
enter the choice
1
enter the item to be inserted
14

 1:push
 2:pop
 3:display
 4:exit
enter the choice
1
enter the item to be inserted
15

 1:push
 2:pop
 3:display
 4:exit
enter the choice
1
enter the item to be inserted
16
Stack over_flow

 1:push
 2:pop
 3:display
 4:exit
enter the choice
3
contents of the stack
15
14
```



```
15
14
13
12
11

 1:push
 2:pop
 3:display
 4:exit
enter the choice
2
item_deleted is 15

 1:push
 2:pop
 3:display
 4:exit
enter the choice
2
item_deleted is 14

 1:push
 2:pop
 3:display
 4:exit
enter the choice
2
item_deleted is 13

 1:push
 2:pop
 3:display
```



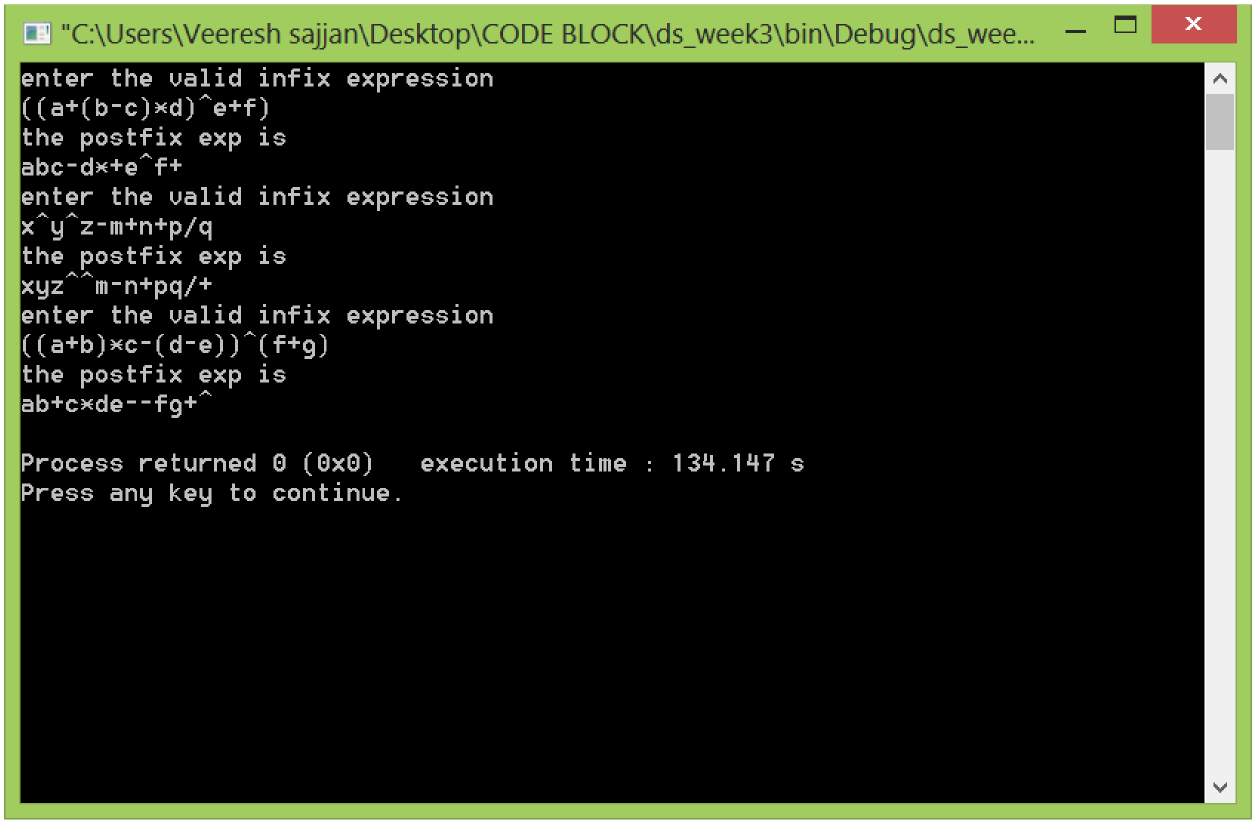
```
 3:display
 4:exit
enter the choice
2
item_deleted is 12

 1:push
 2:pop
 3:display
 4:exit
enter the choice
2
item_deleted is 11

 1:push
 2:pop
 3:display
 4:exit
enter the choice
2
Stack is empty

 1:push
 2:pop
 3:display
 4:exit
enter the choice
4

Process returned 0 (0x0)   execution time : 33.933 s
Press any key to continue.
```

PROGRAM 2:

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide)

```
#include <stdio.h>
#include <string.h>
#include<process.h>

int F(char symbol)
{
   switch(symbol)
   {
   case '+':
   case '-':return 2;
   case '*':
   case '/':return 4;
   case '^':
   case '$':return 5;
   case '(':return 0;
   case '#':return -1;
   default :return 8;
   }
}

int G(char symbol)
{
   switch(symbol)
   {
   case '+':
   case '-':return 1;
   case '*':
   case '/':return 3;
   case '^':
   case '$':return 6;
   case '(':return 9;
   case ')':return 0;
   default :return 7;
   }
}

void infix_postfix(char infix[],char postfix[])
{
   int top,i,j;
   char s[30],symbol;
   top = -1;
   s[++top]='#';
   j=0;

   for (i=0;i<strlen(infix);i++)
   {
       symbol=infix[i];
       while(F(s[top])>G(symbol))
       {
           postfix[j]=s[top--];
           j++;
       }
       if (F(s[top])!= G(symbol))
           s[++top]=symbol;
       else
           top--;
   }

   while (s[top] != '#')
   {
       postfix [j++]=s[top--];
   }
   postfix[j]='\0';
}

void main()
{
   char infix[20];
   char postfix[200];
   int i;

   printf("enter the valid infix expression\n");
   scanf("%s",infix);

   infix_postfix(infix,postfix);
   printf("the postfix exp is \n");
   printf("%s\n",postfix);

}
```

OUTPUT:



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ds_week3\bin\Debug\ds_wee...

```
enter the valid infix expression
((a+(b-c)*d)^e+f)
the postfix exp is
abc-d*+e^f+
enter the valid infix expression
x^y^z-m+n+p/q
the postfix exp is
xyz^^m-n+pq/+
enter the valid infix expression
((a+b)*c-(d-e))^(f+g)
the postfix exp is
ab+c*de--fg+^

Process returned 0 (0x0)   execution time : 134.147 s
Press any key to continue.
```

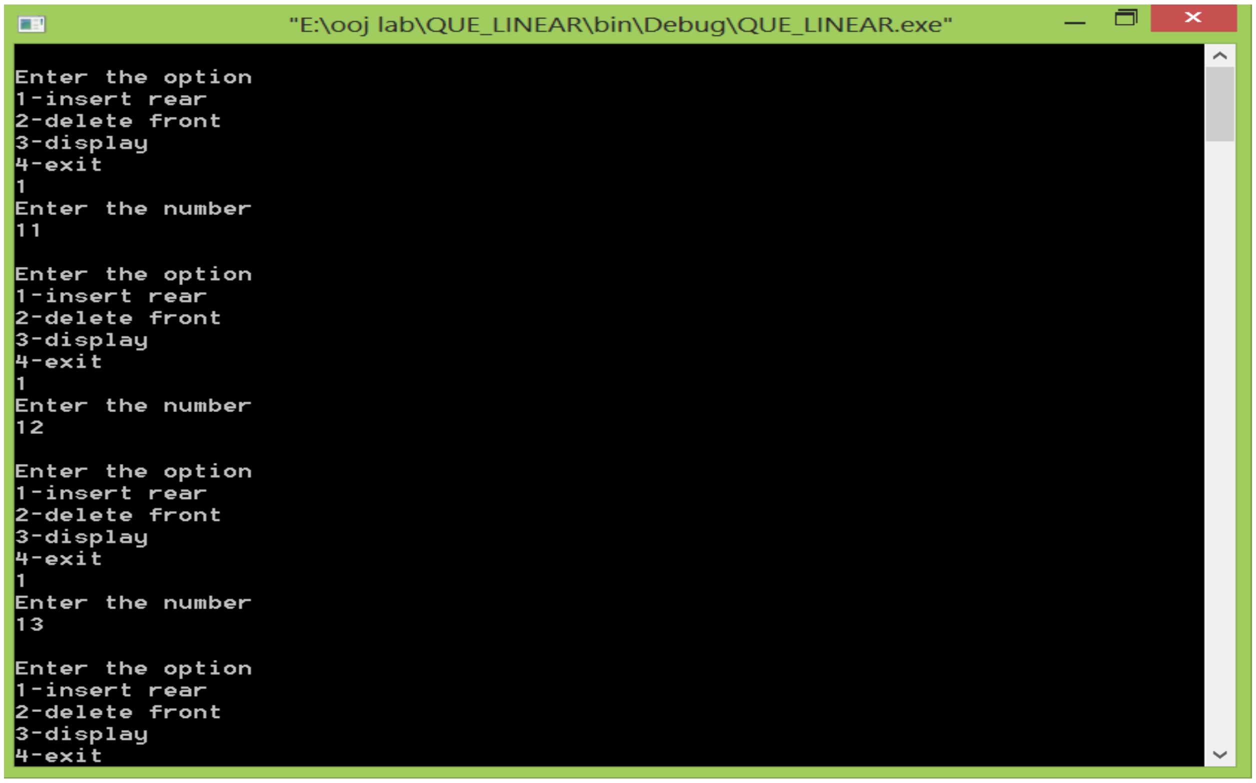
PROGRAM 3:

WAP to simulate the working of a queue of integers using an array. Provide the following operations a) Insert b) Delete c) Display. The program should print appropriate messages for queue empty and queue overflow conditions

```
#include<stdio.h>
#define size 5

int f=0,r=-1,value;
int q[size];

void insertRear(){
	if(r==size-1){
		printf("Queue Overflow\n");
		return;
	}
	q[++r]=value;
}

void deleteFront(){
	if(f>r){
		printf("Queue Underflow\n");
		return;
	}
	printf("deleted=%d\n",q[f++]);
	if(f>r){
		f=0;
		r=-1;
	}
}

void display(){
	if(f>r){
		printf("null\n");
		return;
	}
```

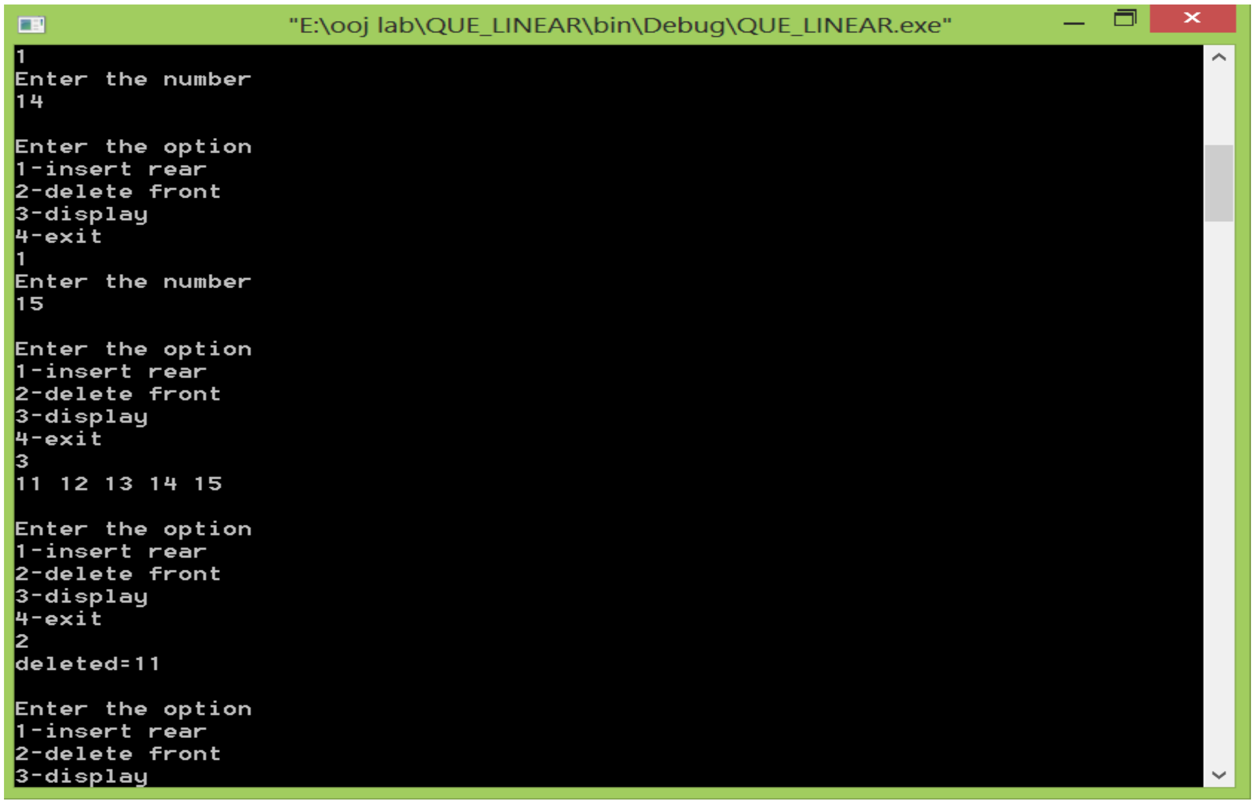
```
        int i;
        for(i=f;i<=r;i++){
                printf("%d ",q[i]);
        }
        printf("\n");
}

int main(){
        int ch;
        while(1){
                printf("\nEnter the option\n1-insert rear\n2-delete front\n3-display\n4-
exit\n");
                scanf("%d",&ch);
                switch(ch){
                        case 1:
                                        printf("Enter the number\n");
                                        scanf("%d",&value);
                                        insertRear(value);
                                break;
                        case 2:
                                deleteFront();
                                break;
                        case 3:
                                display();
                                break;
                        default:
                                return 0;
                }
        }
}
```

OUTPUT:



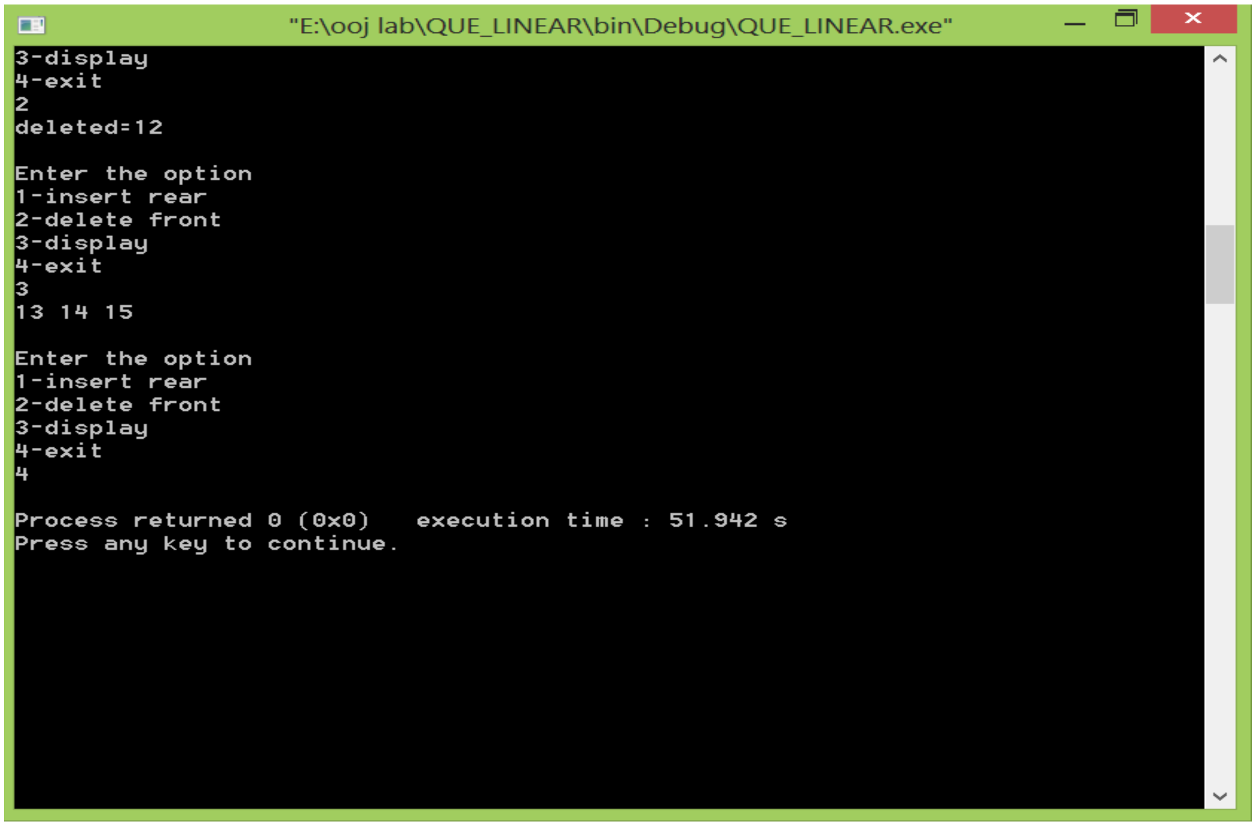
"E:\ooj lab\QUE_LINEAR\bin\Debug\QUE_LINEAR.exe"

```
Enter the option
1-insert rear
2-delete front
3-display
4-exit
1
Enter the number
11

Enter the option
1-insert rear
2-delete front
3-display
4-exit
1
Enter the number
12

Enter the option
1-insert rear
2-delete front
3-display
4-exit
1
Enter the number
13

Enter the option
1-insert rear
2-delete front
3-display
4-exit
```



"E:\ooj lab\QUE_LINEAR\bin\Debug\QUE_LINEAR.exe"

```
1
Enter the number
14

Enter the option
1-insert rear
2-delete front
3-display
4-exit
1
Enter the number
15

Enter the option
1-insert rear
2-delete front
3-display
4-exit
3
11 12 13 14 15

Enter the option
1-insert rear
2-delete front
3-display
4-exit
2
deleted=11

Enter the option
1-insert rear
2-delete front
3-display
```


"E:\ooj lab\QUE_LINEAR\bin\Debug\QUE_LINEAR.exe"

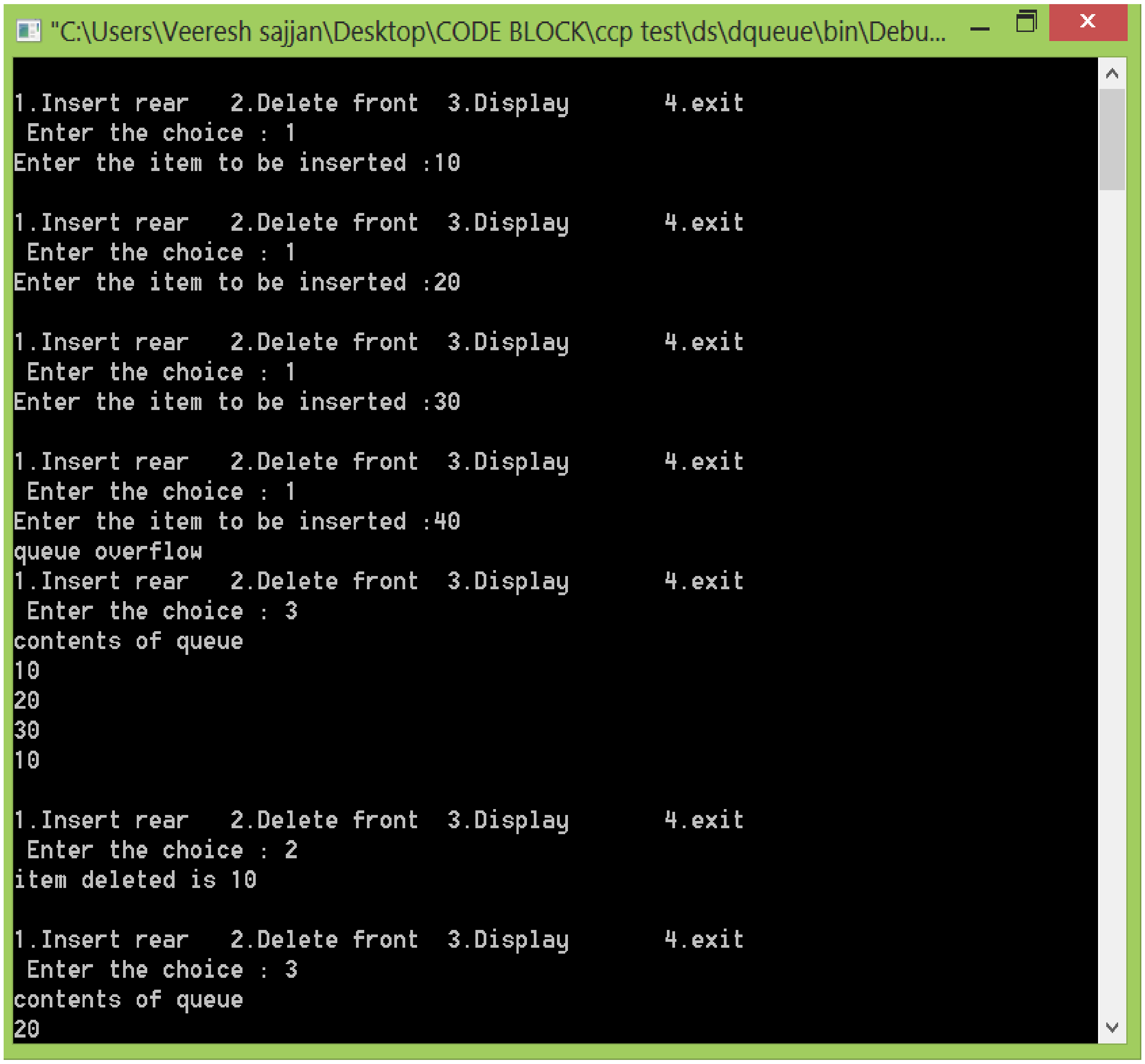
```
3-display
4-exit
2
deleted=12

Enter the option
1-insert rear
2-delete front
3-display
4-exit
3
13 14 15

Enter the option
1-insert rear
2-delete front
3-display
4-exit
4

Process returned 0 (0x0)   execution time : 51.942 s
Press any key to continue.
```

PROGRAM 4:

WAP to simulate the working of a circular queue of integers using an array. Provide the following operations. a) Insert b) Delete c) Display. The program should print appropriate messages for queue empty and queue overflow conditions

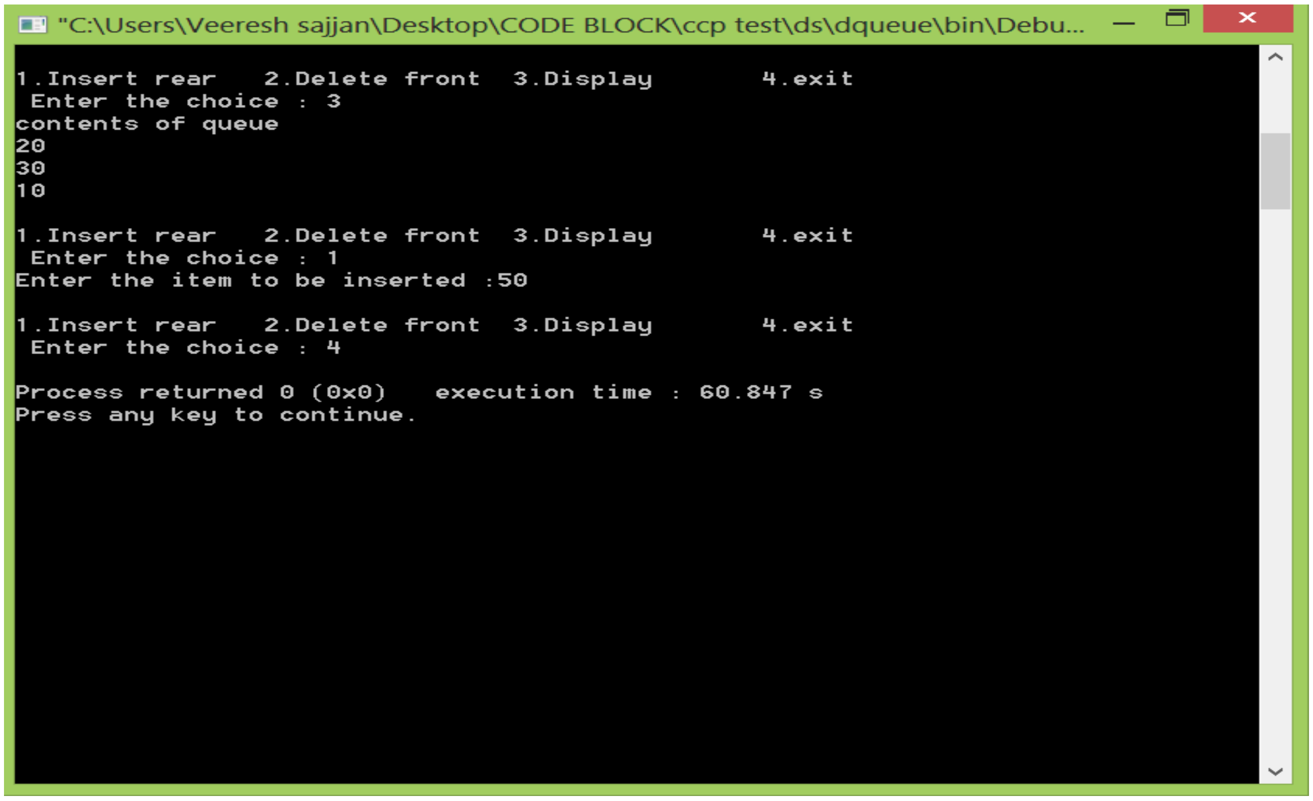
```
#include<stdio.h>
#include<stdlib.h>
#include<process.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size],count=0;
void insertrear()
{
    if(count==que_size)
    {
            printf("queue overflow");
            return;
    }
    rear=(rear+1)%que_size;
    q[rear]=item;
    count++;
}
int deletefront()
{
    if(count==0) return -1;
    item = q[front];
    front=(front+1)%que_size;
    count=count-1;
    return item;
}
void displayq()
{
    int i,f;
    if(count==0)
    {
            printf("queue is empty");
            return;
    }
    f=front;
    printf("contents of queue \n");
    for(i=0;i<=count;i++)
    {
            printf("%d\n",q[f]);
            f=(f+1)%que_size;
    }
}
void main()
{
    int choice;
    for(;;)
    {
            printf("\n1.Insert rear \t2.Delete front \t3.Display \t4.exit \n ");
            printf("Enter the choice : ");
            scanf("%d",&choice);
            switch(choice)
            {
                    case 1:printf("Enter the item to be inserted :");
                         scanf("%d",&item);
                         insertrear();
                         break;
                    case 2:item=deletefront();
                              if(item==-1)
                              printf("queue is empty\n");
                              else
                              printf("item deleted is %d \n",item);
                              break;
                 case 3:displayq();
                              break;
                    default:exit(0);
                }
        }

    }
```

OUTPUT:



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp test\ds\dqueue\bin\Debu...

```
1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 1
Enter the item to be inserted :10

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 1
Enter the item to be inserted :20

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 1
Enter the item to be inserted :30

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 1
Enter the item to be inserted :40
queue overflow
1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 3
contents of queue
10
20
30
10

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 2
item deleted is 10

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 3
contents of queue
20
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp test\ds\dqueue\bin\Debu...

```
1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 3
contents of queue
20
30
10

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 1
Enter the item to be inserted :50

1.Insert rear   2.Delete front  3.Display       4.exit
 Enter the choice : 4

Process returned 0 (0x0)   execution time : 60.847 s
Press any key to continue.
```

PROGRAM 5 & 6:

WAP to Implement Singly Linked List with following operations
a) Create a linked list.
b) Insertion of a node at first position, at any position and at end of list.
c) Deletion of first element, specified element and last element in the list.
d) Display the contents of the linked list.

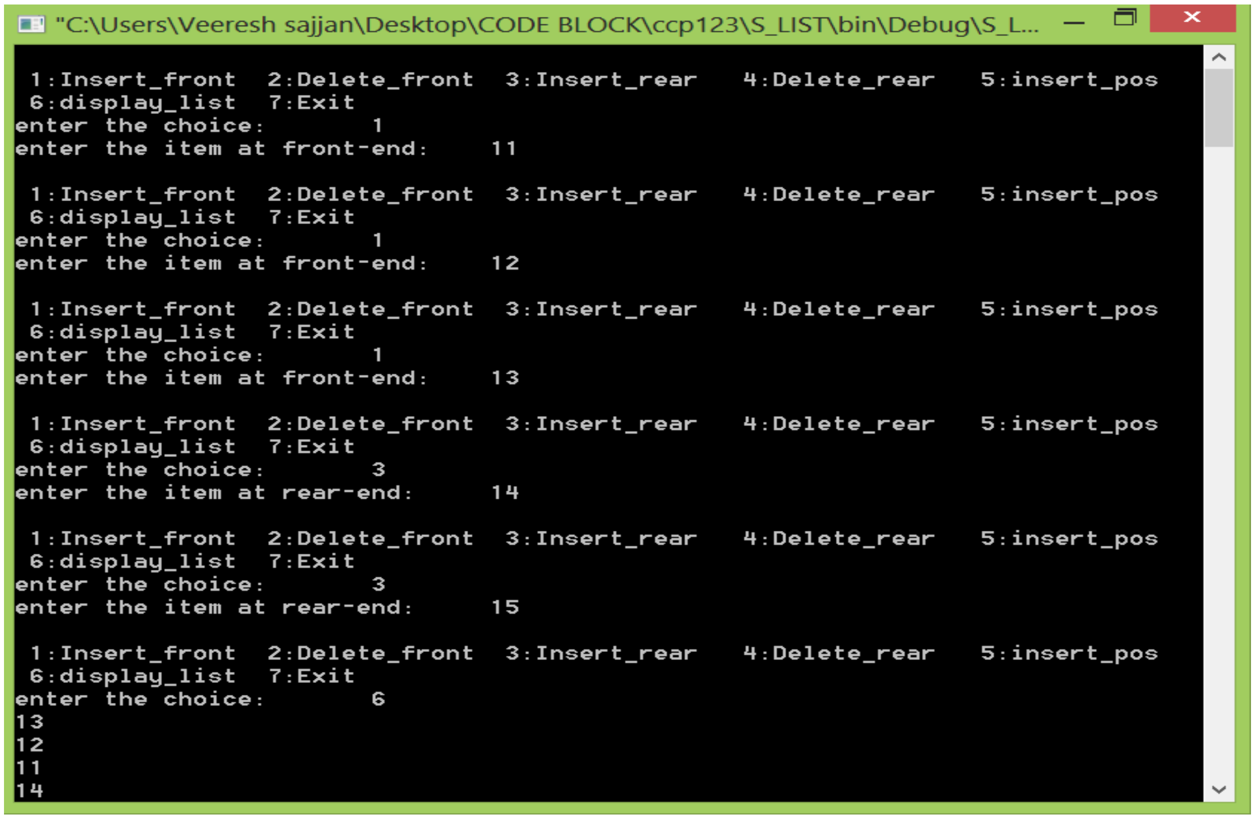
```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<process.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
```

```
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
NODE insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
 return temp;
cur=first;
while(cur->link!=NULL)
 cur=cur->link;
cur->link=temp;
return first;
}
NODE delete_rear(NODE first)
{
NODE cur,prev;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
if(first->link==NULL)
{
printf("item deleted is %d\n",first->info);
free(first);
return NULL;
}
prev=NULL;
cur=first;
while(cur->link!=NULL)
{
prev=cur;
cur=cur->link;
}
printf("iten deleted at rear-end is %d",cur->info);
free(cur);
prev->link=NULL;
return first;
}
NODE insert_pos(int item,int pos,NODE first)
{
NODE temp;
NODE prev,cur;
int count;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL && pos==1)
return temp;
if(first==NULL)
{
 printf("invalid pos\n");
 return first;
}
if(pos==1)
{
temp->link=first;
return temp;
}
```

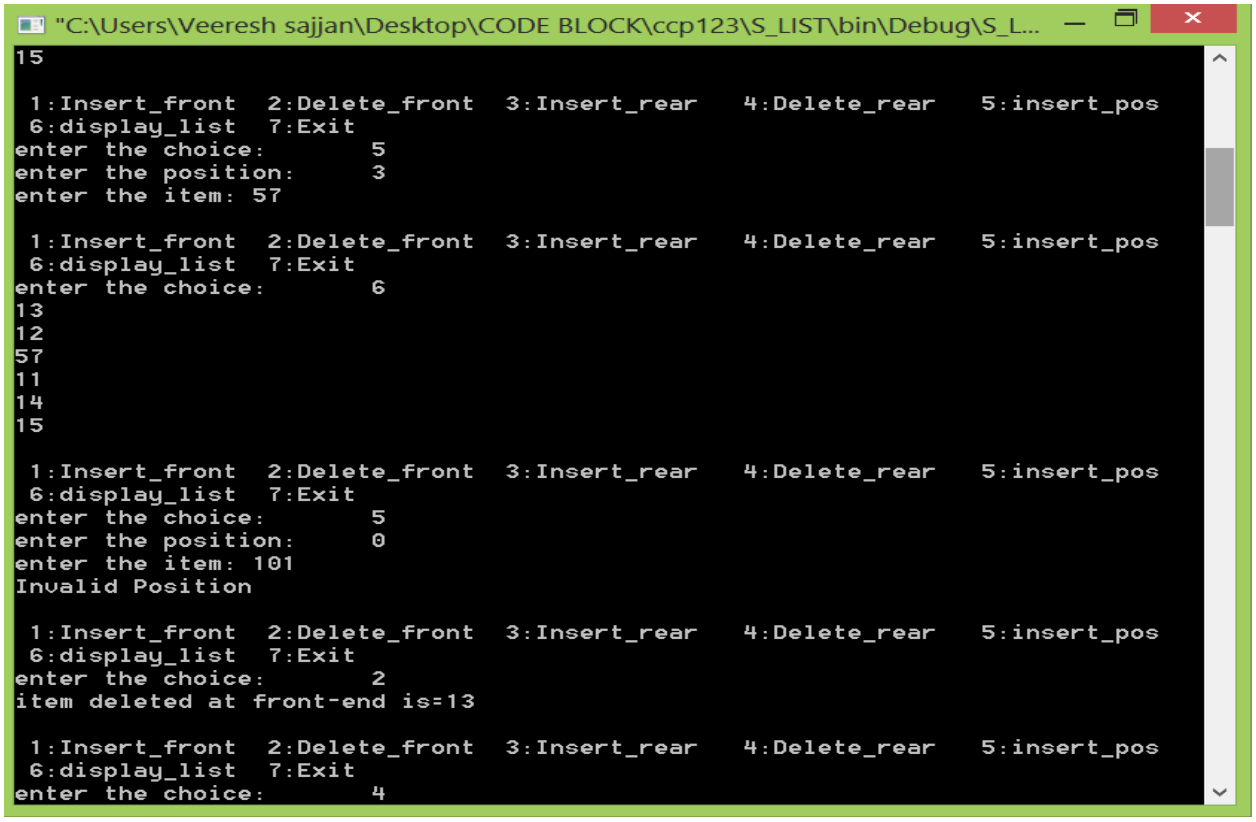
```
count=1;
prev=NULL;
cur=first;
while(cur!=NULL && count!=pos)
{
 prev=cur;
 cur=cur->link;
 count++;
}
if(count==pos)
{
prev->link=temp;
temp->link=cur;
return first;
}
printf("IP\n");
return first;
}
void display(NODE first)
{
 NODE temp;
 if(first==NULL)
 printf("list empty cannot display items\n");
 for(temp=first;temp!=NULL;temp=temp->link)
  {
  printf("%d\n",temp->info);
  }
}
void main()
{
int item,choice,pos;
NODE first=NULL;
clrscr();
for(;;)
{
printf("\n 1:Insert_front\n 2:Delete_front\n 3:Insert_rear\n 4:Delete_rear\n
5:insert_pos\n 6:display_list\n7:Exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
```

```
 {
  case 1:printf("enter the item at front-end\n");
          scanf("%d",&item);
          first=insert_front(first,item);
          break;
  case 2:first=delete_front(first);
          break;
  case 3:printf("enter the item at rear-end\n");
          scanf("%d",&item);
          first=insert_rear(first,item);
          break;
  case 4:first=delete_rear(first);
          break;
  case 5:printf("enter the position\n");
                  scanf("%d",&pos);
                  first=insert_pos(item,pos,first);
                  break;
  case 6:display(first);
          break;
 default:exit(0);
          break;
 }
}
getch();
}
```

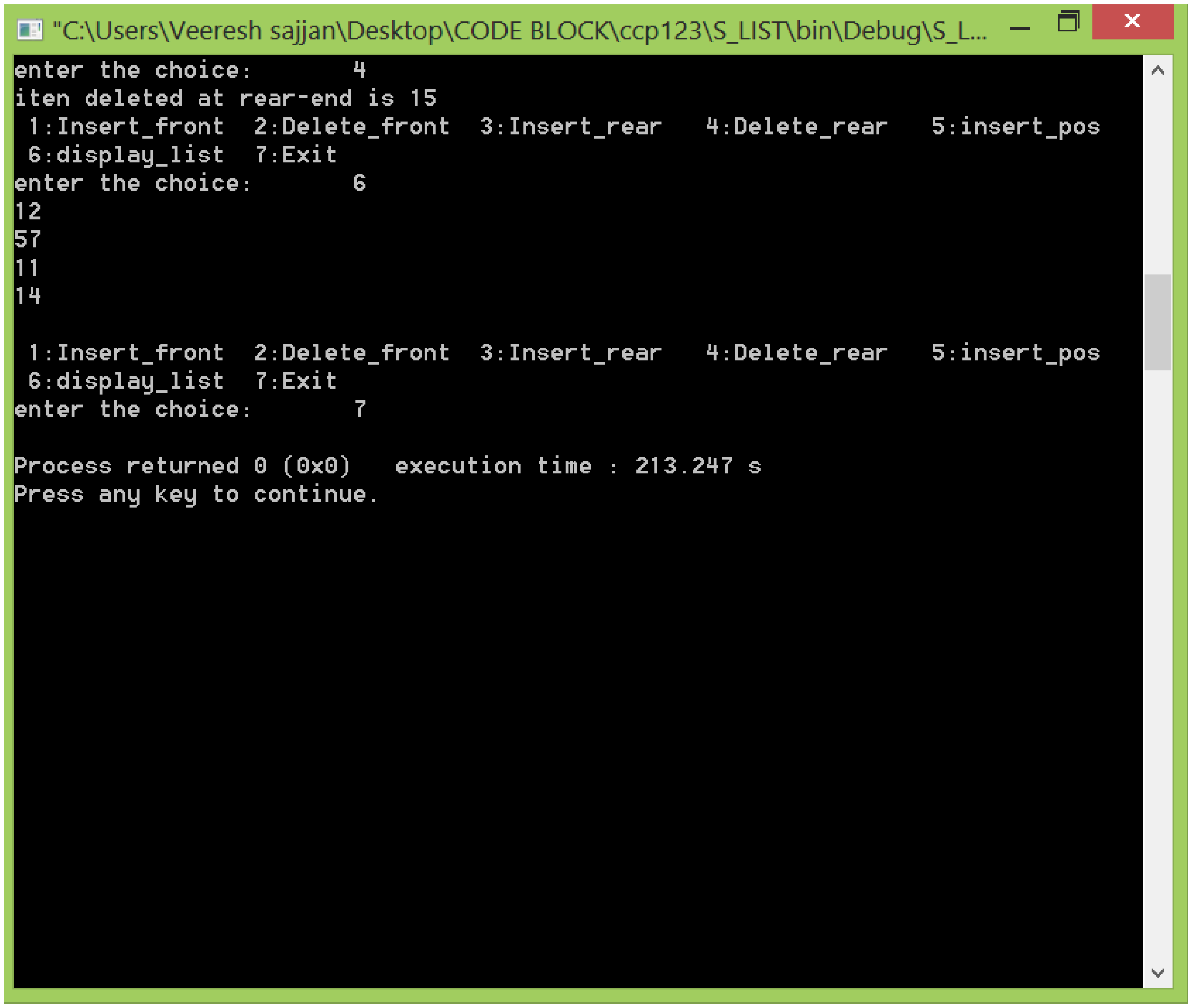
OUTPUT:



```
"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\S_LIST\bin\Debug\S_L...

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       1
enter the item at front-end:    11

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       1
enter the item at front-end:    12

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       1
enter the item at front-end:    13

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       3
enter the item at rear-end:     14

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       3
enter the item at rear-end:     15

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       6
13
12
11
14
```



```
"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\S_LIST\bin\Debug\S_L...
15

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       5
enter the position:     3
enter the item: 57

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       6
13
12
57
11
14
15

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       5
enter the position:     0
enter the item: 101
Invalid Position

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       2
item deleted at front-end is=13

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       4
```


"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\S_LIST\bin\Debug\S_L...

```
enter the choice:       4
iten deleted at rear-end is 15
 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       6
12
57
11
14

 1:Insert_front  2:Delete_front  3:Insert_rear   4:Delete_rear   5:insert_pos
 6:display_list  7:Exit
enter the choice:       7

Process returned 0 (0x0)   execution time : 213.247 s
Press any key to continue.
```

PROGRAM 7

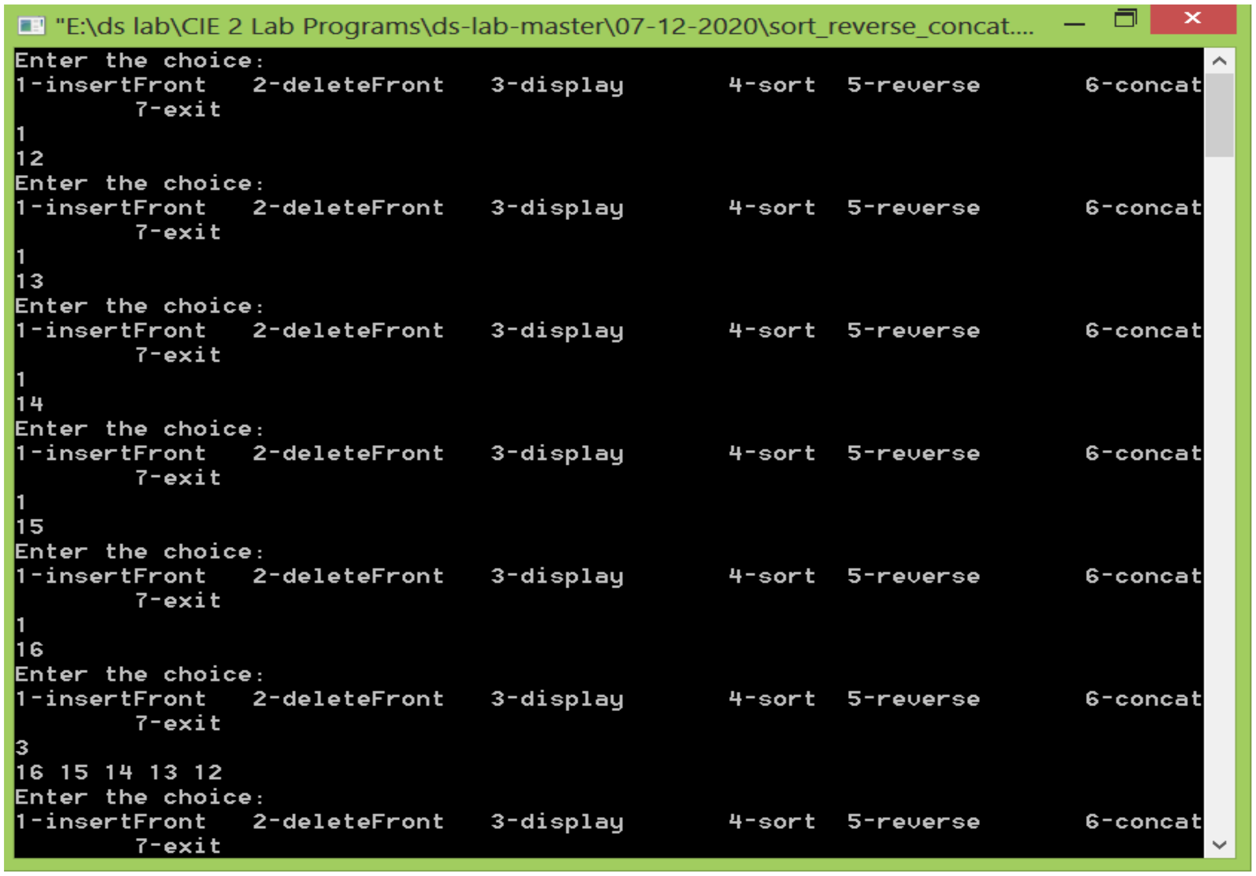
WAP Implement Single Link List with following operations
a) Sort the linked list.
b) Reverse the linked list.
c) Concatenation of two linked lists

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

struct node{
	int num;
	struct node *next;
};

typedef struct node *NODE;

NODE getNode(){
	NODE temp = (NODE)malloc(sizeof(struct node));
	if(temp == NULL){
		return NULL;
	}
	return temp;
}

void freeNode(NODE temp){
	free(temp);
}

NODE insertFront(NODE first){
	NODE temp;
	temp = getNode();
	int num;
	scanf("%d",&num);
	temp->num = num;
	temp->next = NULL;
	if(first==NULL){
```

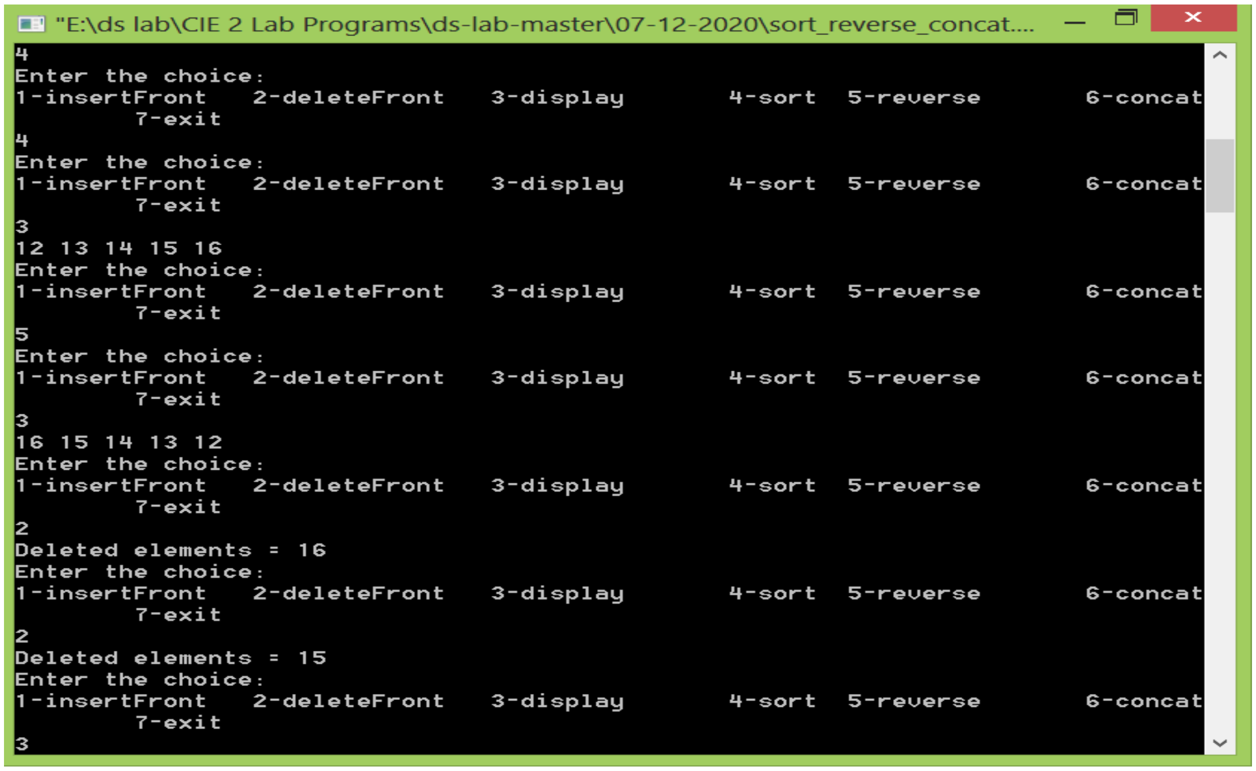
```
                return temp;
        }
        temp->next = first;
        first = temp;
        return first;
}

NODE deleteFront(NODE first){
        NODE temp;
        if(first==NULL){
                printf("List is empty\n");
                return NULL;
        }
        if(first->next == NULL){
                printf("Deleted element = %d\n",first->num);
                freeNode(first);
                return NULL;
        }
        temp = first;
        temp = temp->next;
        printf("Deleted elements = %d\n",first->num);
        freeNode(first);
        return temp;
}

NODE sort(NODE first){
        NODE curr,temp;
        if(first==NULL){
                return NULL;
        }
        curr = first;
        while(curr!=NULL){
                temp = curr->next;
                while(temp!=NULL){
                        if(temp->num<curr->num){
                                int num = curr->num;
                                curr->num=temp->num;
```

```
                    temp->num = num;
                }
                temp = temp->next;
            }
            curr = curr->next;
        }
        return first;
}

void display(NODE first){
        NODE curr;
        if(first==NULL){
            printf("List is empty\n");
            return;
        }
        curr = first;
        while(curr!=NULL){
            printf("%d ",curr->num);
            curr=curr->next;
        }
        printf("\n");
}

NODE reverse(NODE first){
        NODE curr=NULL;
        NODE temp = getNode();
        while(first!=NULL){
            temp = first;
            first = first->next;
            temp->next = curr;
            curr = temp;
            //printf("%d ",first->num);
        }
        return temp;
}

NODE concat(NODE first){
        NODE sec = NULL;
        int chq;
        while(1){
                printf("Enter the choice:\n1-insertFront\t2-deleteFront\t3-display\t4-
concat\n");
                scanf("%d",&chq);
                if(chq==4){
                        break;
                }
                switch(chq){
                        case 1:
                                sec = insertFront(sec);
                                break;
                        case 2:
                                sec = deleteFront(sec);
                                break;
                        case 3:
                                display(sec);
                                break;
                }
        }
        NODE curr;
        if(first==NULL){
                return sec;
        }
        if(sec==NULL){
                return first;
        }
        curr = first;
        while(curr->next!=NULL){
                curr = curr->next;
        }
        curr->next = sec;
        return first;
}

int main(){
        int chq;
        NODE first = NULL;
        while(1){
                printf("Enter the choice:\n1-insertFront\t2-deleteFront\t3-display\t4-
    sort\t5-reverse\t6-concat\t7-exit\n");
                scanf("%d",&chq);
                switch(chq){
                        case 1:
                                first = insertFront(first);
                                break;
                        case 2:
                                first = deleteFront(first);
                                break;
                        case 3:
                                display(first);
                                break;
                        case 4:
                                first = sort(first);
                                break;
                        case 5:
                                first = reverse(first);
                                break;
                        case 6:
                                printf("Creating the second list  for concat\n");
                                concat(first);
                                break;
                        case 7:
                                return 0;

                }
        }
}
```

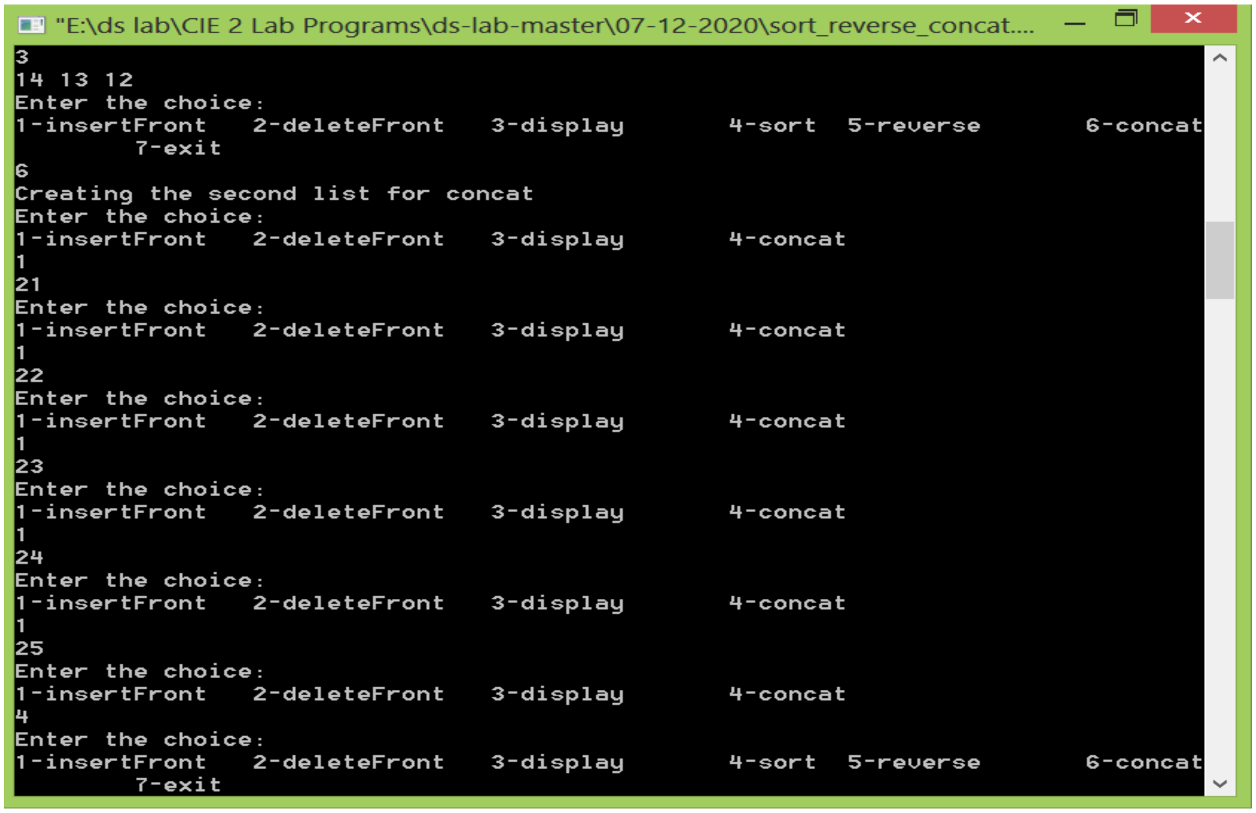
OUTPUT:


"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\07-12-2020\sort_reverse_concat....

```
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
1
12
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
1
13
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
1
14
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
1
15
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
1
16
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
3
16 15 14 13 12
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
```

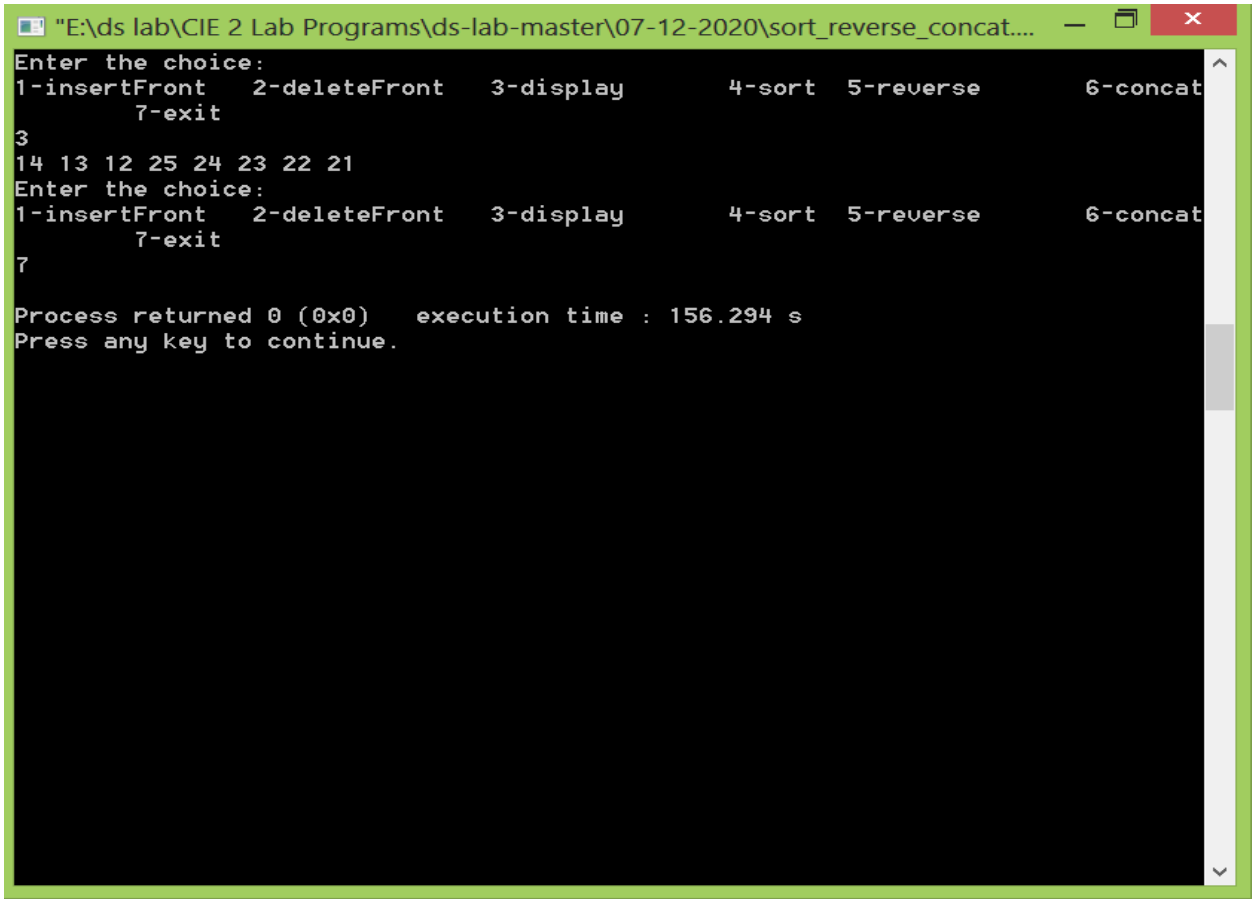

"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\07-12-2020\sort_reverse_concat....

```
4
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
4
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
3
12 13 14 15 16
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
5
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
3
16 15 14 13 12
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
2
Deleted elements = 16
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
2
Deleted elements = 15
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
3
```



"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\07-12-2020\sort_reverse_concat....

```
3
14 13 12
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
6
Creating the second list for concat
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
1
21
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
1
22
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
1
23
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
1
24
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
1
25
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-concat
4
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
```



"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\07-12-2020\sort_reverse_concat....

```
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
3
14 13 12 25 24 23 22 21
Enter the choice:
1-insertFront   2-deleteFront   3-display       4-sort  5-reverse       6-concat
        7-exit
7

Process returned 0 (0x0)   execution time : 156.294 s
Press any key to continue.
```

PROGRAM 8:

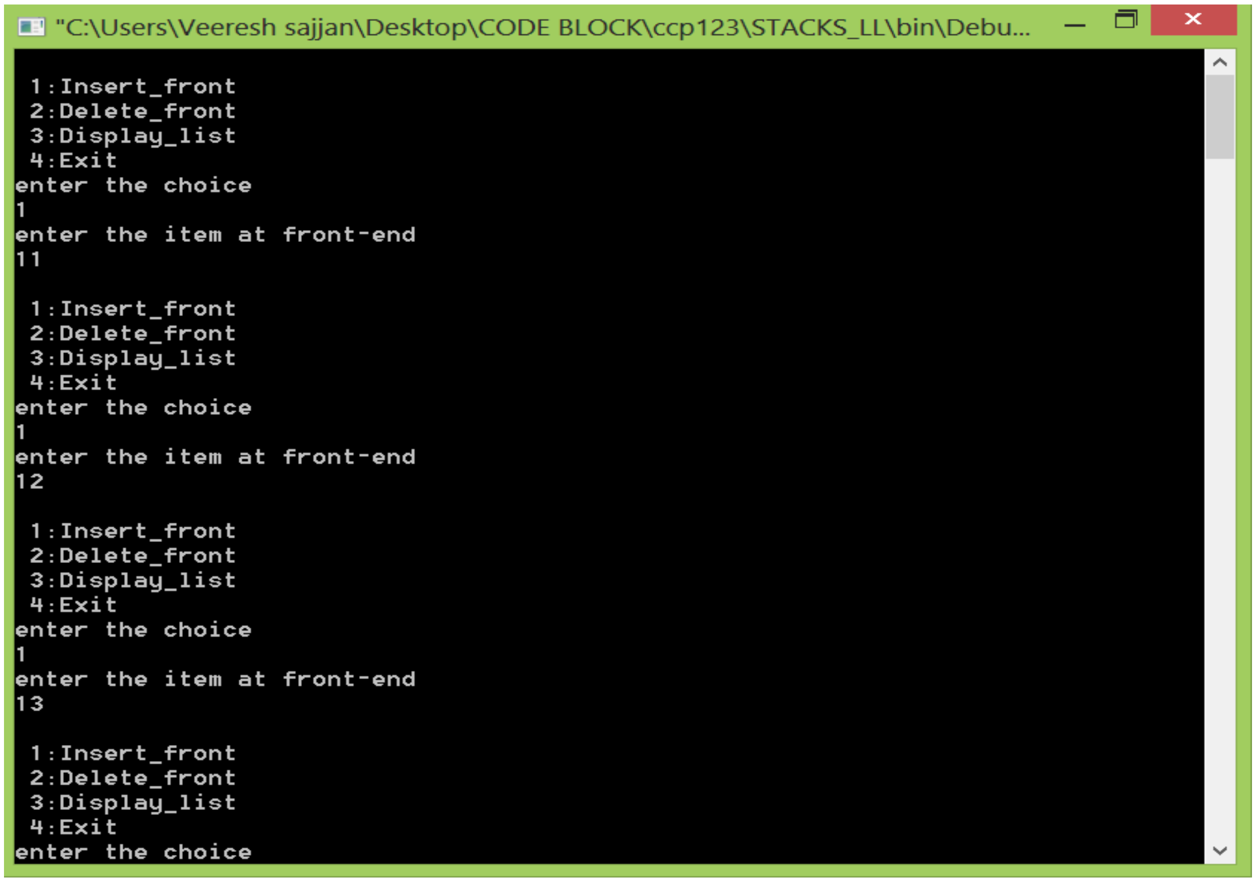
WAP to implement Stack & Queues using Linked Representation

→ **8.1 STACKS IMPLIMENTATION**

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<process.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
```

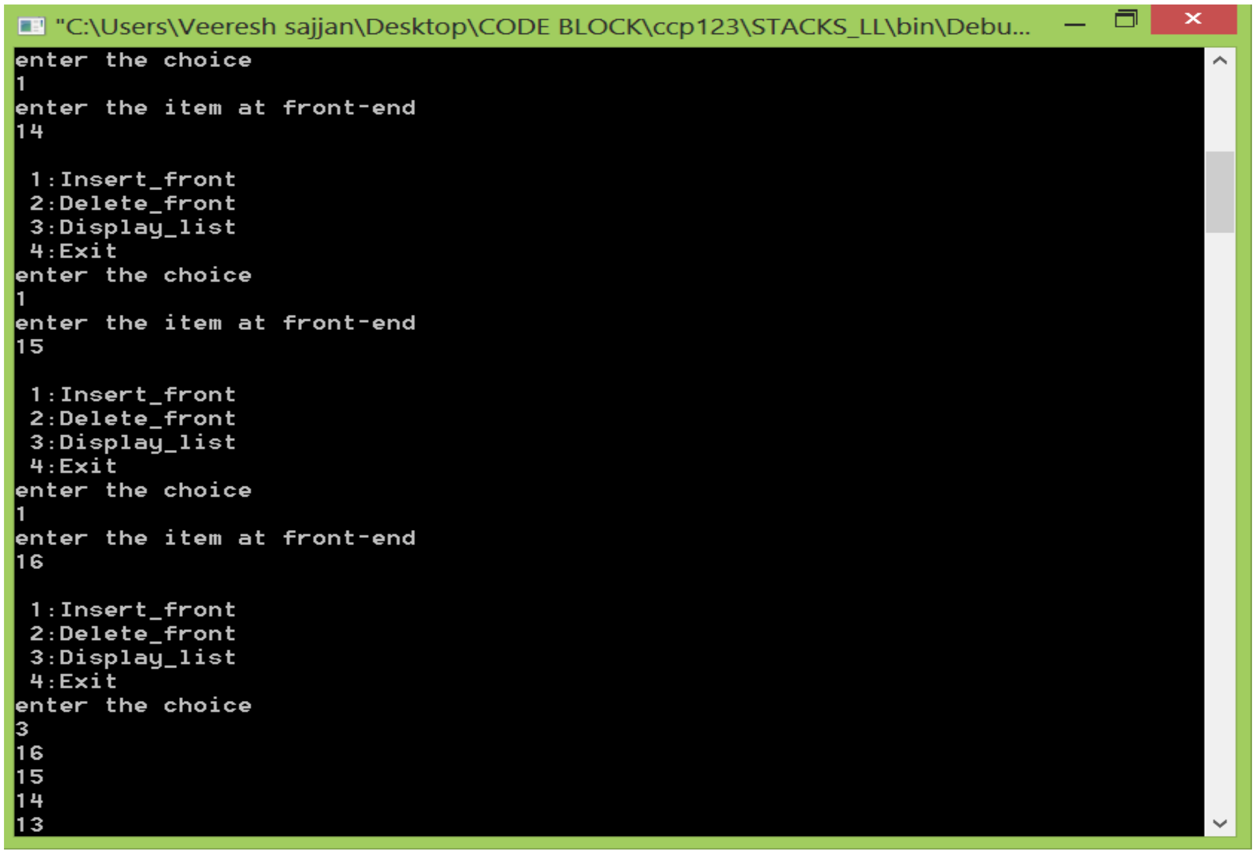
```
return temp;
temp->link=first;
first=temp;
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("stack is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
void display(NODE first)
{
 NODE temp;
 if(first==NULL)
 printf("stack empty cannot display items\n");
 for(temp=first;temp!=NULL;temp=temp->link)
  {
  printf("%d\n",temp->info);
  }
}
void main()
{
int item,choice,pos;
NODE first=NULL;
clrscr();
for(;;)
{
printf("\n 1:Insert_front\n 2:Delete_front\n 3:Display_list\n 4:Exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
 {
  case 1:printf("enter the item at front-end\n");
          scanf("%d",&item);
          first=insert_front(first,item);
          break;
  case 2:first=delete_front(first);
          break;
  case 3:display(first);
          break;
 default:exit(0);
          break;
 }
}
```

OUTPUT

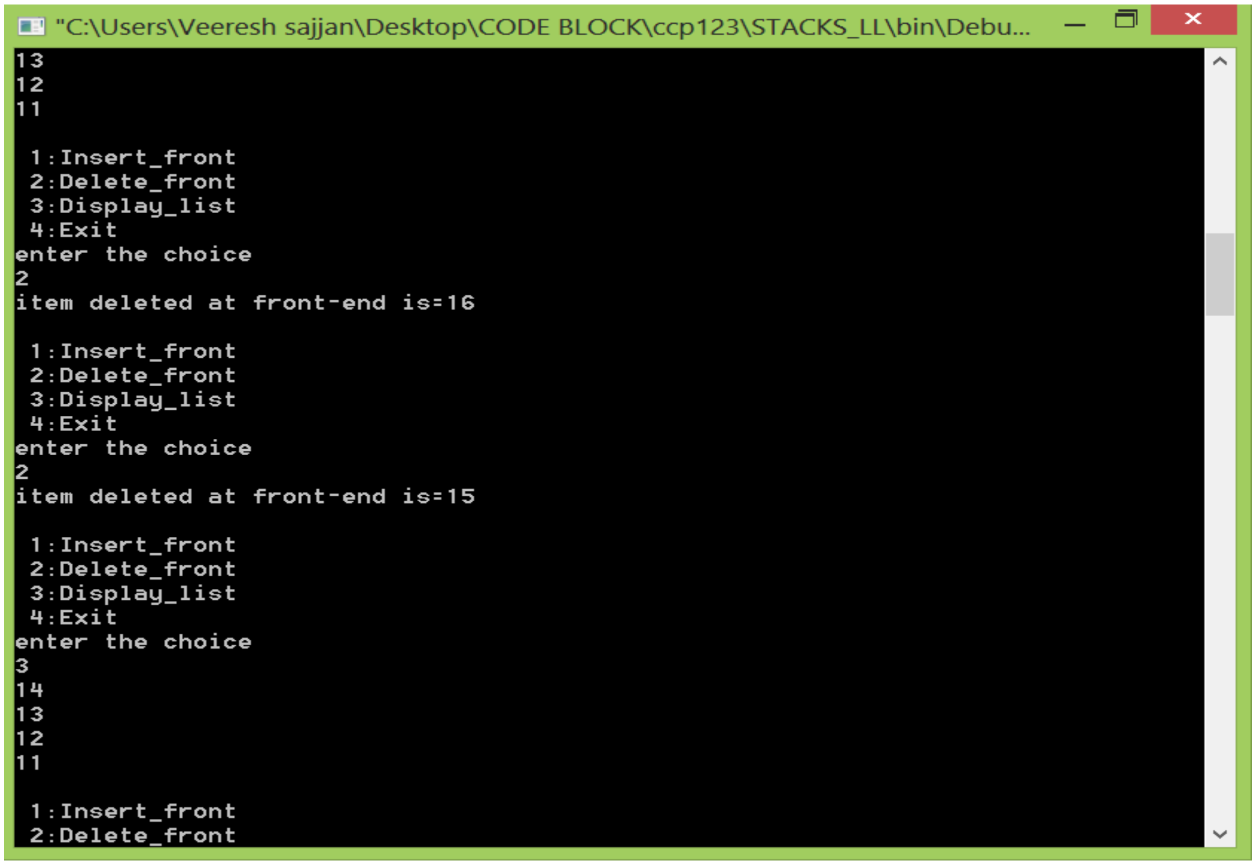


"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\STACKS_LL\bin\Debu...

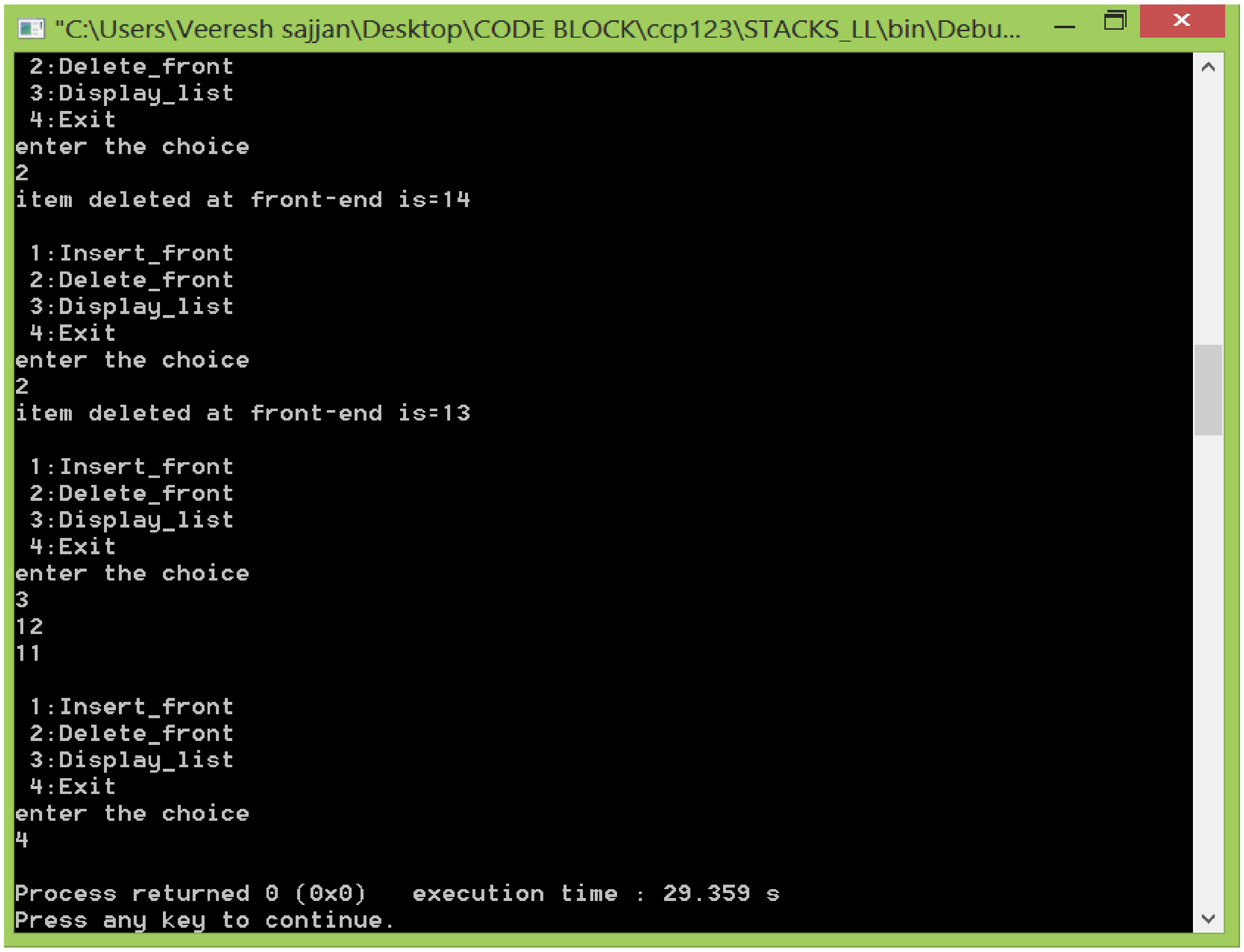
```
 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
1
enter the item at front-end
11

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
1
enter the item at front-end
12

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
1
enter the item at front-end
13

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
```



```
"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\STACKS_LL\bin\Debu...
enter the choice
1
enter the item at front-end
14

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
1
enter the item at front-end
15

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
1
enter the item at front-end
16

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
3
16
15
14
13
```



```
"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\STACKS_LL\bin\Debu...
13
12
11

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
2
item deleted at front-end is=16

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
2
item deleted at front-end is=15

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
3
14
13
12
11

 1:Insert_front
 2:Delete_front
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\STACKS_LL\bin\Debu...

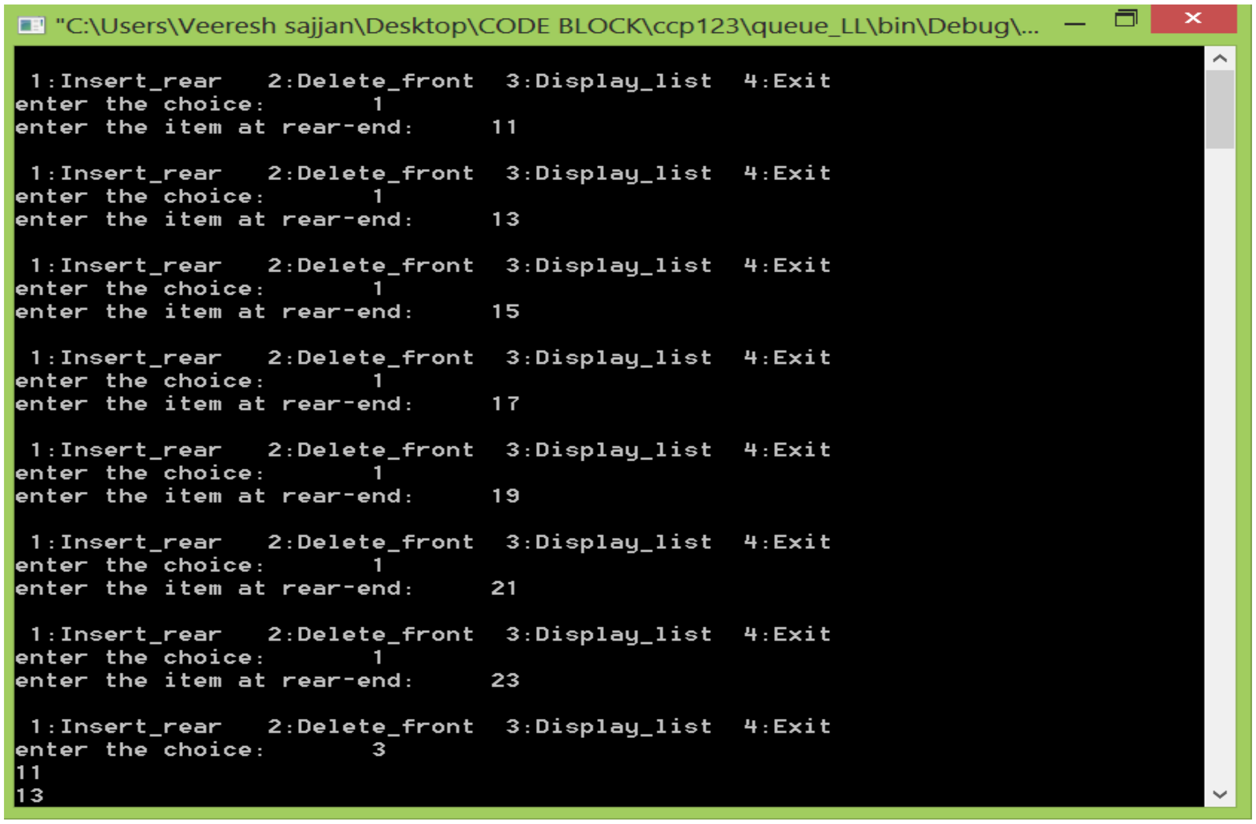
```
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
2
item deleted at front-end is=14

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
2
item deleted at front-end is=13

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
3
12
11

 1:Insert_front
 2:Delete_front
 3:Display_list
 4:Exit
enter the choice
4

Process returned 0 (0x0)   execution time : 29.359 s
Press any key to continue.
```

→ 8.2 QUEUE IMPLIMENTATION:

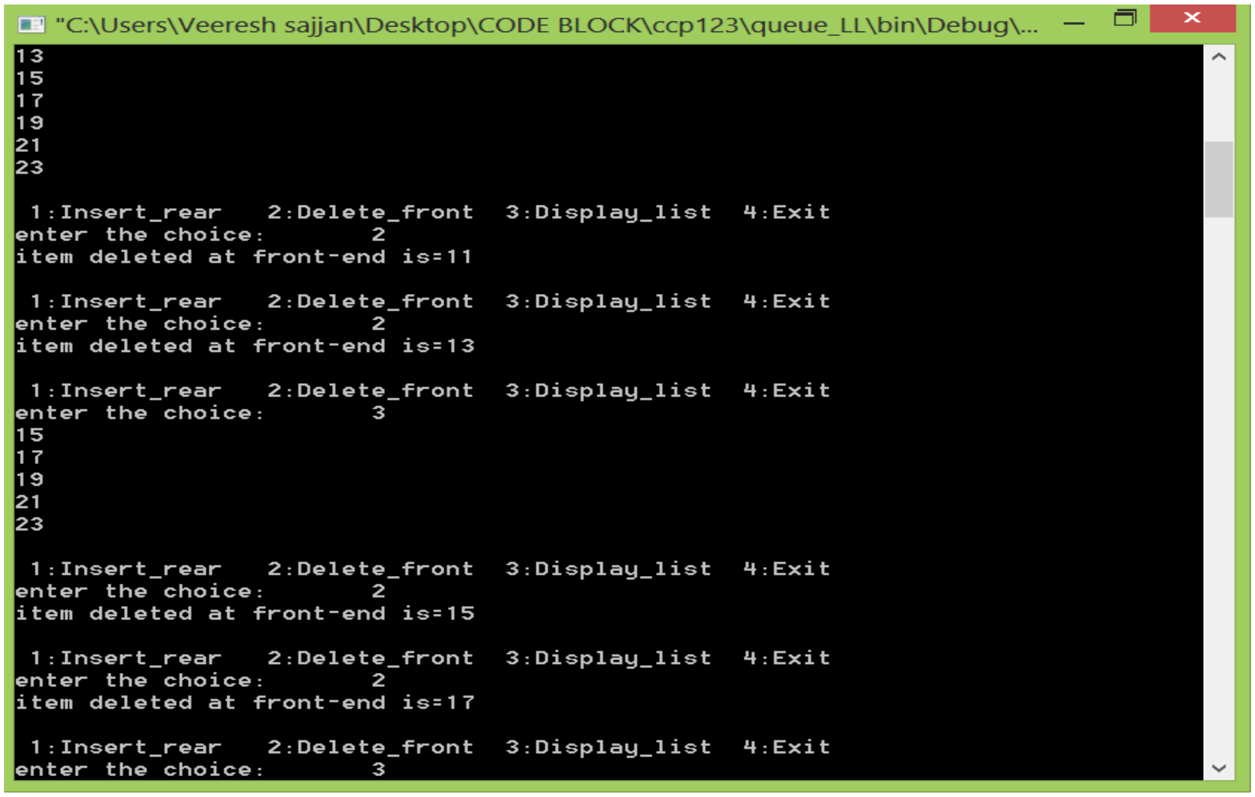
```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<process.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
 return temp;
cur=first;
while(cur->link!=NULL)
 cur=cur->link;
```

```
cur->link=temp;
return first;
}

NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
void display(NODE first)
{
 NODE temp;
 if(first==NULL)
 printf("list empty cannot display items\n");
 for(temp=first;temp!=NULL;temp=temp->link)
  {
  printf("%d\n",temp->info);
  }
}
void main()
{
int item,choice,pos;
NODE first=NULL;
for(;;)
{
printf("\n 1:Insert_rear\t 2:Delete_front\t 3:Display_list\t 4:Exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
 {
  case 1:printf("enter the item at rear-end\n");
          scanf("%d",&item);
          first=insert_rear(first,item);
          break;
  case 2:first=delete_front(first);
          break;
  case 3:display(first);
          break;
 default:exit(0);
          break;
 }
}
}
```

OUTPUT:

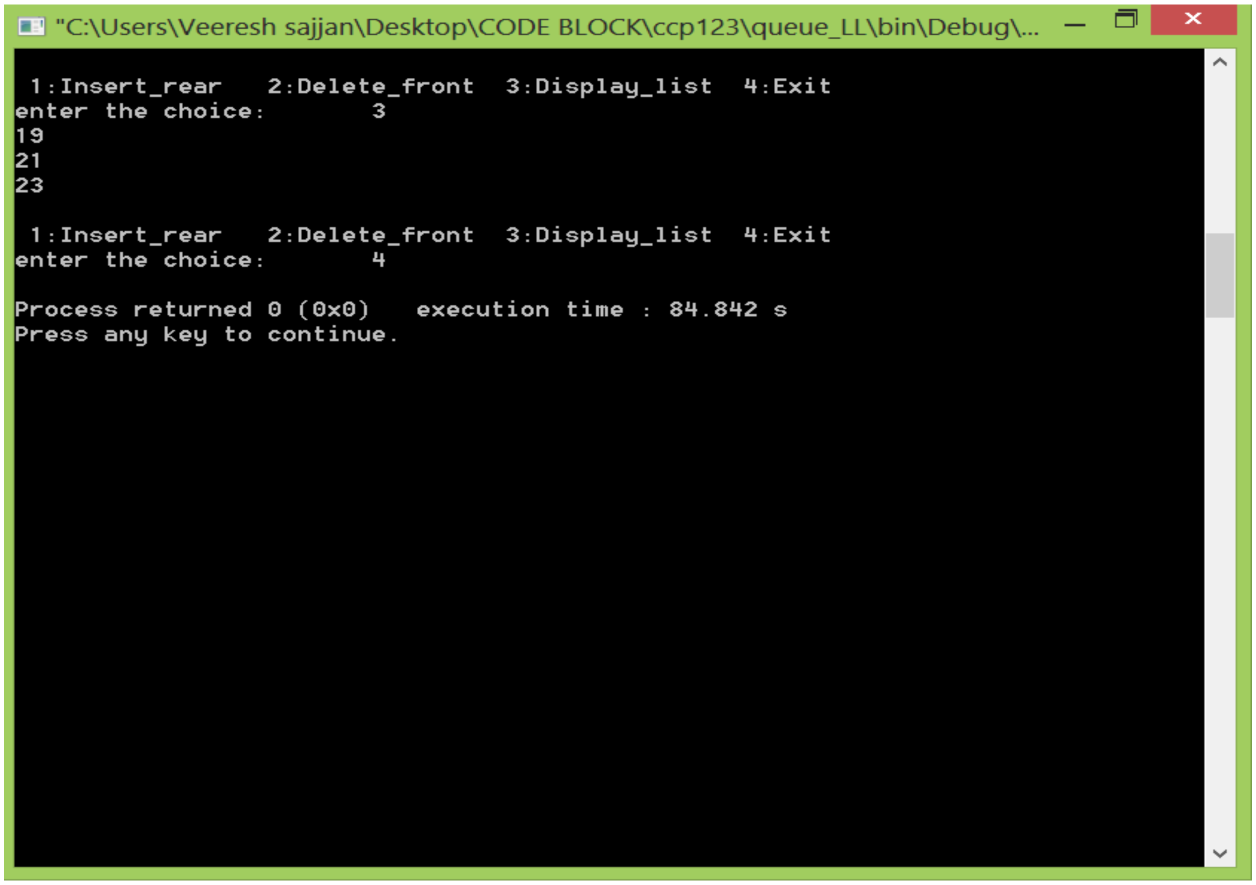


```
"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\queue_LL\bin\Debug\...

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     11

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     13

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     15

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     17

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     19

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     21

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       1
enter the item at rear-end:     23

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       3
11
13
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\queue_LL\bin\Debug\...

```
13
15
17
19
21
23

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       2
item deleted at front-end is=11

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       2
item deleted at front-end is=13

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       3
15
17
19
21
23

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       2
item deleted at front-end is=15

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       2
item deleted at front-end is=17

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       3
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\queue_LL\bin\Debug\...

```

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       3
19
21
23

 1:Insert_rear   2:Delete_front  3:Display_list  4:Exit
enter the choice:       4

Process returned 0 (0x0)   execution time : 84.842 s
Press any key to continue.
```

PROGRAM 9:

WAP Implement doubly link list with primitive operations
a) Create a doubly linked list. b) Insert a new node to the left of the node.
c) Delete the node based on a specific value d) Display the contents of the list

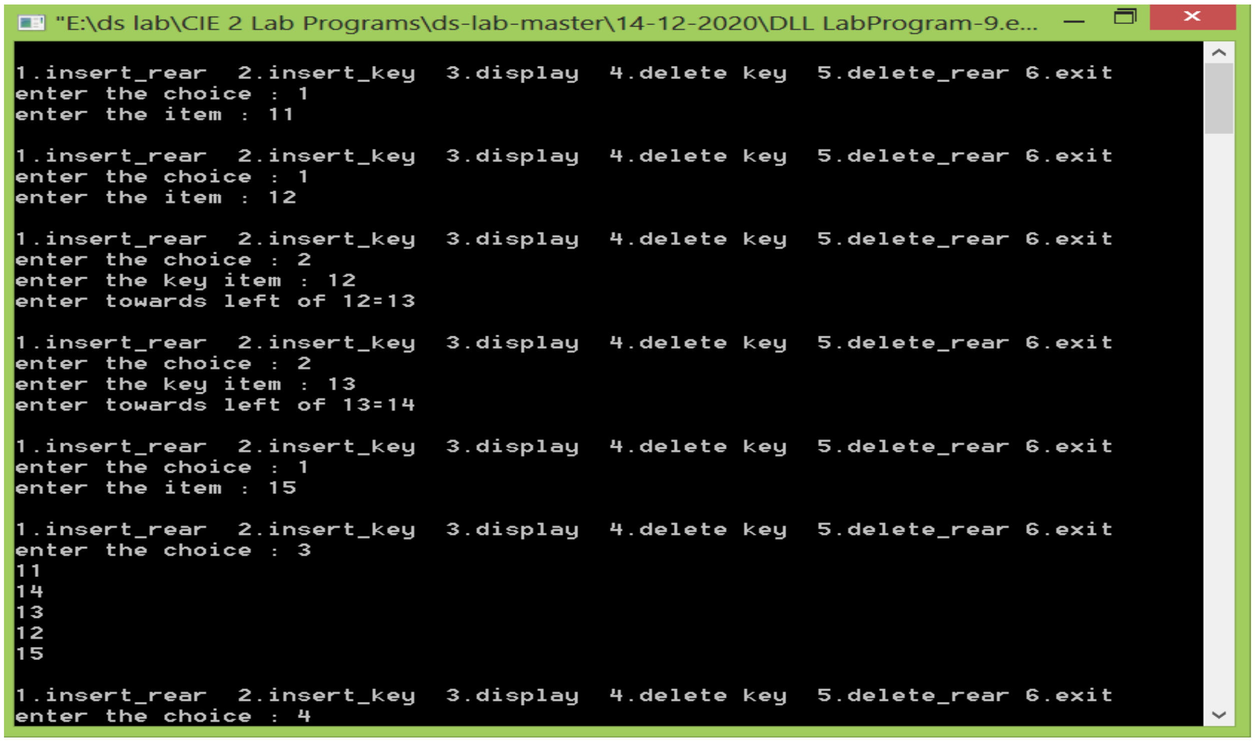
```
#include<stdio.h>
#include<stdlib.h>
struct node
 {
  int info;
  struct node *rlink;
  struct node *llink;
 };
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_rear(NODE head,int item)
{
NODE temp,cur;
temp=getnode();
temp->rlink=NULL;
temp->llink=NULL;
temp->info=item;
cur=head->llink;
temp->llink=cur;
cur->rlink=temp;
head->llink=temp;
temp->rlink=head;
head->info=head->info+1;
return head;
}
NODE insert_leftpos(int item,NODE head)
{
NODE temp,cur,prev;
if(head->rlink==head)
{
printf("list empty\n");
return head;
}
cur=head->rlink;
while(cur!=head)
{
if(item==cur->info)break;
cur=cur->rlink;
}
if(cur==head)
{
 printf("key not found\n");
 return head;
 }
 prev=cur->llink;
 printf("enter towards left of %d=",item);
 temp=getnode();
 scanf("%d",&temp->info);
 prev->rlink=temp;
 temp->llink=prev;
 cur->llink=temp;
 temp->rlink=cur;
 return head;
}
NODE delete_all_key(int item,NODE head)
```

```
{
NODE prev,cur,next;
int count;
   if(head->rlink==head)
    {
     printf("LE");
     return head;
     }
count=0;
cur=head->rlink;
while(cur!=head)
{
  if(item!=cur->info)
  cur=cur->rlink;
  else
 {
  count++;
  prev=cur->llink;
  next=cur->rlink;
  prev->rlink=next;
  next->llink=prev;
  freenode(cur);
  cur=next;
 }
}
if(count==0)
  printf("key not found");
  else
 printf("key found at %d positions and are deleted\n", count);

return head;
}
NODE ddelete_rear(NODE head)
{
NODE cur,prev;
if(head->rlink==head)
{
```

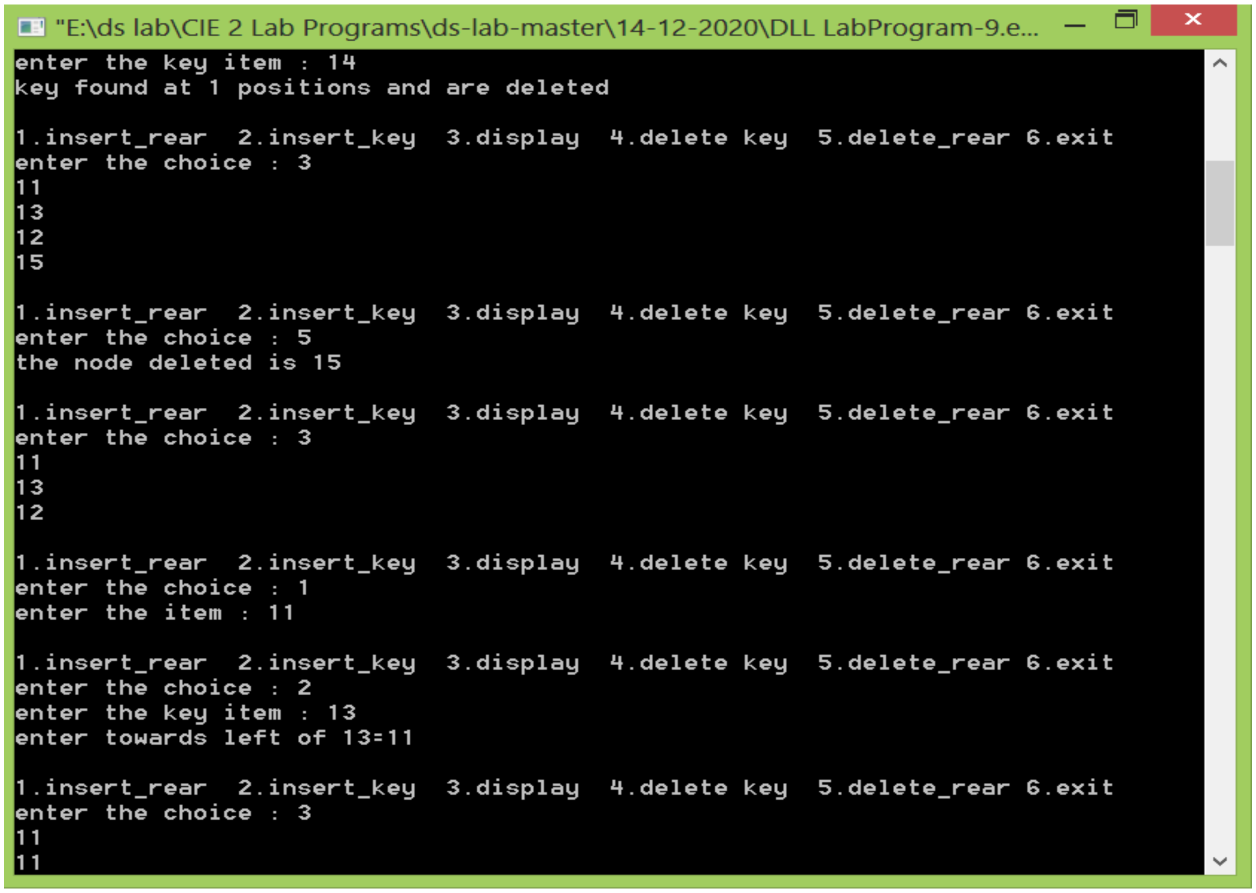
```
printf("list is empty\n");
return head;
}
cur=head->llink;
prev=cur->llink;
head->llink=prev;
prev->rlink=head;
printf("the node deleted is %d \n",cur->info);
freenode(cur);
return head;
}
void display(NODE head)
{
NODE temp;
if(head->rlink==head)
{
printf("list empty\n");
return;
}
for(temp=head->rlink;temp!=head;temp=temp->rlink)
printf("%d\n",temp->info);
}
void main()
{
int item,choice,key;
NODE head,tem;
head=getnode();
head->rlink=head;
head->llink=head;
for(;;)
{
printf("\n1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
6.exit\n");
printf("enter the choice : ");
scanf("%d",&choice);
switch(choice)
 {
```

```
case 1:printf("enter the item : ");
                scanf("%d",&item);
                head=insert_rear(head,item);
                break;
case 2:printf("enter the key item : ");
                scanf("%d",&item);
                head=insert_leftpos(item,head);
                break;
case 3:display(head);
                break;
case 4:printf("enter the key item : ");
                scanf("%d",&item);
                head=delete_all_key(item,head);
                break;
case 5:head=ddelete_rear(head);
                        break;
default:exit(0);
                 break;    }
}   }
```

OUTPUT:

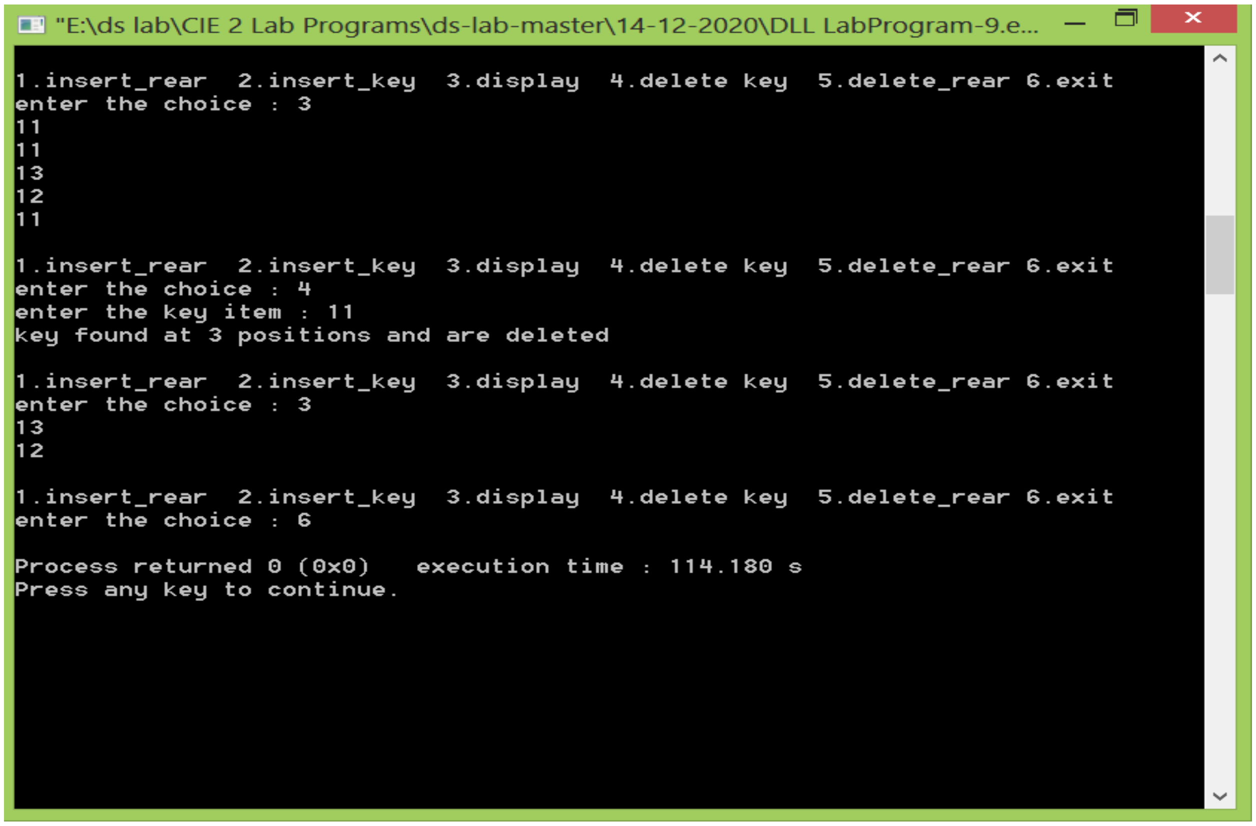


```
"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\14-12-2020\DLL LabProgram-9.e...

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 1
enter the item : 11

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 1
enter the item : 12

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 2
enter the key item : 12
enter towards left of 12=13

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 2
enter the key item : 13
enter towards left of 13=14

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 1
enter the item : 15

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
11
14
13
12
15

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 4
```



"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\14-12-2020\DLL LabProgram-9.e...

```
enter the key item : 14
key found at 1 positions and are deleted

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
11
13
12
15

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 5
the node deleted is 15

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
11
13
12

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 1
enter the item : 11

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 2
enter the key item : 13
enter towards left of 13=11

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
11
11
```



"E:\ds lab\CIE 2 Lab Programs\ds-lab-master\14-12-2020\DLL LabProgram-9.e...

```
1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
11
11
13
12
11

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 4
enter the key item : 11
key found at 3 positions and are deleted

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 3
13
12

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear 6.exit
enter the choice : 6

Process returned 0 (0x0)   execution time : 114.180 s
Press any key to continue.
```

PROGRAM 10:

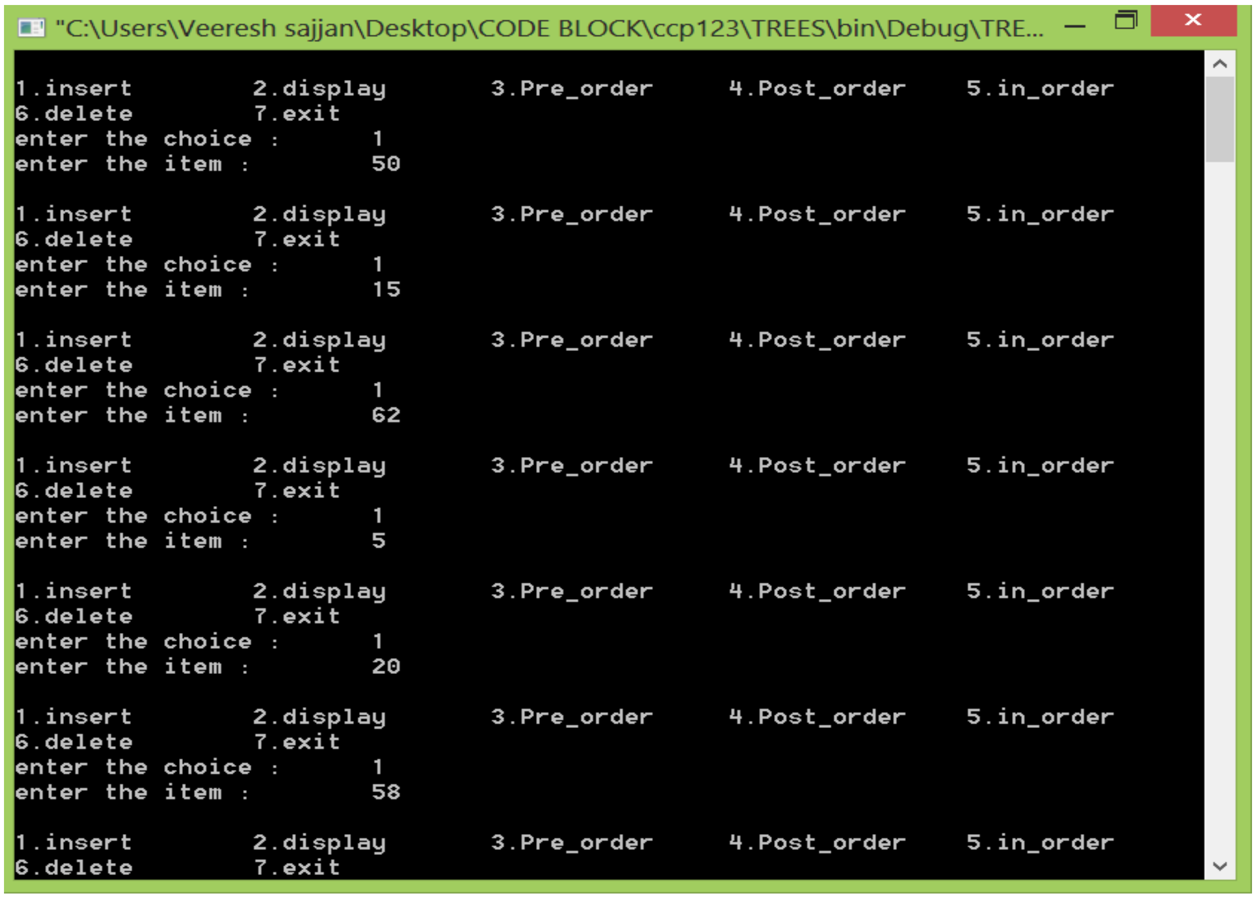
Write a program
a) To construct a binary Search tree.
b) To traverse the tree using all the methods i.e., in-order, pre order and post order
c) To display the elements in the tree.

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<process.h>
struct node
 {
  int info;
  struct node *rlink;
  struct node *llink;
 };
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
```

```
}
NODE insert(NODE root,int item)
{
NODE temp,cur,prev;
temp=getnode();
temp->rlink=NULL;
temp->llink=NULL;
temp->info=item;
if(root==NULL)
 return temp;
prev=NULL;
cur=root;
while(cur!=NULL)
{
prev=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
}
if(item<prev->info)
 prev->llink=temp;
else
 prev->rlink=temp;
return root;
}
void display(NODE root,int i)
{
int j;
if(root!=NULL)
 {
  display(root->rlink,i+1);
  for(j=0;j<i;j++)
          printf("  ");
   printf("%d\n",root->info);
         display(root->llink,i+1);
 }
}
NODE delete(NODE root,int item)
{
NODE cur,parent,q,suc;
if(root==NULL)
{
printf("empty\n");
return root;
}
parent=NULL;
cur=root;
while(cur!=NULL&&item!=cur->info)
{
parent=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
}
if(cur==NULL)
{
 printf("not found\n");
 return root;
}
if(cur->llink==NULL)
 q=cur->rlink;
else if(cur->rlink==NULL)
 q=cur->llink;
else
 {
 suc=cur->rlink;
 while(suc->llink!=NULL)
  suc=suc->llink;
 suc->llink=cur->llink;
 q=cur->rlink;
 }
 if(parent==NULL)
  return q;
 if(cur==parent->llink)
  parent->llink=q;
 else
  parent->rlink=q;
 freenode(cur);
 return root;
 }

void preorder(NODE root)
{
if(root!=NULL)
 {
  printf("%d\n",root->info);
  preorder(root->llink);
  preorder(root->rlink);
  }
 }
void postorder(NODE root)
{
if(root!=NULL)
 {

  postorder(root->llink);
  postorder(root->rlink);
  printf("%d\n",root->info);
  }
 }
void inorder(NODE root)
{
if(root!=NULL)
```

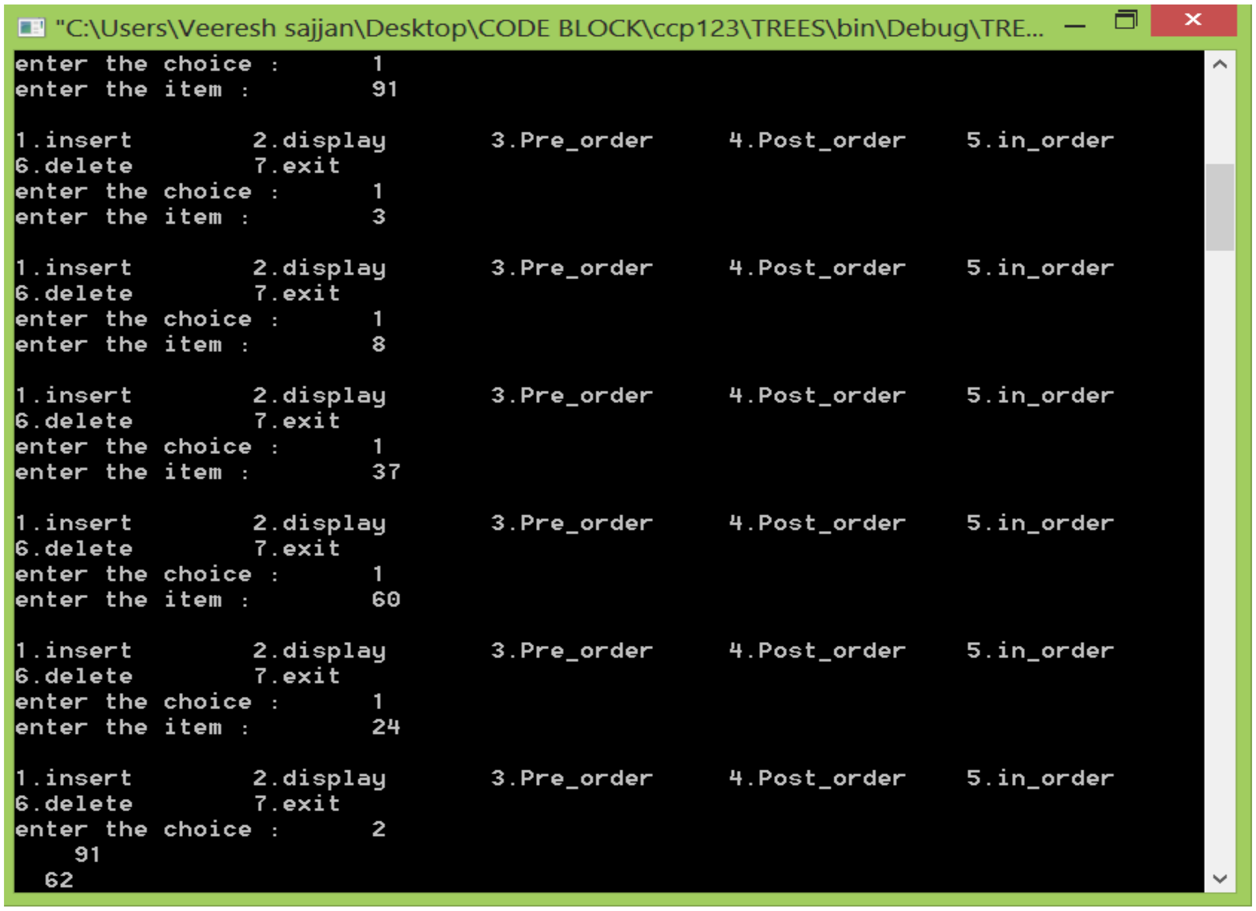
```
 {

  inorder(root->llink);
  printf("%d\n",root->info);
  inorder(root->rlink);
  }
 }
void main()
{
int item,choice;
NODE root=NULL;
clrscr();
for(;;)
{
printf("\n1.insert\n2.display\n3.pre\n4.post\n5.in\n6.delete\n7.exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
 {
  case 1:printf("enter the item\n");
                scanf("%d",&item);
                root=insert(root,item);
                break;
  case 2:display(root,0);
                break;
  case 3:preorder(root);
                break;
  case 4:postorder(root);
                break;
  case 5:inorder(root);
                break;
  case 6:printf("enter the item\n");
                scanf("%d",&item);
                root=delete(root,item);
                break;
default: exit(0);
                 break;
            }
        }
}
```

OUTPUT:

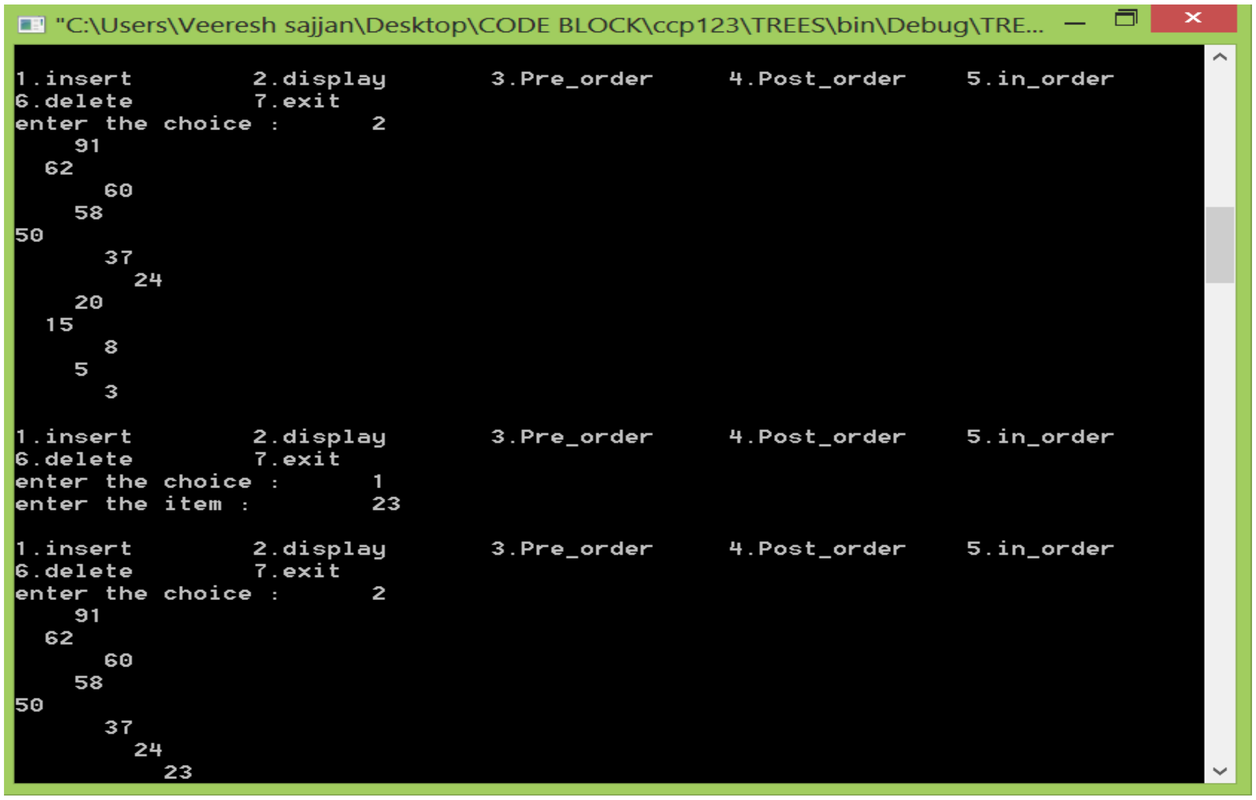


"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\TREES\bin\Debug\TRE...

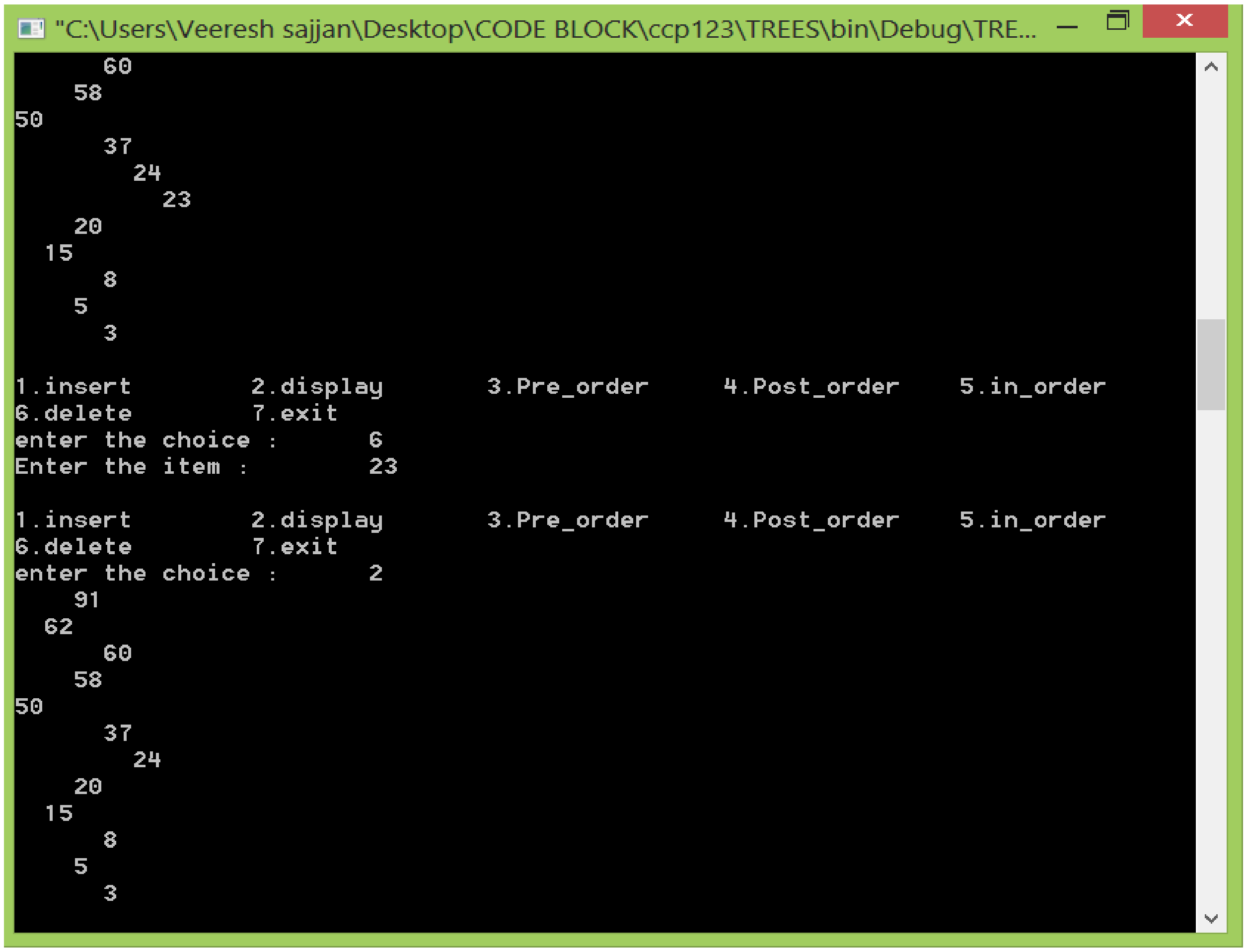
```
1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         50

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         15

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         62

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         5

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         20

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :       1
enter the item :         58

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
```


"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\TREES\bin\Debug\TRE...

```
enter the choice :        1
enter the item :          91

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          3

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          8

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          37

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          60

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          24

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        2
    91
  62
```

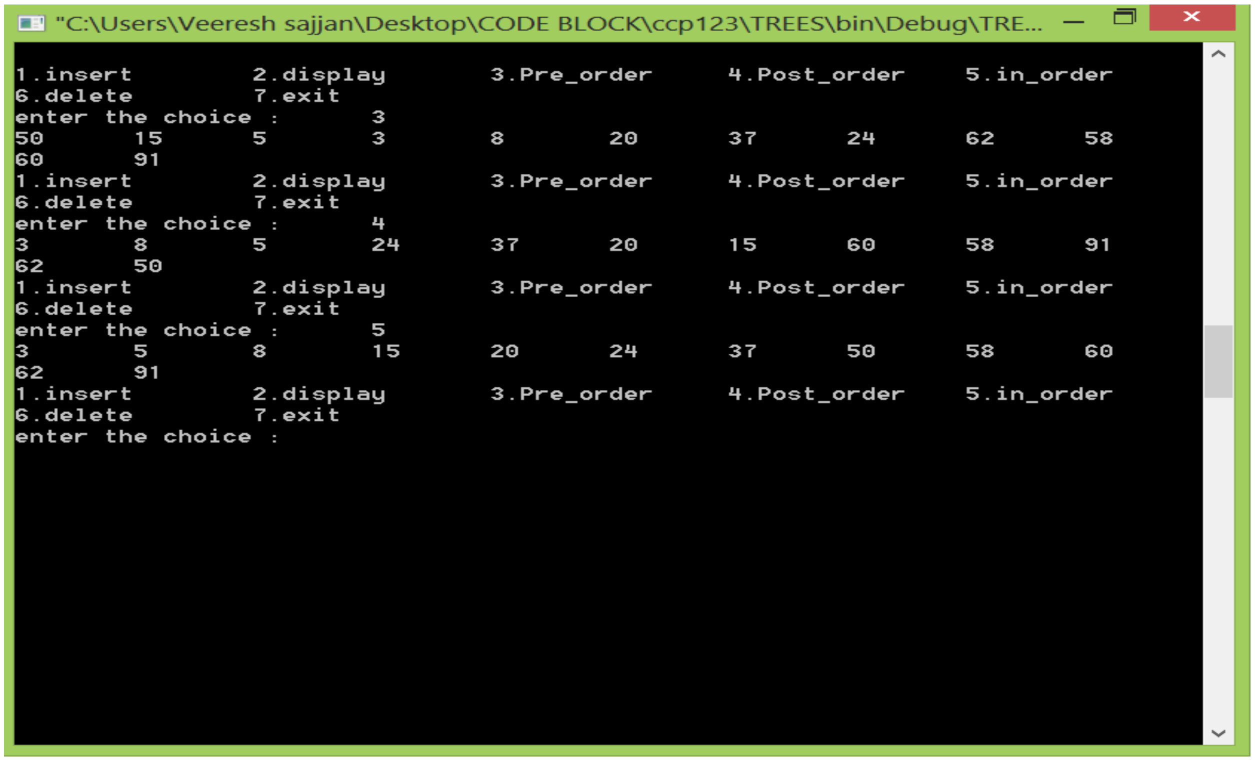

"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\TREES\bin\Debug\TRE...

```

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        2
    91
  62
      60
    58
50
      37
        24
    20
  15
      8
    5
      3

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        1
enter the item :          23

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :        2
    91
  62
      60
    58
50
      37
        24
          23
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\TREES\bin\Debug\TRE...

```
      60
    58
50
      37
        24
          23
    20
  15
      8
    5
      3

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :      6
Enter the item :        23

1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :      2
    91
  62
      60
    58
50
      37
        24
    20
  15
      8
    5
      3
```



"C:\Users\Veeresh sajjan\Desktop\CODE BLOCK\ccp123\TREES\bin\Debug\TRE...

```
1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :      3
50      15      5       3       8       20      37      24      62      58
60      91
1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :      4
3       8       5       24      37      20      15      60      58      91
62      50
1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :      5
3       5       8       15      20      24      37      50      58      60
62      91
1.insert        2.display       3.Pre_order     4.Post_order    5.in_order
6.delete        7.exit
enter the choice :
```