DATA STRUCTURE LAB RECORD

NAME : ANSHUL SURANA

USN : 1BM19CS020

COURSE : DATA STRUCTURE LAB

DEPARTMENT : CSE

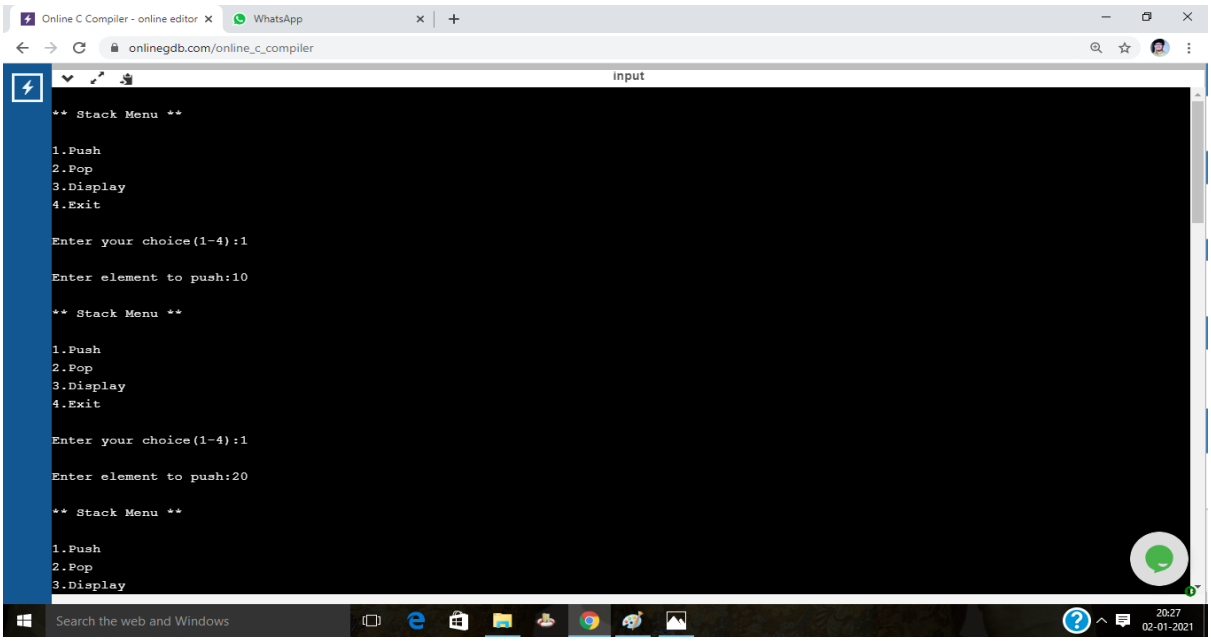
SECTION : 3A

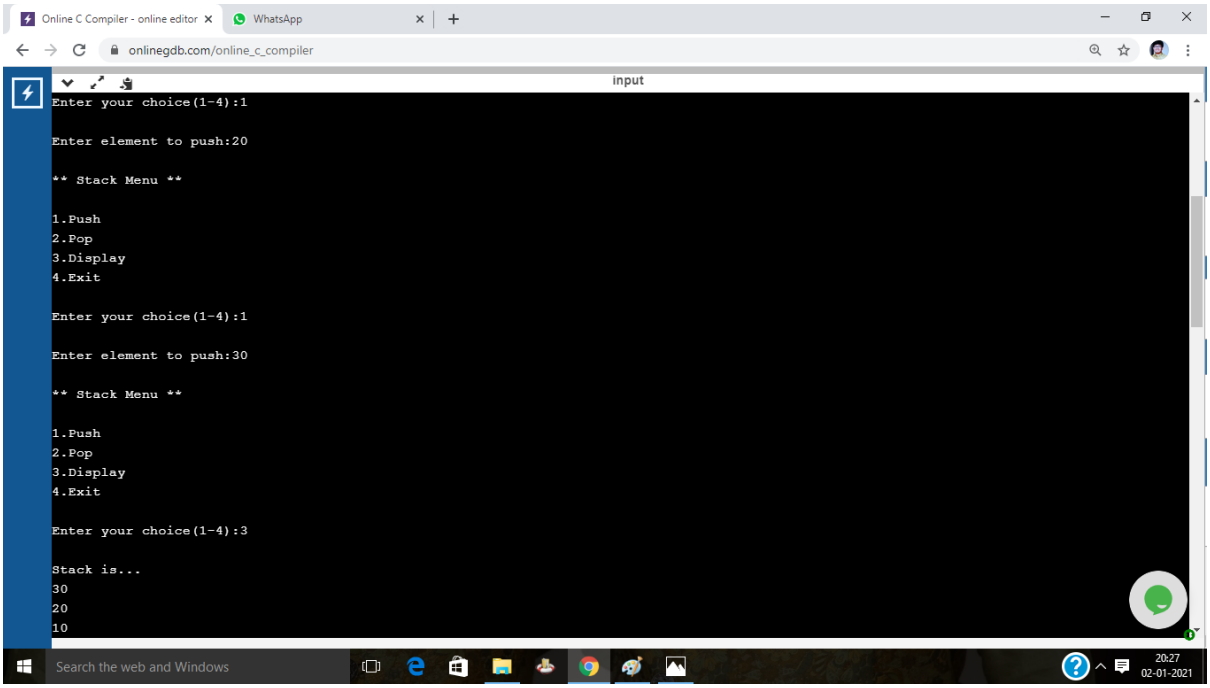
ACADEMIC YEAR : 2020-2021

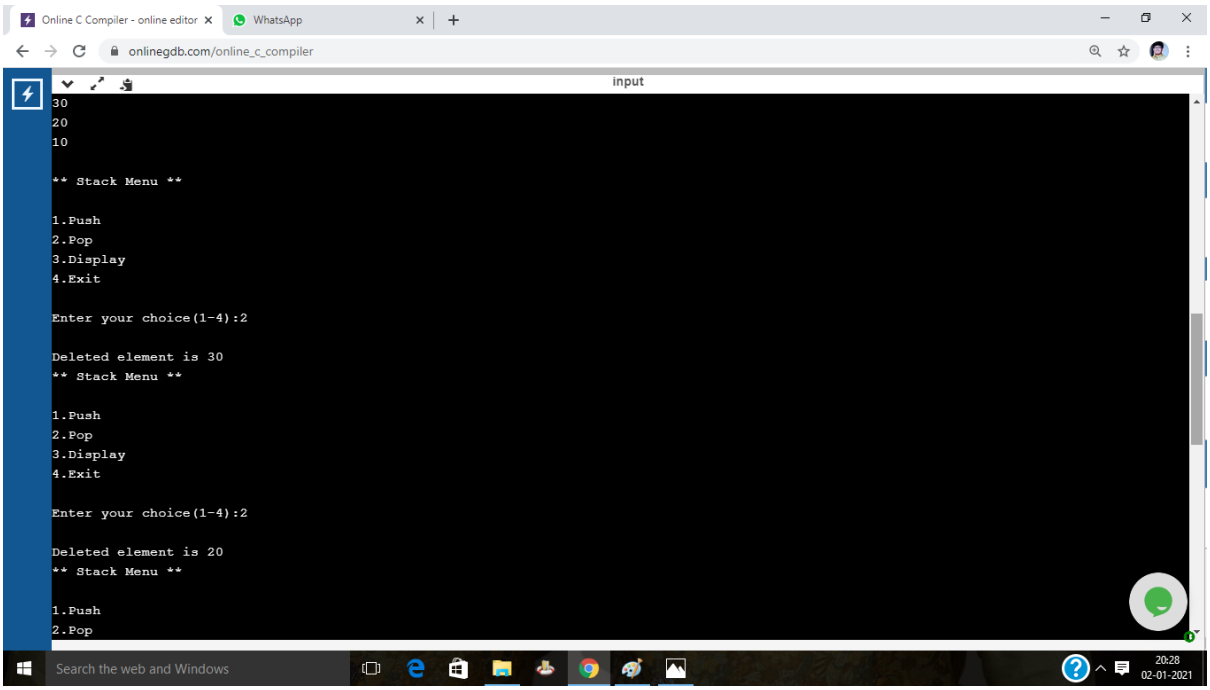
LAB PROGRAM -1

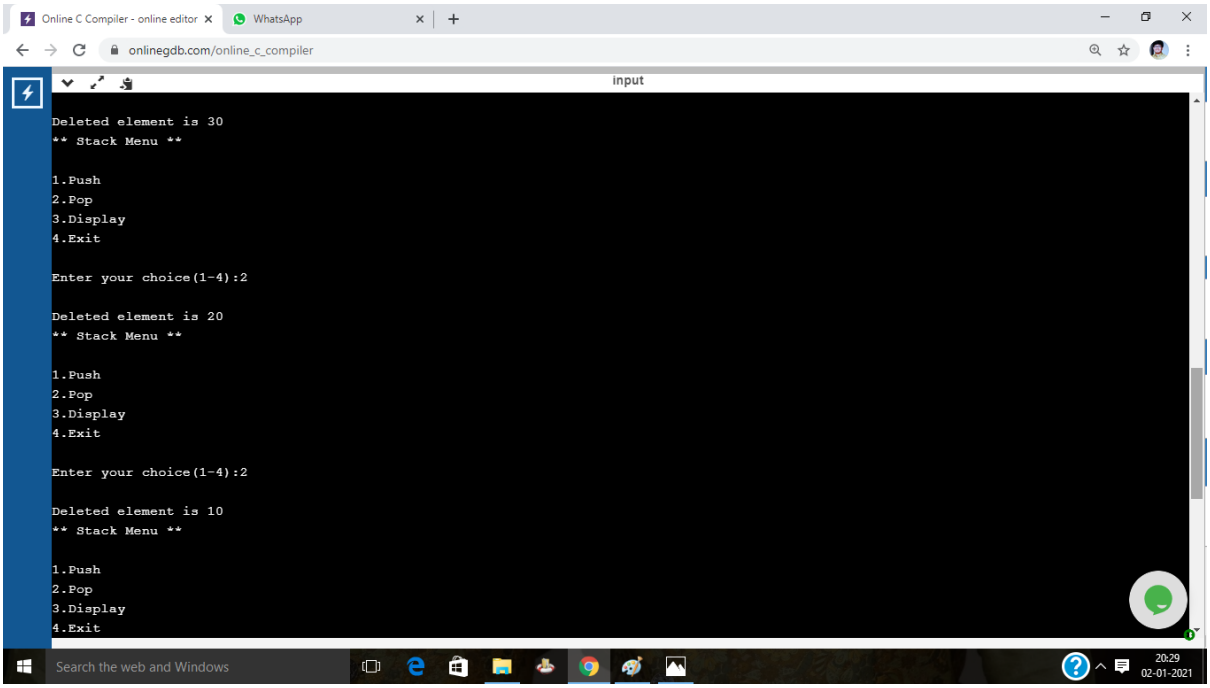
```
#include<stdio.h>
                    #include<process.h>
                    #include<stdlib.h>
                    #define MAX 5     //Maximum number of elements that can
                    be stored
                    int top=-1,stack[MAX];
                    void push();
                    void pop();
                    void display();
                    void main()
                    {
                          int ch;
                          while(1)    //infinite loop, will end when choice
                    will be 4
                          {
                                printf("\n*** Stack Menu ***");

                          printf("\n\n1.Push\n2.Pop\n3.Display\n4.Exit");
                                printf("\n\nEnter your choice(1-4):");
                                scanf("%d",&ch);
                                switch(ch)
                                {
                                      case 1: push();
                                                  break;
                                      case 2: pop();
                                                  break;
                                      case 3: display();
                                                  break;
                                      case 4: exit(0);
                                      default: printf("\nWrong Choice!!");
                                }
                          }
                    }
                    void push()
                    {
                          int val;
                          if(top==MAX-1)
                          {
                                printf("\nStack is full!!");
                          }
                          else
                          {
                                printf("\nEnter element to push:");
                                scanf("%d",&val);
                                top=top+1;
                                stack[top]=val;
                          }
                    }
                    void pop()
                    {
                          if(top==-1)
```

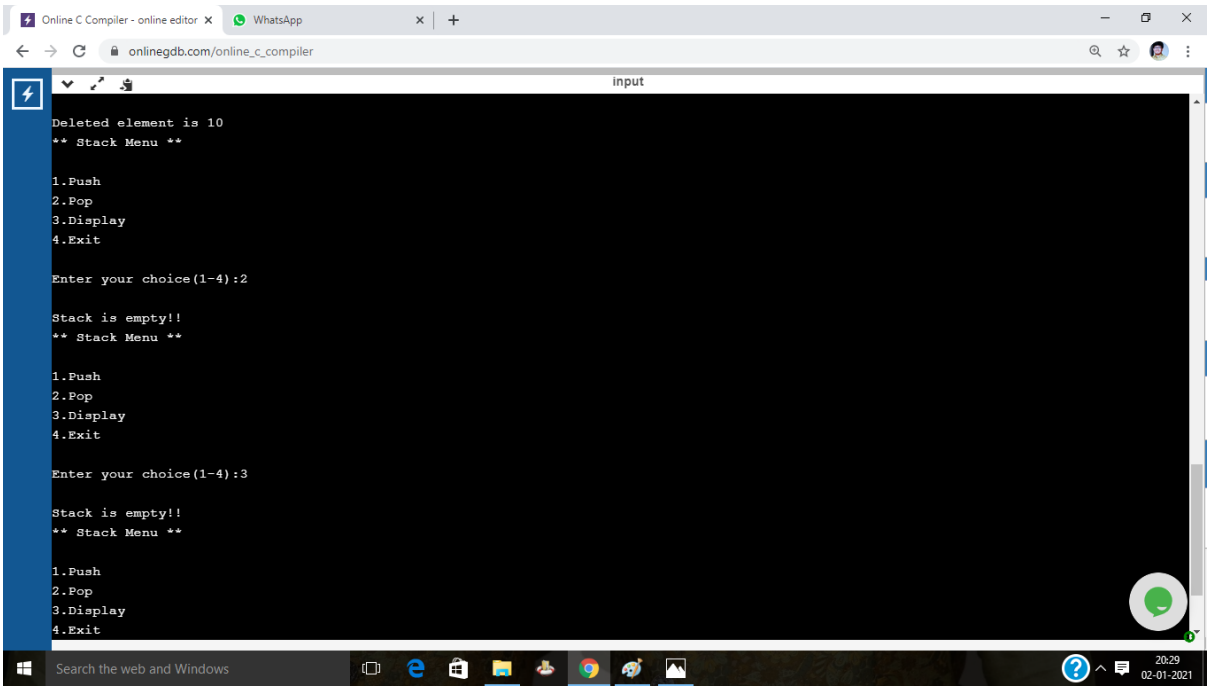
```
        {
                printf("\nStack is empty!!");
        }
        else
        {
                printf("\nDeleted element is
%d",stack[top]);
                top=top-1;
        }
}
void display()
{
        int i;
        if(top==-1)
        {
                printf("\nStack is empty!!");
        }
        else
        {
                printf("\nStack is...\n");
                for(i=top;i>=0;--i)
                        printf("%d\n",stack[i]);
        }
}
```

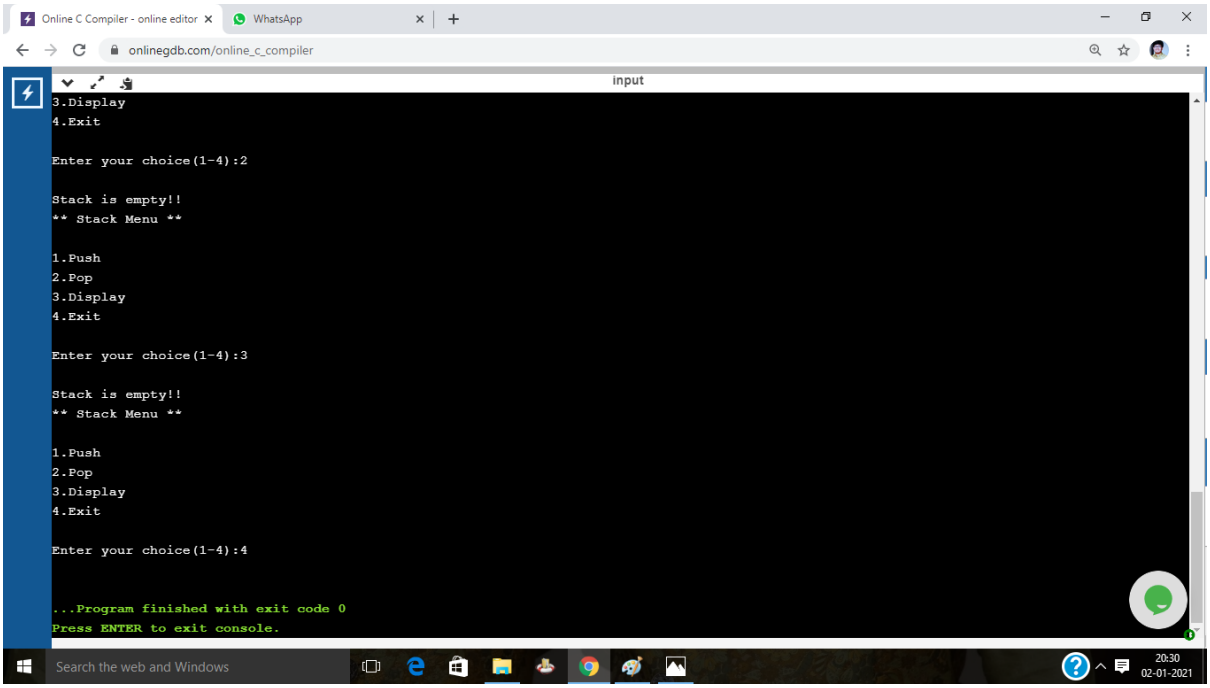



Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
Enter your choice(1-4):1
Enter element to push:20
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):1
Enter element to push:30
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):3
Stack is...
30
20
10
Search the web and Windows
20:27
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
30
20
10
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):2
Deleted element is 30
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):2
Deleted element is 20
** Stack Menu **
1.Push
2.Pop
Search the web and Windows
20:28
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
Deleted element is 30
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):2
Deleted element is 20
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):2
Deleted element is 10
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Search the web and Windows
20:29
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
Deleted element is 10
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):2
Stack is empty!!
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Enter your choice(1-4):3
Stack is empty!!
** Stack Menu **
1.Push
2.Pop
3.Display
4.Exit
Search the web and Windows
20:29
02-01-2021

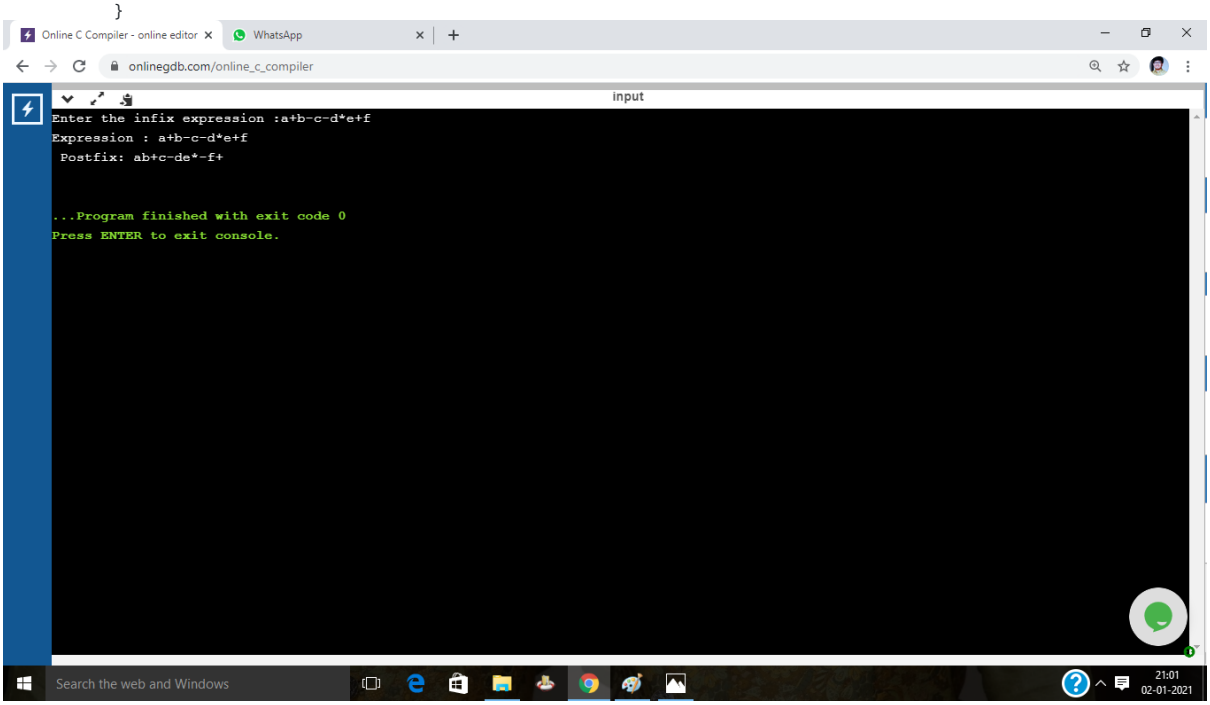


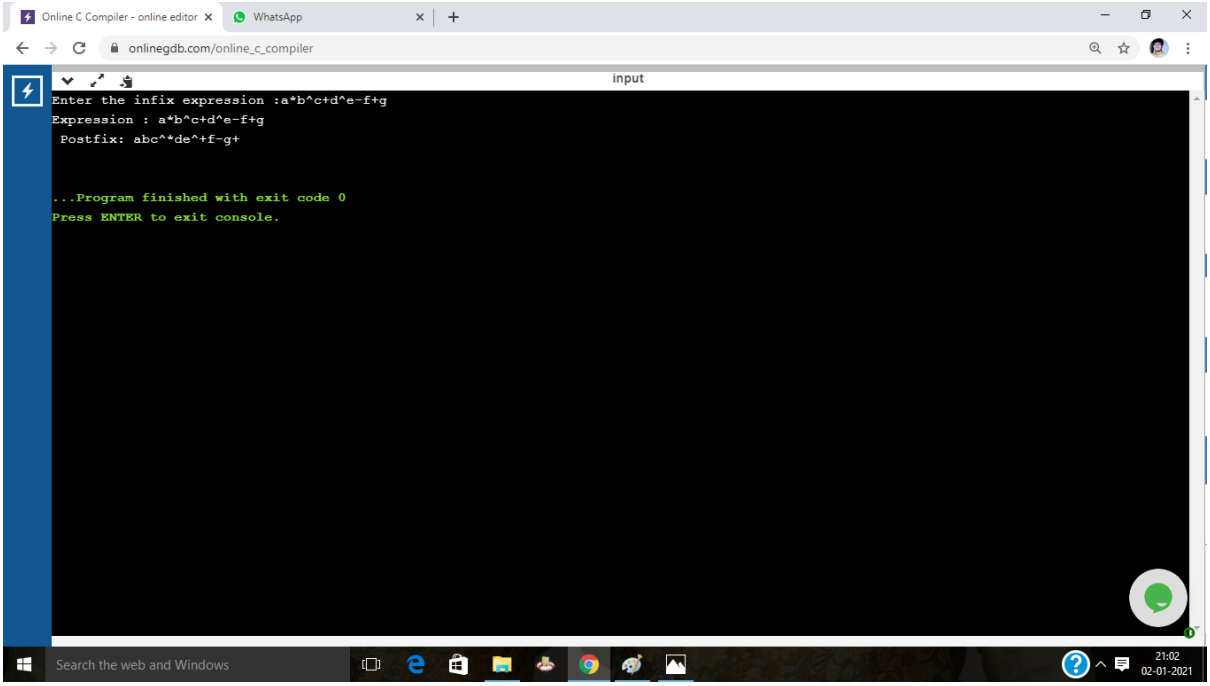
LAB PROGRAM – 2

```
#include
<stdio.h>
            # define MAX 100
            char stack[MAX];
            int top=-1;
            void push(char ch)
            {
                if (top==MAX-1)
                    printf("Stack is full\n");
                else
                {
                    top++;
                    stack[top]=ch;
                }
            }
            char pop()
            {
                char item;
                if (top==-1)
                    printf("\n stack is empty !");
                else
                {
                    item=stack[top];
                    top--;
                    return item;
                }

            }
            int stackempty()
            {
                if(top==-1) return 1;
                else return 0;
            }
            char stacktop()
            {
                if( top==-1)
```

```
        printf("\n stack is empty!");
    else
        return stack[top];
}
int priority(char ch)
{
    switch(ch)
    {
        case '+':
        case '-':return (1);
        case '*':
        case '/':return (2);
        case '^': return (3);
        default : return (0);
    }
}



int main(int argc, char **argv)
{
          char infix[100];
    int i, item;
    printf("Enter the infix expression :");
    scanf("%s",infix);
    printf("Expression : %s",infix);
    printf("\n Postfix: ");
    i=0;
     while (infix[i]!='\0')
    {


        switch (infix[i])
        {
            case '(': push(infix[i]);
                      break;
            case ')':while(( item=pop())!='(')
                       printf("%c",item);
                      break;
            case '+':
            case '-':
            case '*':
            case '/':
            case '^':
                      while(!stackempty() && priority(infix[i])<=priority(stacktop()))
                      {
                            item=pop();

                            printf("%c", item);
                      }

                      push(infix[i]);
                      break;
            default : printf("%c", infix[i]);
                      break;


        }
         i++;
    }

    while(!stackempty())
    {
        char item;
```

```
            item=pop();
            printf("%c", item);

        }
        printf("\n");
        return 0;
    }
```

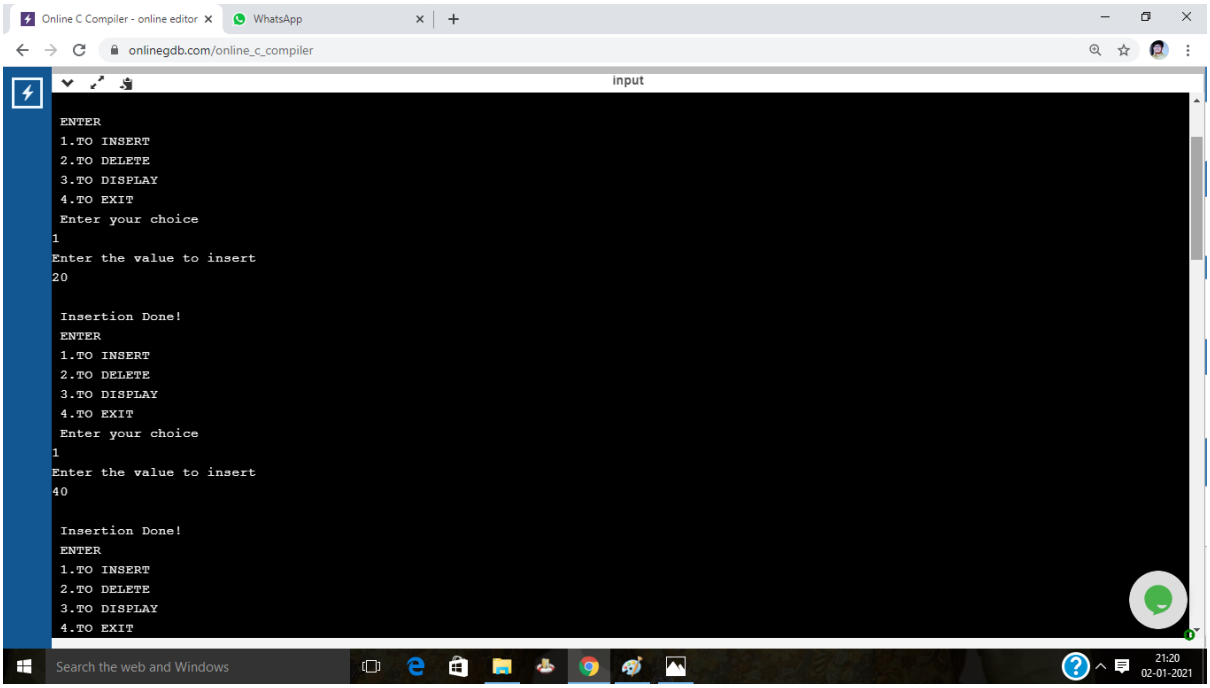


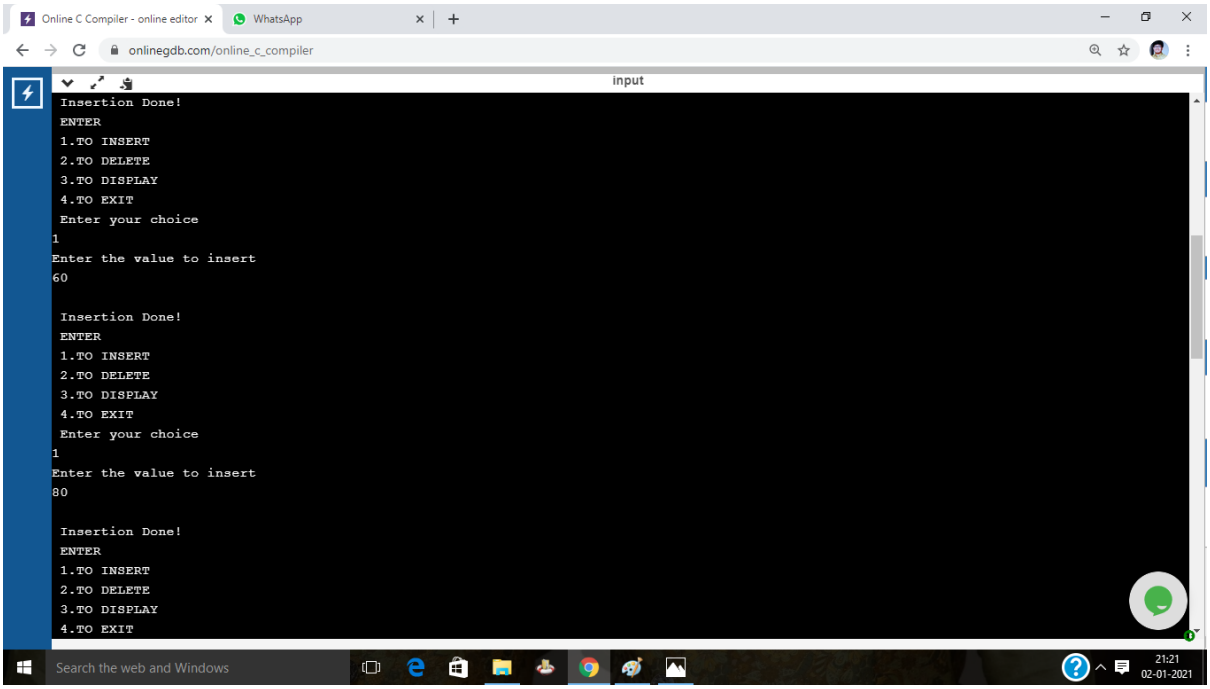


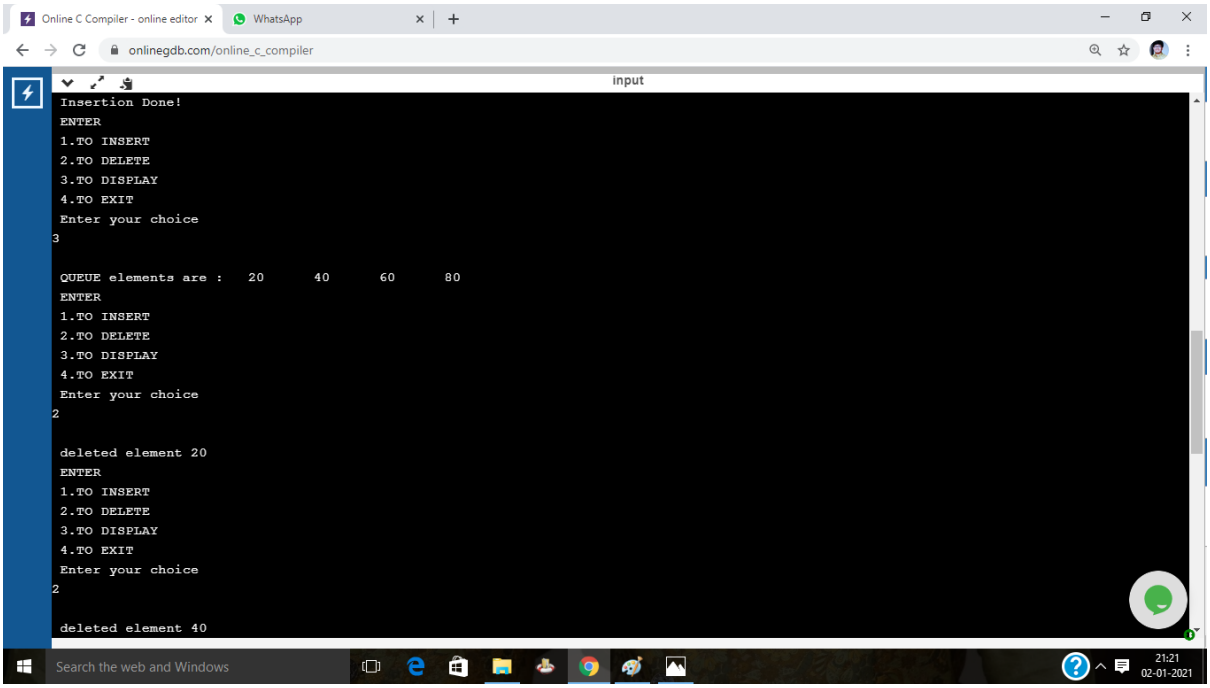
LAB PROGRAM – 3

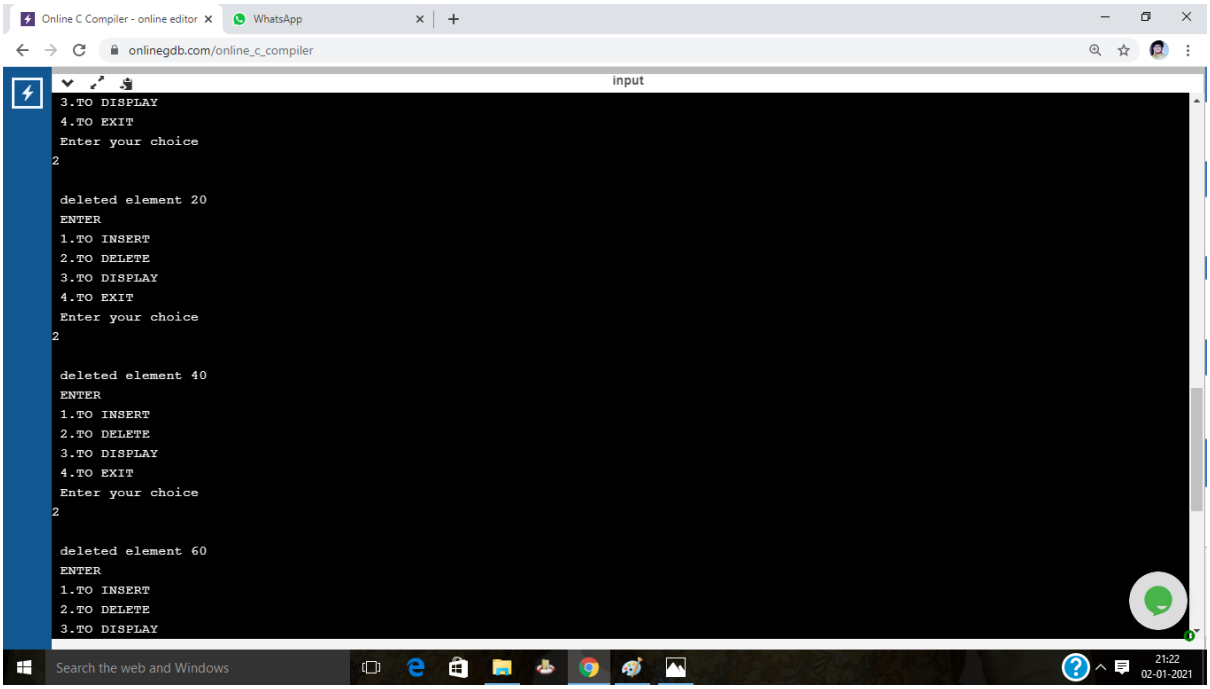
```
#include
<stdio.h>
            #include <stdlib.h>
            #define N 5
```

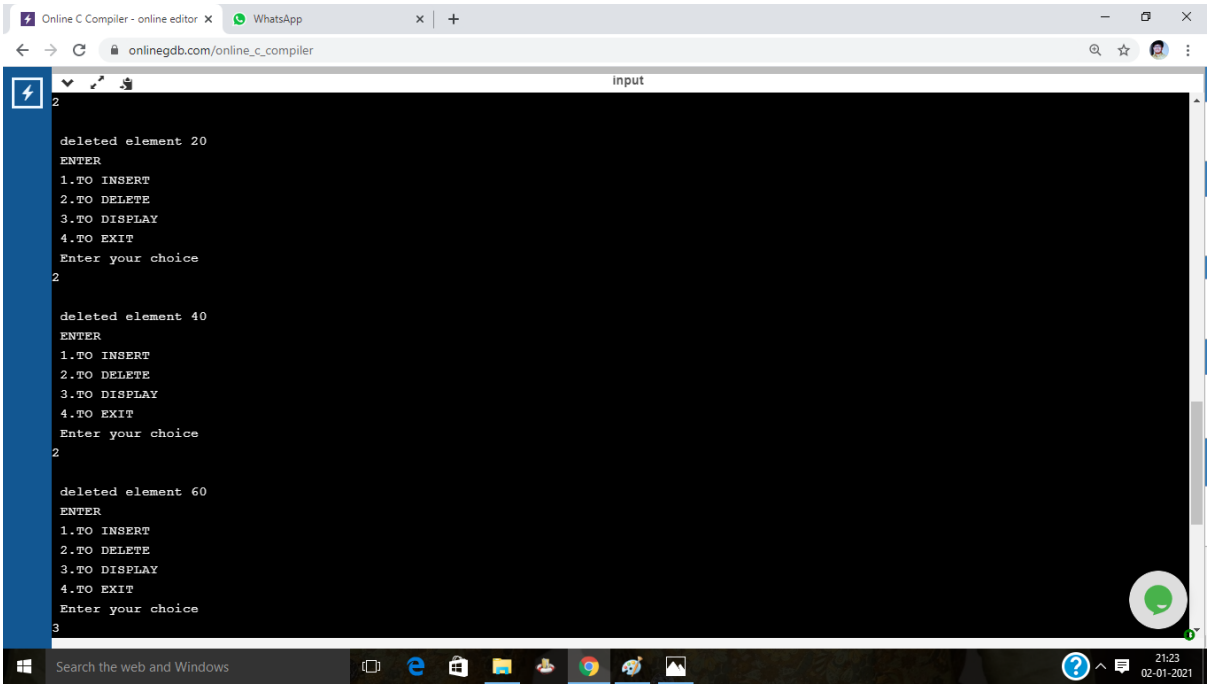
```
int queue[N];
void enq(int a,int *front,int *rear){
if(*rear==N-1)
  printf("\n QUEUE IS FULL");
else {
  if(*front ==-1)
    *front=0;
  (*rear)++;
  queue[*rear]=a;
  printf("\n Insertion Done!");}}
void deq(int *front ,int *rear){
if(*front==*rear)
  printf("\n QUEUE is empty");
else {
  printf("\n deleted element %d",queue[*front]);
  *front=*front+1;
}}
void display(int *front,int *rear){
if(*rear==-1)
  printf("\n QUEUE IS EMPTY!!");
else{int i;
  printf("\n QUEUE elements are :\t");
  for(i=*front ;i<=*rear;i++)
    printf("%d\t",queue[i]);}}
int main(){
int a,choice;
int *front=-1,rear=-1;
while(1){
  printf("\n ENTER \n");
  printf(" 1.TO INSERT");
  printf("\n 2.TO DELETE");
  printf("\n 3.TO DISPLAY");
  printf("\n 4.TO EXIT");
  printf("\n Enter your choice");
  scanf("%d",&choice);
  switch(choice){
case 1:
  printf("Enter the value to inset");
  scanf("%d",& a);
  enq(a,&front,&rear);
  break;
case 2:
  deq(&front,&rear);
  break;
case 3:
  display(&front,&rear);
  break;
case 4:
  return 0;
default:
  printf("\n INVALID ");
  }
}
return 0;
}
```

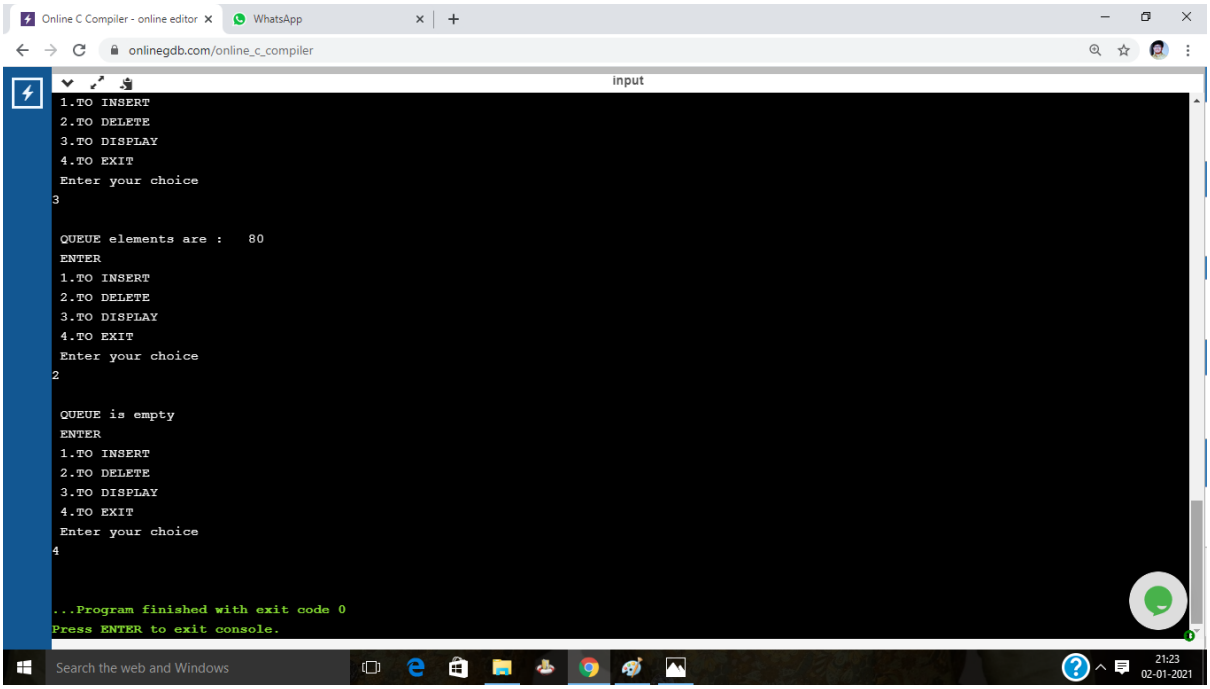

Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
1
Enter the value to insert
20
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
1
Enter the value to insert
40
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Search the web and Windows
21:20
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
1
Enter the value to insert
60
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
1
Enter the value to insert
80
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Search the web and Windows
21:21
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
Insertion Done!
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
3
QUEUE elements are : 20 40 60 80
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
2
deleted element 20
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
2
deleted element 40
Search the web and Windows
21:21
02-01-2021


Online C Compiler - online editor
WhatsApp
onlinegdb.com/online_c_compiler
input
3.TO DISPLAY
4.TO EXIT
Enter your choice
2
deleted element 20
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
2
deleted element 40
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
4.TO EXIT
Enter your choice
2
deleted element 60
ENTER
1.TO INSERT
2.TO DELETE
3.TO DISPLAY
Search the web and Windows
21:22
02-01-2021



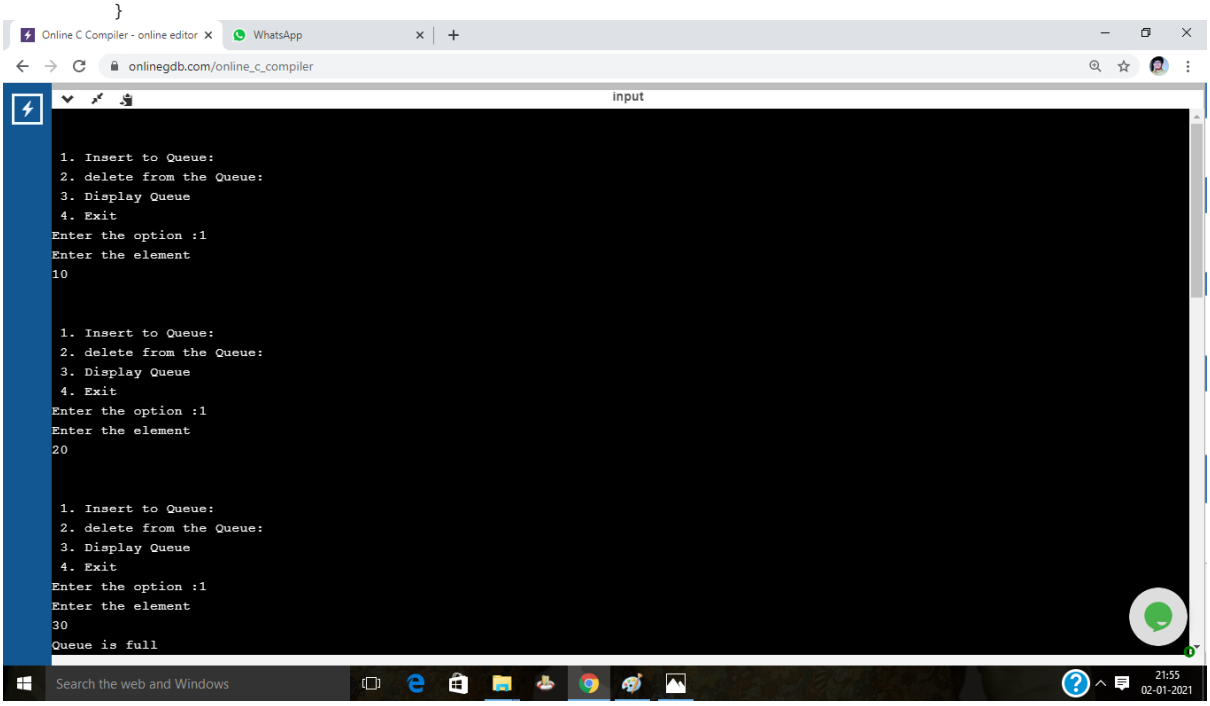


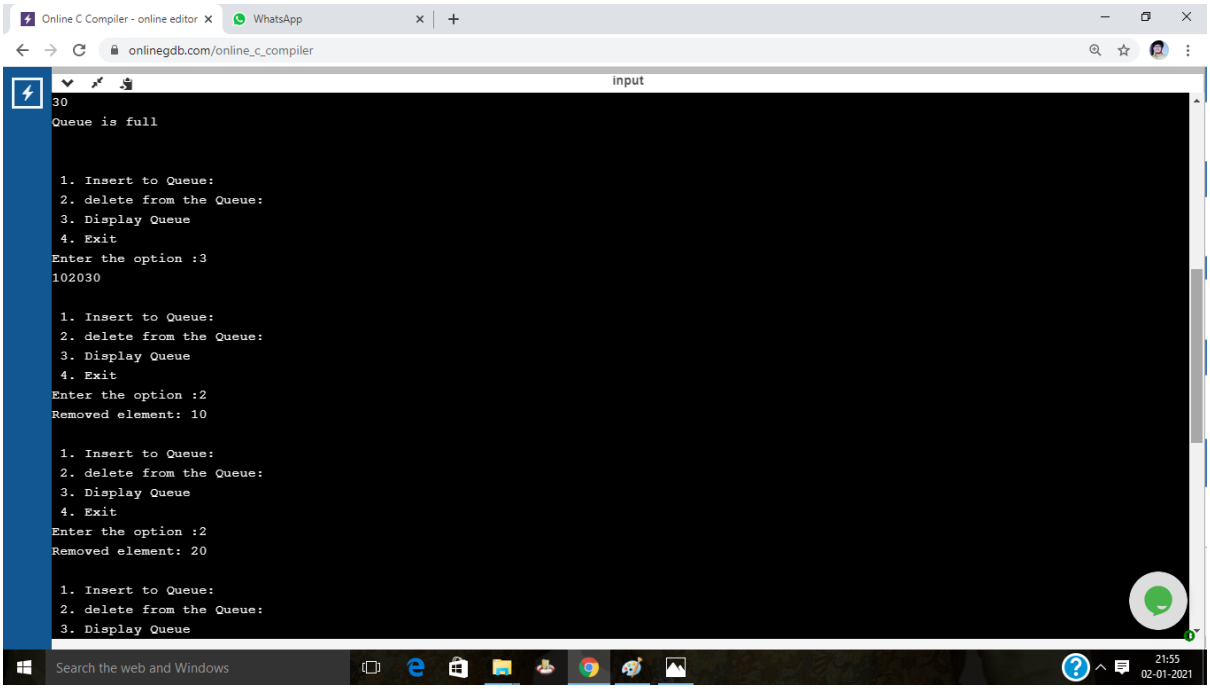
LAB PROGRAM – 4

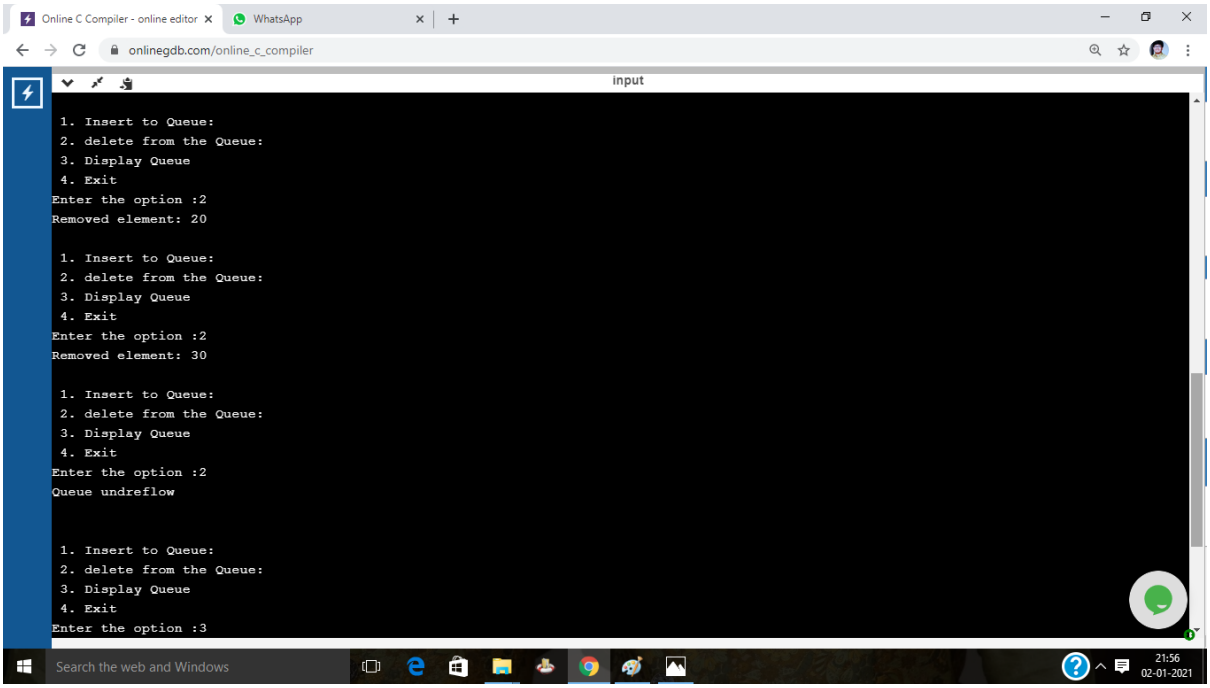
```
#include
<stdio.h>
            #include <stdlib.h>
            #define size 3
            int front=-1;
            int rear=-1;
            void insert(int,int []);
            int delete(int []);
            void display(int []);
```

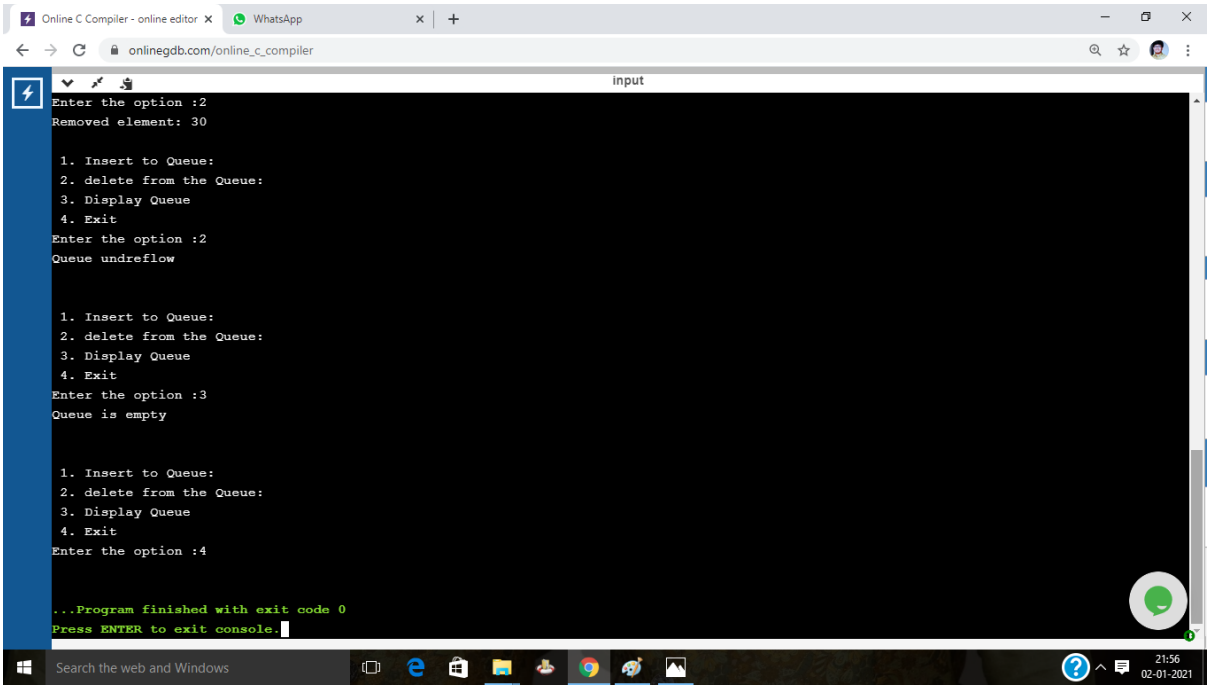
```
int main()
{
    int ch,queue[size];;
    int item;
    do{
        printf("\n\n 1. Insert to Queue: ");
        printf("\n 2. delete from the Queue: ");
        printf("\n 3. Display Queue ");
        printf("\n 4. Exit\n");
        printf("Enter the option :");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:  printf("Enter the element\n");
                     scanf("%d",&item);
                     insert(item,queue);
                     break;
            case 2: item=delete(queue);
                    if(item==-9999)
                      printf("Queue undreflow\n");
                    else
                    printf("Removed element: %d",item);
                    break;
            case 3: display(queue);
                    break;
            case 4: exit(0);
        }
    } while (ch!=4);
          return 0;
}
void insert(int ele,int queue[])
{
    if ((rear+1)%size==front)
       printf("Queue overflow\n");
    else
    {
      rear=(rear+1)%size;
      queue[rear]=ele;
      if ((rear+1)%size==front)
       printf("Queue is full\n");
      if(front ==-1)
          front=0;
    }
}
int delete(int queue[])
{
    int item;
    if(front == -1)
        return -9999;
    else
    {
        item=queue[front];

        if(front==rear)
        {
            front=-1;
            rear=-1;
        }
        else{front=(front+1)%size;}
        return item;
    }

}
void display(int queue[])
```

```
{
    int i;
    if((front==-1)&& (rear==-1))
        printf("Queue is empty\n");
    else
    { if(front<=rear)
       for(i=front;i<=rear;i++)
            printf("%d", queue[i]);
      else
      { for(i=front;i<size;i++)
            printf("%d", queue[i]);
        for(i=0;i<=rear;i++)
            printf("%d", queue[i]);
      }
    }
}
```







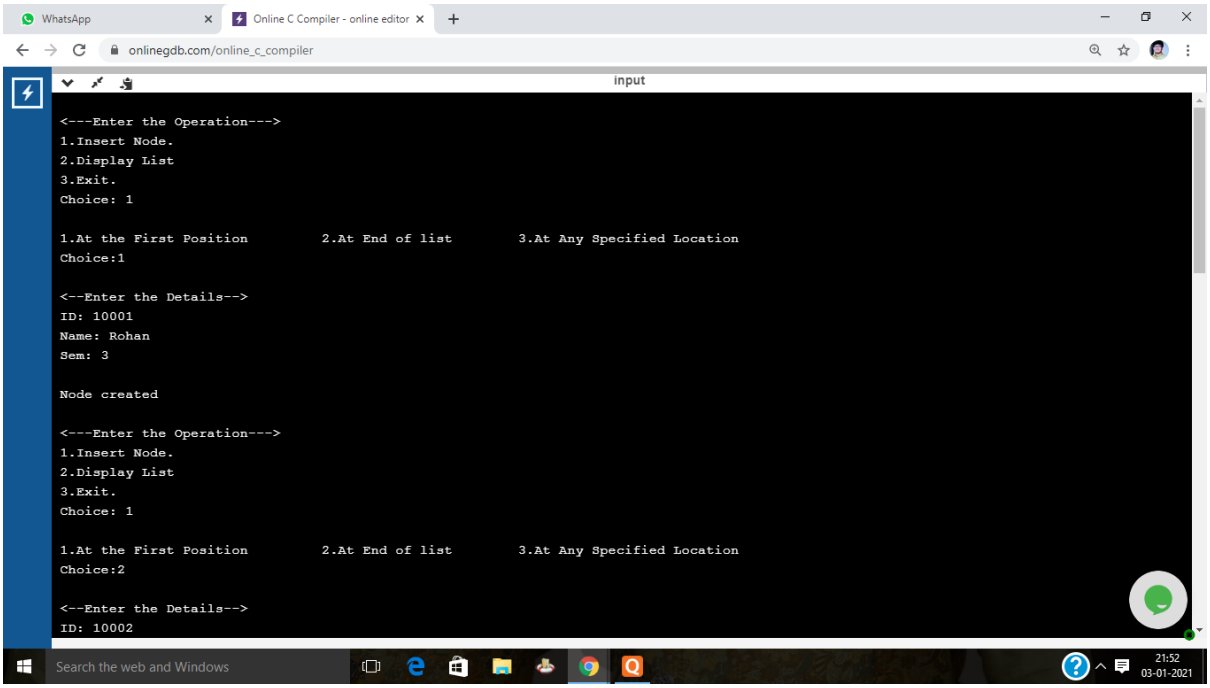


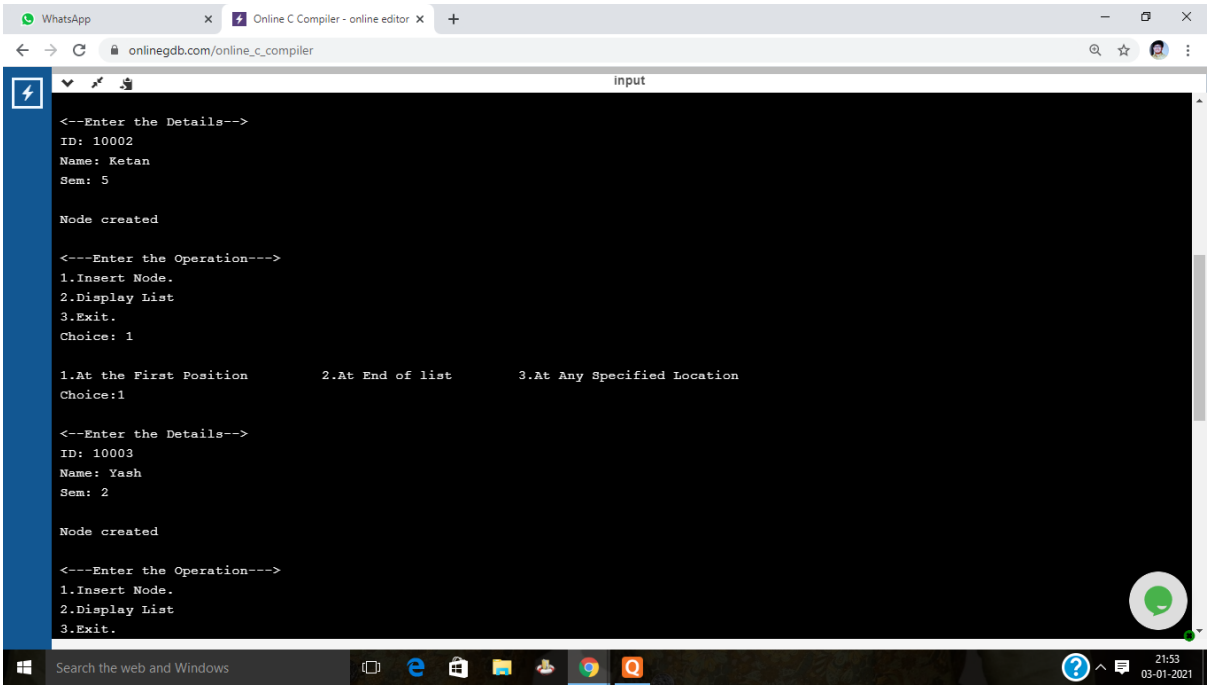
LAB PROGRAM – 5

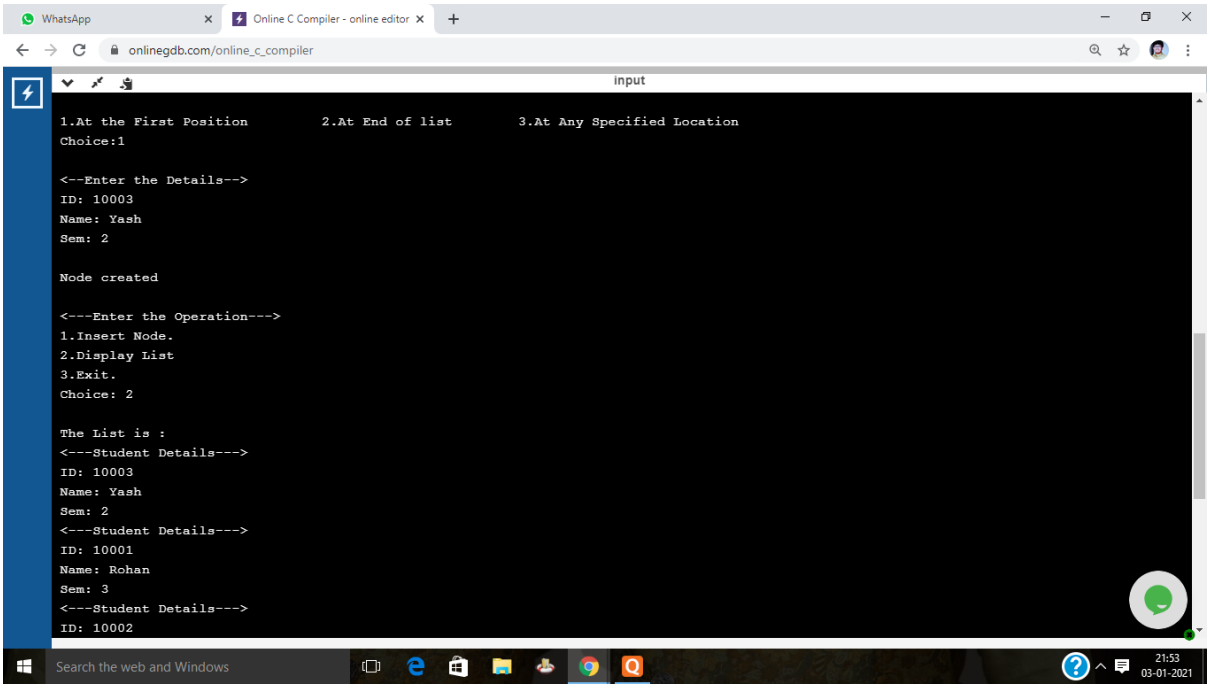
```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                    {
                        int id;
                        char name[20];
                        int sem;
                        struct node *next;
                    };
```

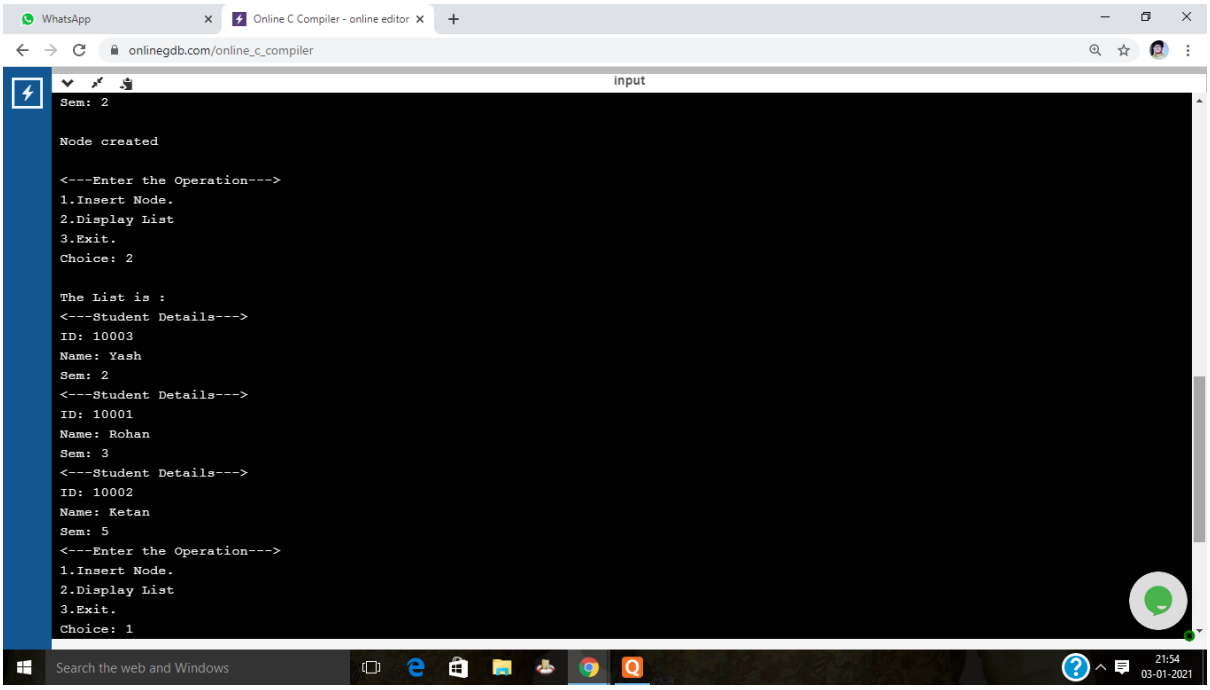
```
struct node *head=NULL;
void linkedList();
void insertNodeAtBegin();
void insertNodeAtEnd();
void insertNodeAtAny();
void displayList();
int size=0;
int main()
{
    linkedList();
    return 0;
}
void linkedList()
{
    int choice1,choice2;
    printf("\n <---Enter the Operation--->");
    printf("\n 1.Insert Node. \n 2.Display List \n 3.Exit.\n Choice: ");
    scanf("%d",&choice1);
    switch(choice1)
    {
        case 1: printf("\n 1.At the First Position \t 2.At End of list\t 3.At Any Specified Location\n
Choice:");
                scanf("%d",&choice2);
                switch(choice2)
                {
                        case 1: insertNodeAtBegin();
                                break;
                        case 2: insertNodeAtEnd();
                                break;
                        case 3: insertNodeAtAny();
                                break;
                        default: printf("\n Input Error, try Again!\n ");
                                 linkedList();
                }
                break;
        case 2: displayList();
                break;
        case 3: exit(0);
        default:printf("\n Input error, Try again!!\n");
                linkedList();
    }
}
void insertNodeAtBegin()
{
    struct node *newnode;
    newnode=(struct node*)malloc(sizeof(struct node));
    printf("\n <--Enter the Details--> ");
    printf("\n ID: ");  scanf("%d",&(newnode->id));
    printf(" Name: ");    scanf("%s",(newnode->name));
    printf(" Sem: ");  scanf("%d",&(newnode->sem));
    newnode->next=head;
    head=newnode;
    size++;
    printf("\n Node created \n");
    linkedList();
}
void insertNodeAtEnd()
{
    struct node *newnode,*temp;
    newnode =(struct node *) malloc (sizeof(struct node));
    printf("\n <--Enter the Details--> ");
    printf("\n ID: ");  scanf("%d",&(newnode->id));
    printf(" Name: ");    scanf("%s",(newnode->name));
    printf(" Sem: ");  scanf("%d",&(newnode->sem));
```

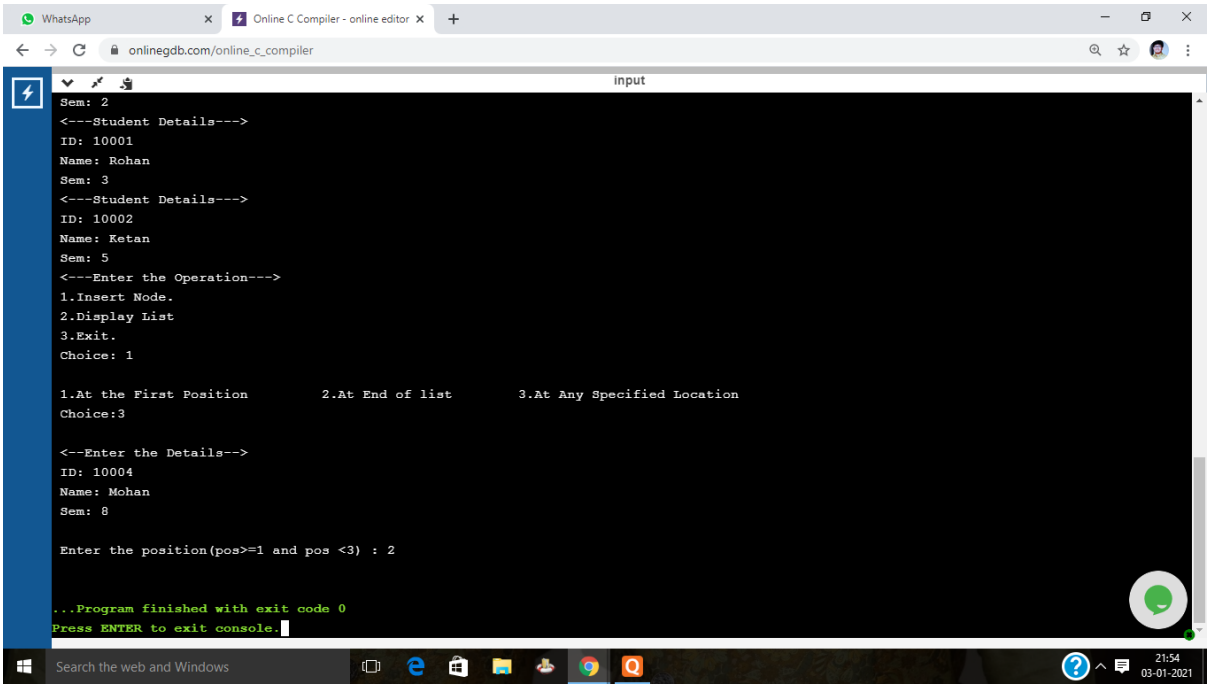
```
    if (head==NULL)
     {
       newnode->next=NULL;
       head=newnode;
       printf("Node created\n");
       linkedList();
     }
    for(temp=head;(temp->next)!=NULL;temp=(temp->next));
    newnode->next=NULL;
    {
            temp->next=newnode;
            size++;
            printf("\n Node created  \n");
            linkedList();
    }
}
void insertNodeAtAny()
{
    struct node *newnode,*temp=head;
    newnode=(struct node*)malloc(sizeof(struct node));
    printf("\n <--Enter the Details--> ");
    printf("\n ID: ");  scanf("%d",&(newnode->id));
    printf(" Name: ");    scanf("%s",(newnode->name));
    printf(" Sem: ");  scanf("%d",&(newnode->sem));
    int pos=0,s=0;
                printf("\n Enter the position(pos>=1 and pos <%d) : ",size);
                scanf("%d",&pos);
                if(pos==0)
                {
                    printf("\n Error position, check the operation menu!");
                    linkedList();
                }
                for(temp;temp->next!=NULL;temp=temp->next)
                {
                    if(s==(pos-1))
                    {
                        newnode->next=(temp->next);
                        temp->next=newnode;
                        size++;
                        printf("\n Node created \n");
                        linkedList();
                    }
                    s++;
                    temp=temp->next;
                }
}
void displayList()
{
    if(head==NULL)
    {
        printf("\n Empty List!\n");
        linkedList();
    }
    printf("\n The List is :");
    for(struct node *temp=head;temp!=NULL;temp=temp->next)
       {
        printf("\n <---Student Details--->");
        printf("\n ID: %d ",temp->id);
        printf("\n Name: %s ",temp->name);
        printf("\n Sem: %d",temp->sem);
       }
       linkedList();
}
```


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 1
1.At the First Position 2.At End of list 3.At Any Specified Location
Choice:1
<--Enter the Details-->
ID: 10001
Name: Rohan
Sem: 3
Node created
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 1
1.At the First Position 2.At End of list 3.At Any Specified Location
Choice:2
<--Enter the Details-->
ID: 10002
Search the web and Windows
21:52
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
<--Enter the Details-->
ID: 10002
Name: Ketan
Sem: 5
Node created
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 1
1.At the First Position 2.At End of list 3.At Any Specified Location
Choice:1
<--Enter the Details-->
ID: 10003
Name: Yash
Sem: 2
Node created
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Search the web and Windows
21:53
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1.At the First Position 2.At End of list 3.At Any Specified Location
Choice:1
<--Enter the Details-->
ID: 10003
Name: Yash
Sem: 2
Node created
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 2
The List is :
<---Student Details--->
ID: 10003
Name: Yash
Sem: 2
<---Student Details--->
ID: 10001
Name: Rohan
Sem: 3
<---Student Details--->
ID: 10002
Search the web and Windows
21:53
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
Sem: 2
Node created
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 2
The List is :
<---Student Details--->
ID: 10003
Name: Yash
Sem: 2
<---Student Details--->
ID: 10001
Name: Rohan
Sem: 3
<---Student Details--->
ID: 10002
Name: Ketan
Sem: 5
<---Enter the Operation--->
1.Insert Node.
2.Display List
3.Exit.
Choice: 1
Search the web and Windows
21:54
03-01-2021



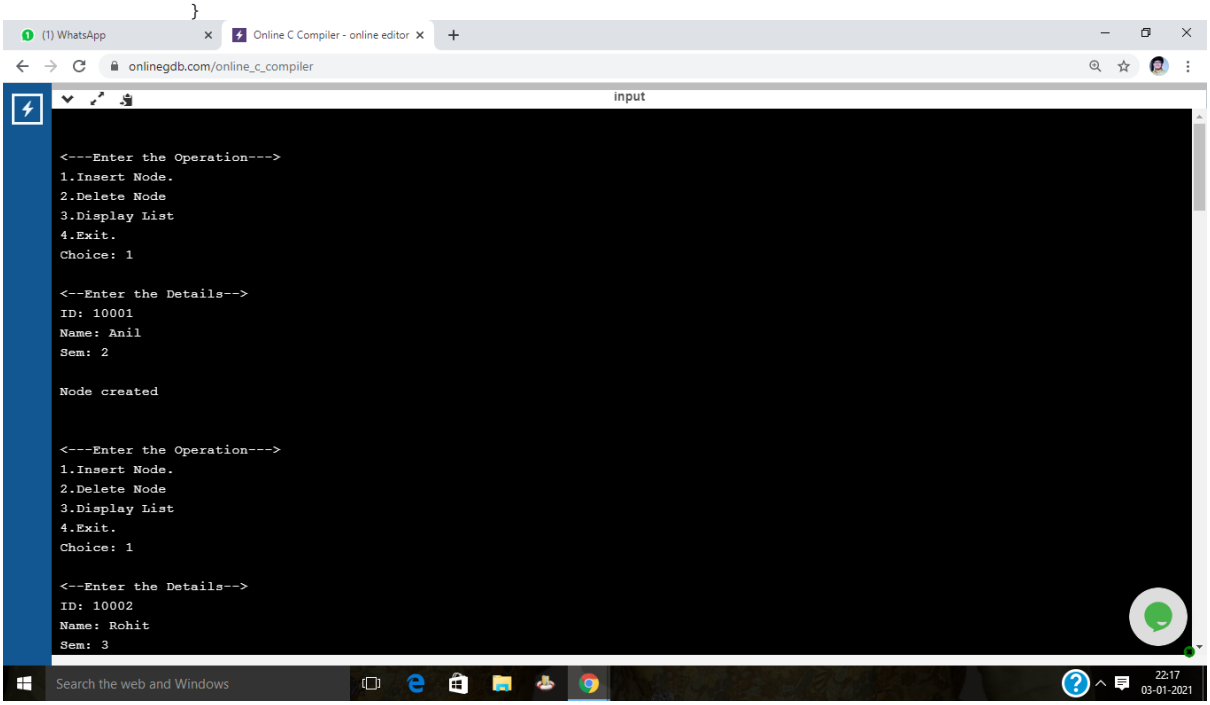
LAB PROGRAM – 6

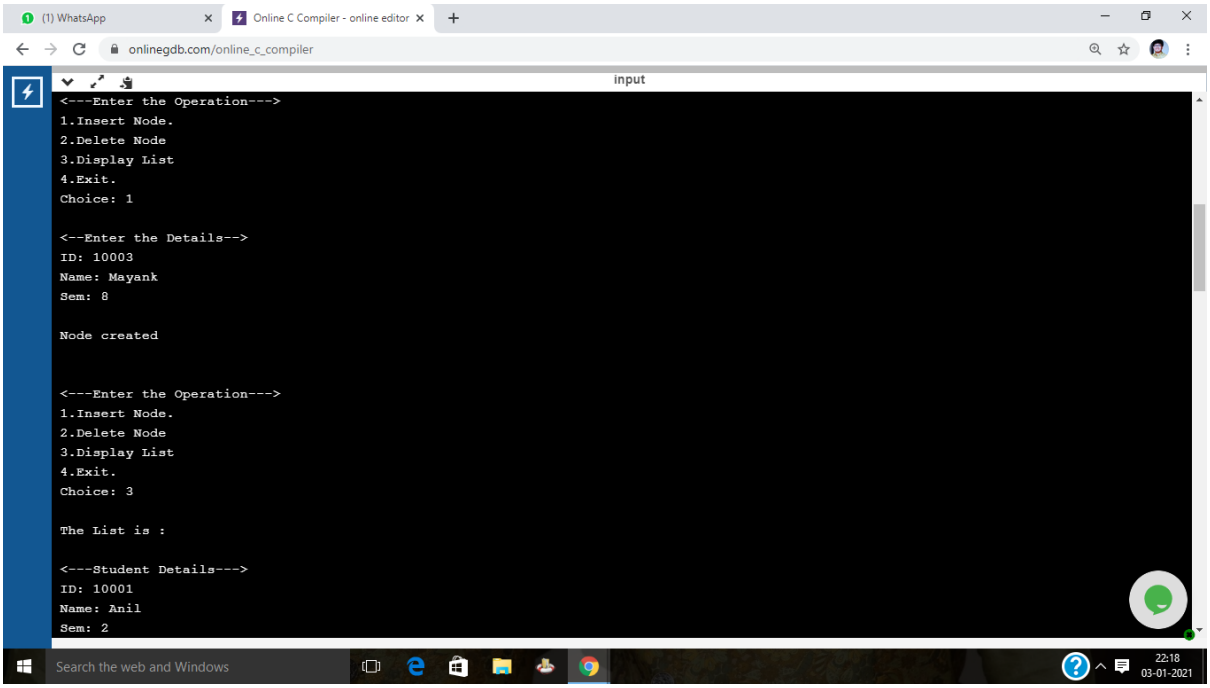
```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                    {
                        int id;
                        char name[20];
                        int sem;
                        struct node *next;
                    };
                    struct node *head=NULL;
                    void linkedList();
                    void insertNode();
                    void deleteNode(int);
                    void deleteNodeAtBegin();
                    void deleteNodeAtEnd();
                    void deleteNodeOFGiven();
                    void displayList();
                    int size=0;
                    int main()
                    {
                        linkedList();
                        return 0;
                    }
                    void linkedList()
                    {
                        int choice1,choice2;
                        printf("\n\n <---Enter the Operation--->");
                        printf("\n 1.Insert Node. \n 2.Delete Node \n 3.Display List\n 4.Exit.\n Choice: ");
                        scanf("%d",&choice1);
                        switch(choice1)
                        {
                            case 1:
                                    insertNode();
```

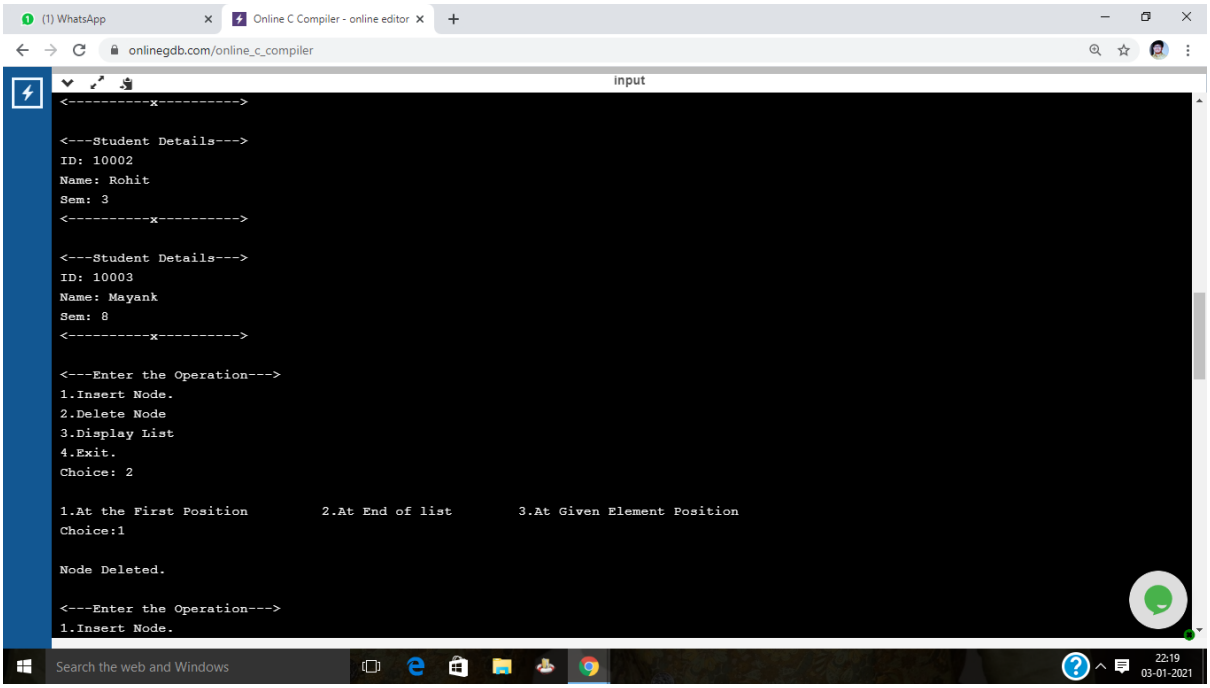
```
                break;
        case 2:
                printf("\n 1.At the First Position \t 2.At End of list\t 3.At Given Element Position\n
Choice:");
                scanf("%d",&choice2);
                switch(choice2)
                {
                        case 1: deleteNode(1);
                                break;
                        case 2: deleteNode(2);
                                break;
                        case 3: deleteNode(3);
                                break;
                        default: printf("\n Input Error, try Again!\n ");
                                 linkedList();
                }
                break;
        case 3: displayList();
                break;
        case 4: exit(0);
        default: printf("\n Input error, Try again!!\n");
                 linkedList();
    }
}
void insertNode()
{
    struct node *newnode,*temp;
    newnode =(struct node *) malloc (sizeof(struct node));
    printf("\n <--Enter the Details--> ");
    printf("\n ID: ");  scanf("%d",&(newnode->id));
    printf(" Name: ");    scanf("%s",(newnode->name));
    printf(" Sem: ");  scanf("%d",&(newnode->sem));
    if (head==NULL)
     {
       newnode->next=NULL;
       head=newnode;
       printf("\n Node created\n");
       linkedList();
       size++;
     }
    for(temp=head;(temp->next)!=NULL;temp=(temp->next));
    newnode->next=NULL;
    {
            temp->next=newnode;
            size++;
            printf("\n Node created  \n");
            linkedList();
    }
}
void deleteNode(int flag)
{
    if(head==NULL)
    {
        printf("\n The List is Empty! \n");
        linkedList();
    }
    else
    if(head->next==NULL)
    {
         printf("\n Node Deleted. \n Now List Is empty! ");
         free(head);
         head=NULL;
        linkedList();
    }
```

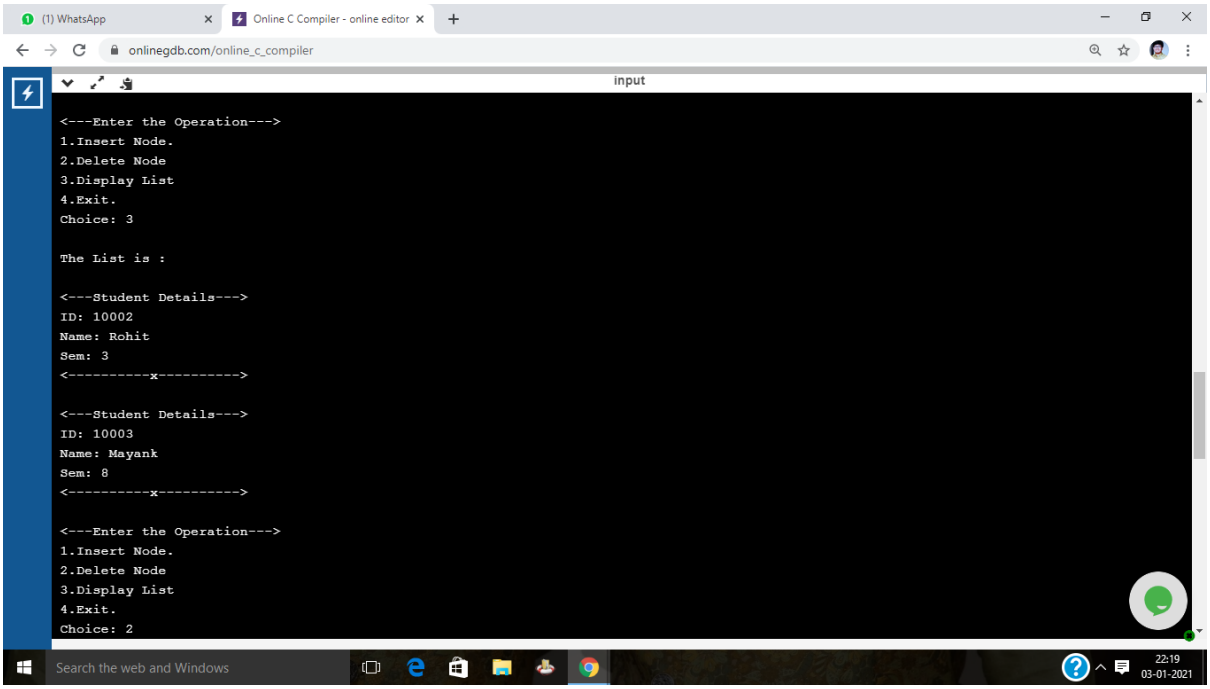
```
    else
    {
        switch(flag)
        {
            case 1: deleteNodeAtBegin();
                    break;
            case 2: deleteNodeAtEnd();
                    break;
            case 3: deleteNodeOFGiven();
                    break;
        }
        linkedList();
    }
}
void deleteNodeAtBegin()
{
    struct node* temp=head;
    head=head->next;
    free(temp);
    printf("\n Node Deleted.");
}
void deleteNodeAtEnd()
{
    struct node* temp=head;
    for(temp;(temp->next)->next!=NULL;temp=temp->next);
    free(temp->next);
    temp->next=NULL;
    printf("\n Node Deleted.");
}
void deleteNodeOFGiven()
{
     struct node* temp1=head,*temp2=temp1;
     int ele;
     printf("\n Enter the Element :");
     scanf("%d",&ele);
     for(temp1;temp1->next!=NULL;temp1=temp1->next)
     {
         temp2=temp1->next;
         if(temp2->id==ele)
         {
             temp1->next=temp2->next;
             free(temp2);
             printf("\n Node Deleted.");
             return;
         }
         if(temp1->id==ele)
         {
             head=temp2;
             free(temp1);
             printf("\n Node Deleted.");
             return;
         }
     }
     printf("\n Element Not Present In List! ");
     linkedList();
}
void displayList()
{
   if(head==NULL)
    {
        printf("\n Empty List!\n");
        linkedList();
    }
    printf("\n The List is :");
```

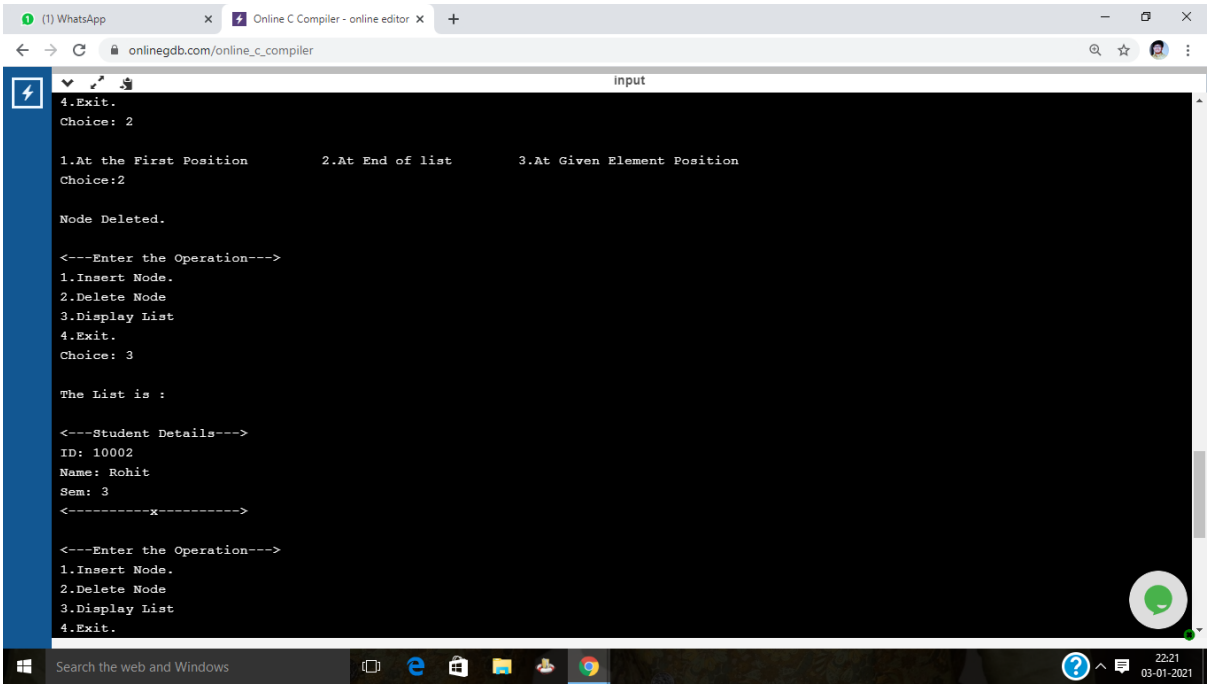
```
        for(struct node *temp=head;temp!=NULL;temp=temp->next)
           {
            printf("\n\n <---Student Details--->");
            printf("\n ID: %d ",temp->id);
            printf("\n Name: %s ",temp->name);
            printf("\n Sem: %d",temp->sem);
             printf("\n <----------x---------->");
           }
           linkedList();
    }
```

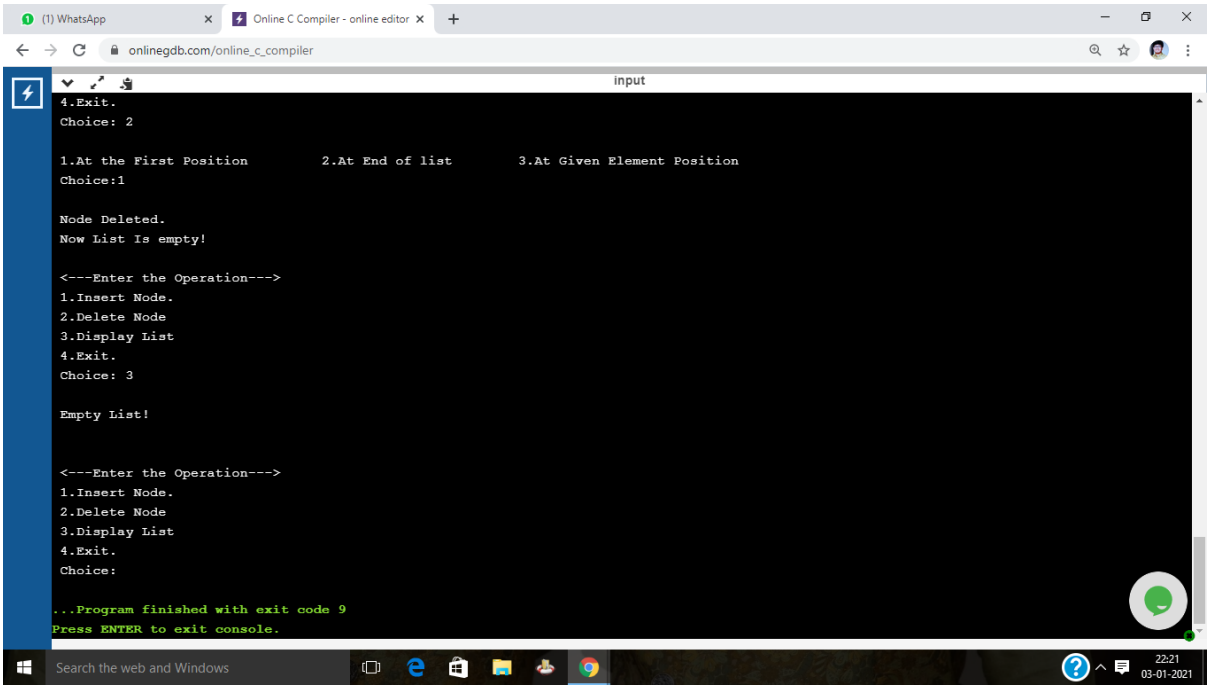





(1) WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
<----------x---------->
<---Student Details--->
ID: 10002
Name: Rohit
Sem: 3
<----------x---------->
<---Student Details--->
ID: 10003
Name: Mayank
Sem: 8
<----------x---------->
<---Enter the Operation--->
1.Insert Node.
2.Delete Node
3.Display List
4.Exit.
Choice: 2
1.At the First Position 2.At End of list 3.At Given Element Position
Choice:1
Node Deleted.
<---Enter the Operation--->
1.Insert Node.
Search the web and Windows
22:19
03-01-2021


(1) WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
<---Enter the Operation--->
1.Insert Node.
2.Delete Node
3.Display List
4.Exit.
Choice: 3
The List is :
<---Student Details--->
ID: 10002
Name: Rohit
Sem: 3
<----------x---------->
<---Student Details--->
ID: 10003
Name: Mayank
Sem: 8
<----------x---------->
<---Enter the Operation--->
1.Insert Node.
2.Delete Node
3.Display List
4.Exit.
Choice: 2
Search the web and Windows
22:19
03-01-2021





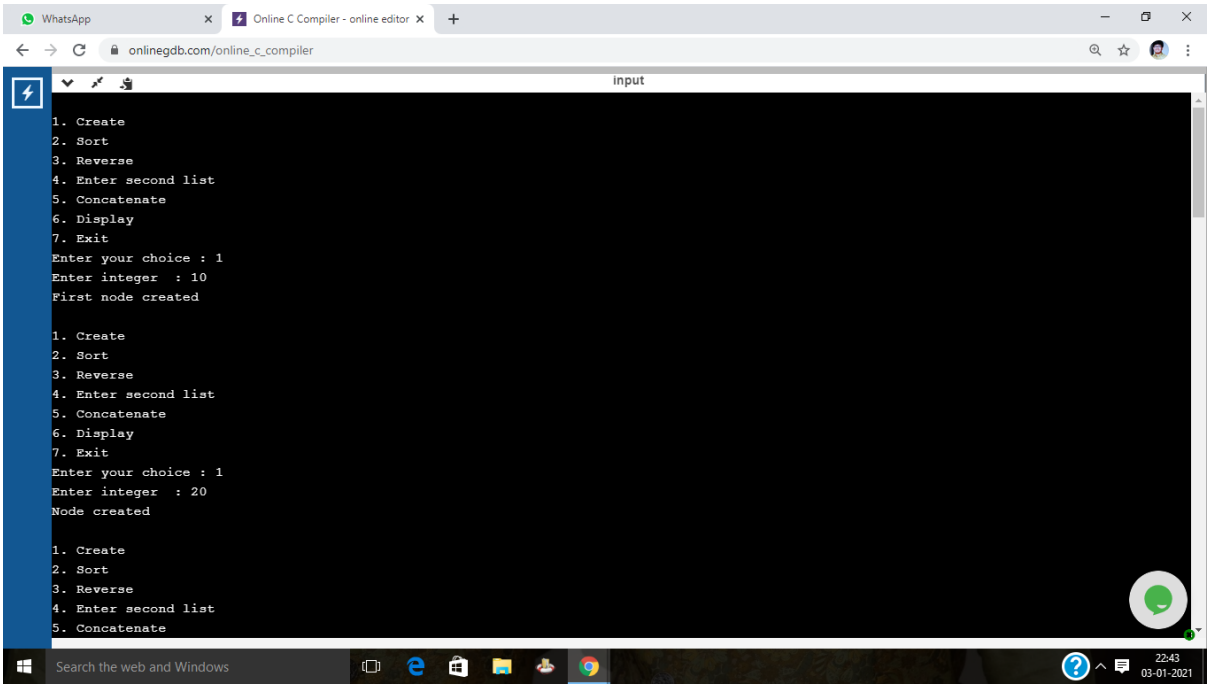
LAB PROGRAM – 7

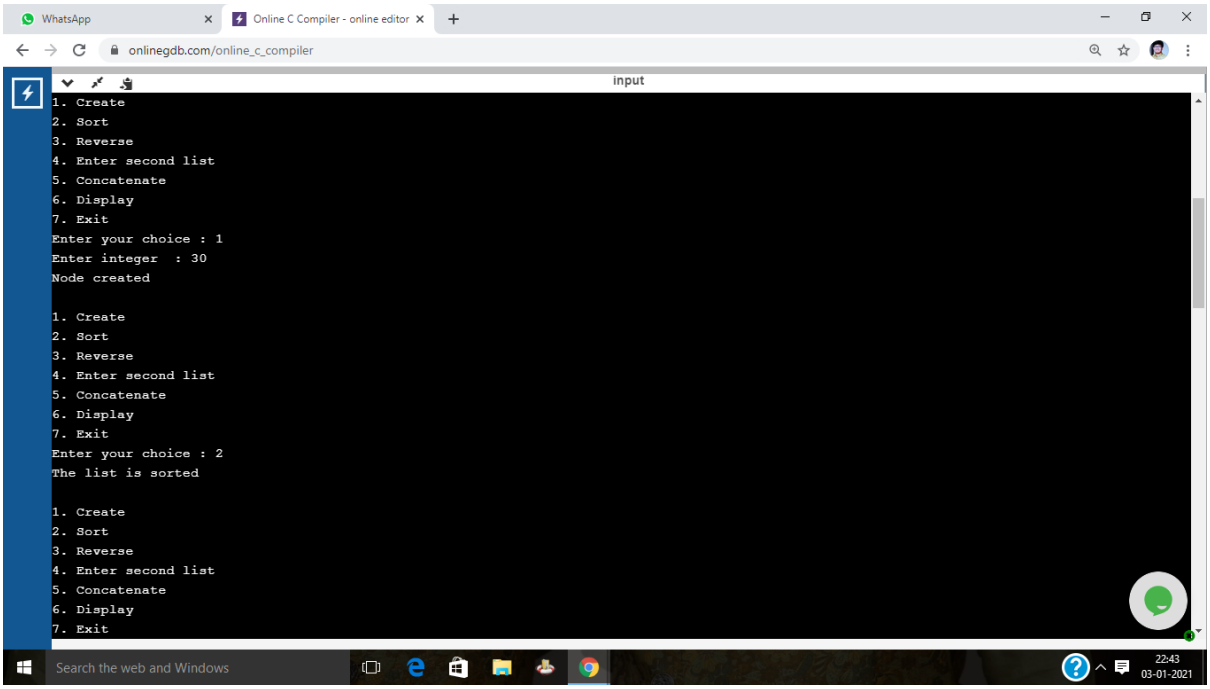
```
#include
<stdio.h>
            #include <stdlib.h>
            void sort();
            void create();
            void reverse();
            void create_second();
            void concatenate();
            void display();
```

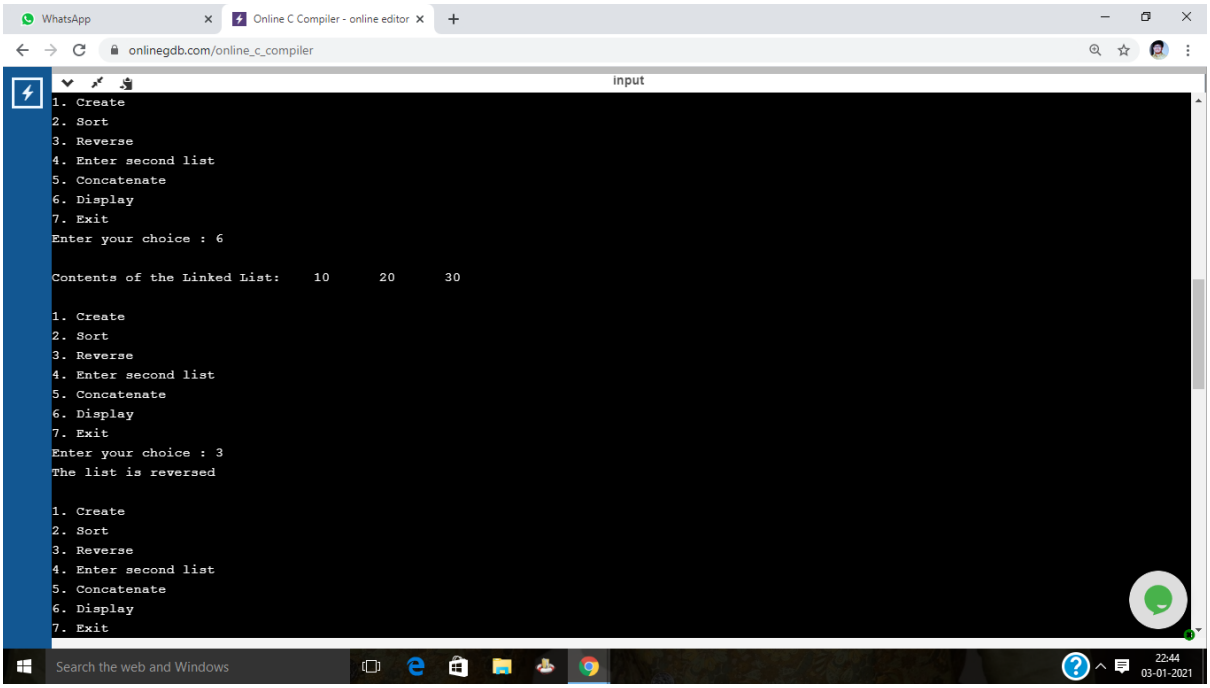
```
struct node
{
  int data;
  struct node *next;
};
struct node *head=NULL;
struct node *head2= NULL;
int c;
int main()
{
          int choice;
    do
    {
    printf("\n1. Create\n2. Sort\n3. Reverse\n4. Enter second list\n5. Concatenate\n6. Display\n7. Exit");
    printf("\nEnter your choice : ");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1: create();
                break;
        case 2: sort();
                break;
        case 3: reverse();
                break;
        case 4: create_second();
                break;
        case 5: concatenate();
                break;
        case 6: display();
                break;
        case 7: exit(0);
    }
    }while(choice != 7);
    return 0;
}
void create()
{
    struct node *newnode;
    struct node *temp;
    int s;
    printf("Enter integer  : ");
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data =s;
    if (head==NULL)
    {
      newnode->next=NULL;
      head=newnode;
      printf("First node created\n");
      c++;
    }
     else
     {
        temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
    temp->next=newnode;
    newnode->next=NULL;
    c++;
    printf("Node created\n");
   }
}
```

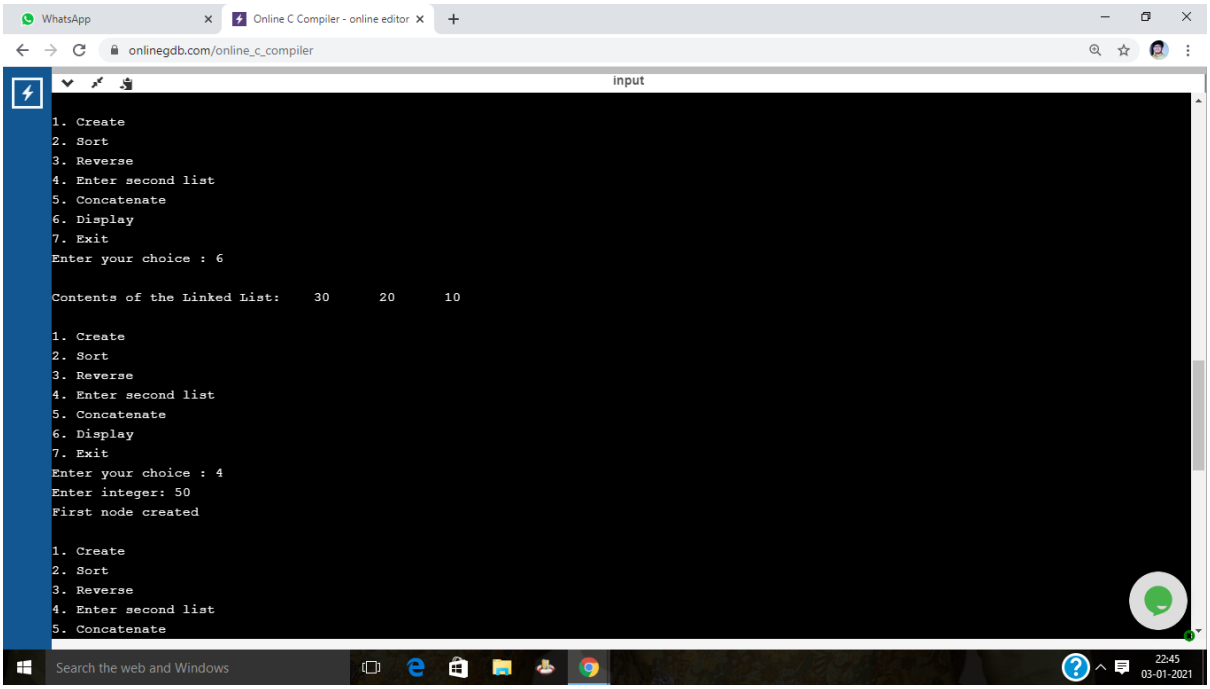
```
void reverse()
{
    struct node *prev=NULL,*current=head, *next=NULL;
    while(current!=NULL)
    {
        next=current->next;
        current->next=prev;
        prev=current;
        current=next;
    }
    head=prev;
    printf("The list is reversed\n");
}
void display()
{
    struct node *ptr=NULL;
    ptr=head;

    if(ptr==NULL)
    {
        printf("List is empty\n");
    }
    else
    {
        printf("\nContents of the Linked List: ");
        while(ptr!=NULL)
       {
          printf("\t%d",ptr->data);
          ptr=ptr->next;
       }
    }
    printf("\n");
}
void create_second()
{
  struct node *newnode;
  struct node *temp;
    int s,y;
    printf("Enter integer: ");
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data =s;
    if (head2==NULL)
    {
      newnode->next=NULL;
      head2=newnode;
      printf("First node created\n");
      c++;
    }
     else
     {
    temp=head2;
        while(temp->next!=NULL)
        {
      temp=temp->next;
        }
    temp->next=newnode;
    newnode->next=NULL;
    c++;
    printf("Node created\n");
   }
}
void concatenate()
{
```

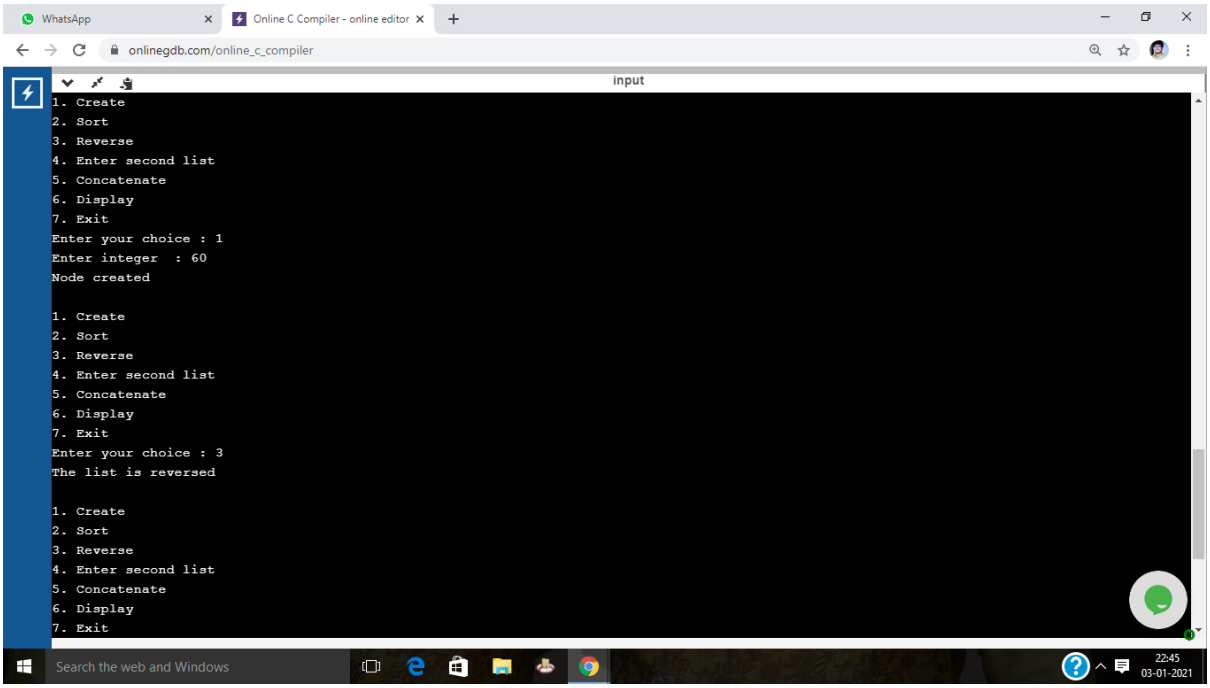
```
    struct node *ptr;
        if(head==NULL)
        {
                head=head2;
        }
        if(head2==NULL)
        {
            head2=head;
        }
        ptr=head;
        while(ptr->next!=NULL)
            ptr=ptr->next;
        ptr->next=head2;
        printf("The list is concatenated\n");
}
void sort()
{
    int swap, i;
    struct node *ptr1;
    struct node *lptr = NULL;

    if (head == NULL)
        return;

    do
    {
        swap = 0;
        ptr1 = head;

        while (ptr1->next != lptr)
        {
            if (ptr1->data > ptr1->next->data)
            {
                int temp = ptr1->data;
                ptr1->data = ptr1->next->data;
                ptr1->next->data = temp;
                swap = 1;
            }
            ptr1 = ptr1->next;
        }
        lptr = ptr1;
    }
    while(swap);
    printf("The list is sorted\n");
}
```

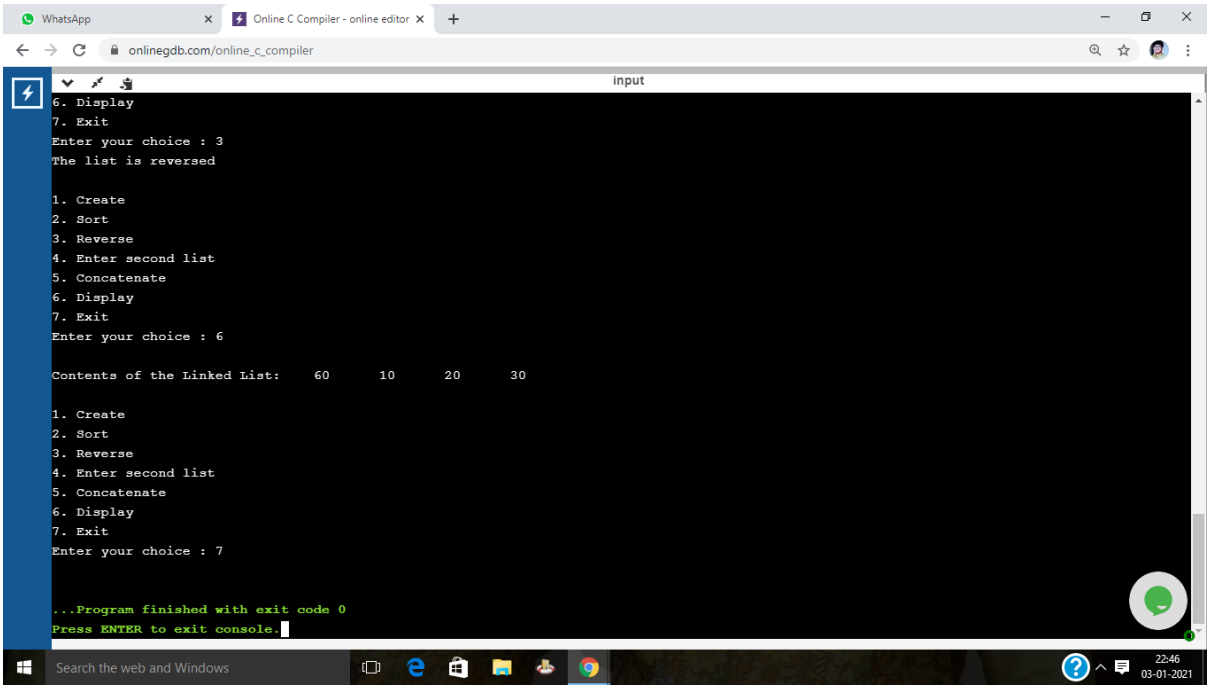

WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 1
Enter integer : 10
First node created
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 1
Enter integer : 20
Node created
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
Search the web and Windows
22:43
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 1
Enter integer : 30
Node created
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 2
The list is sorted
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Search the web and Windows
22:43
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 6
Contents of the Linked List: 10 20 30
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 3
The list is reversed
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Search the web and Windows
22:44
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 6
Contents of the Linked List: 30 20 10
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
6. Display
7. Exit
Enter your choice : 4
Enter integer: 50
First node created
1. Create
2. Sort
3. Reverse
4. Enter second list
5. Concatenate
Search the web and Windows
22:45
03-01-2021





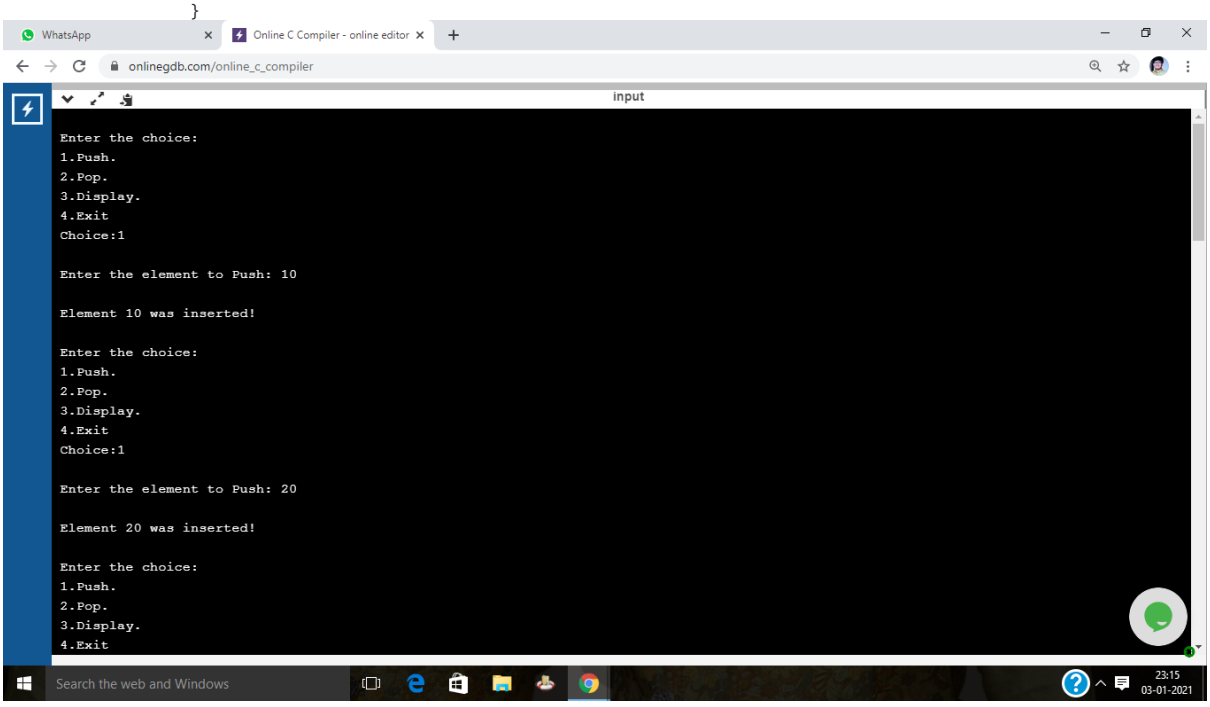
LAB PROGRAM – 8

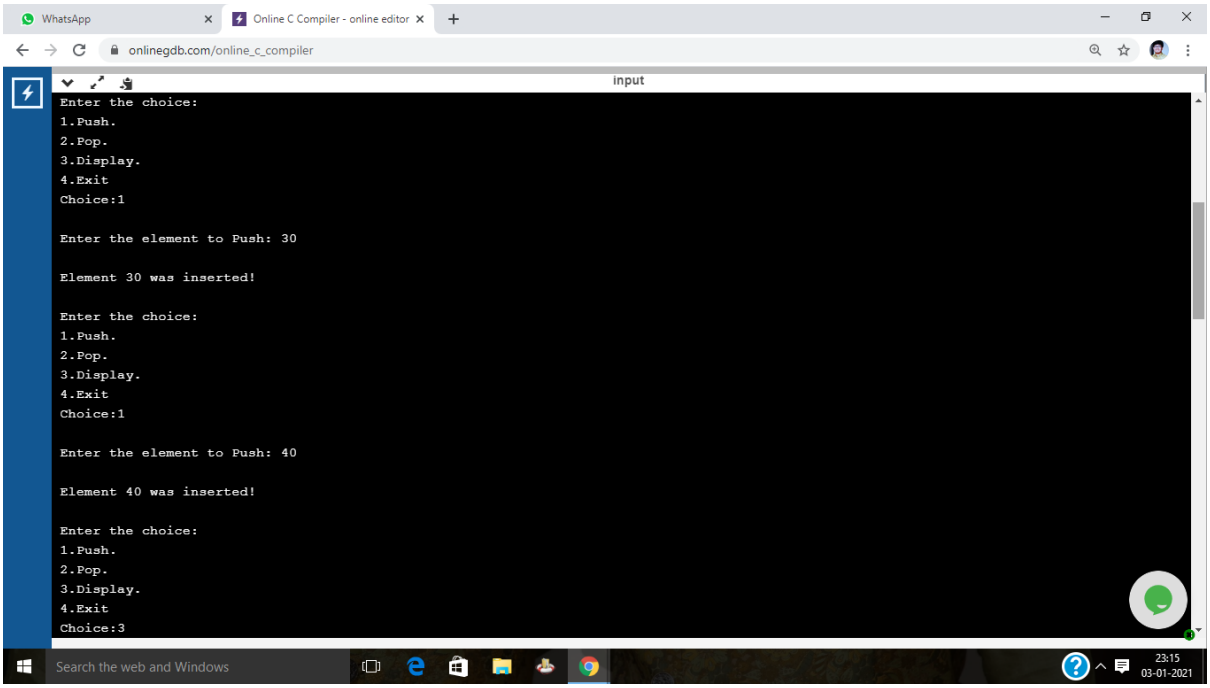
STACK :

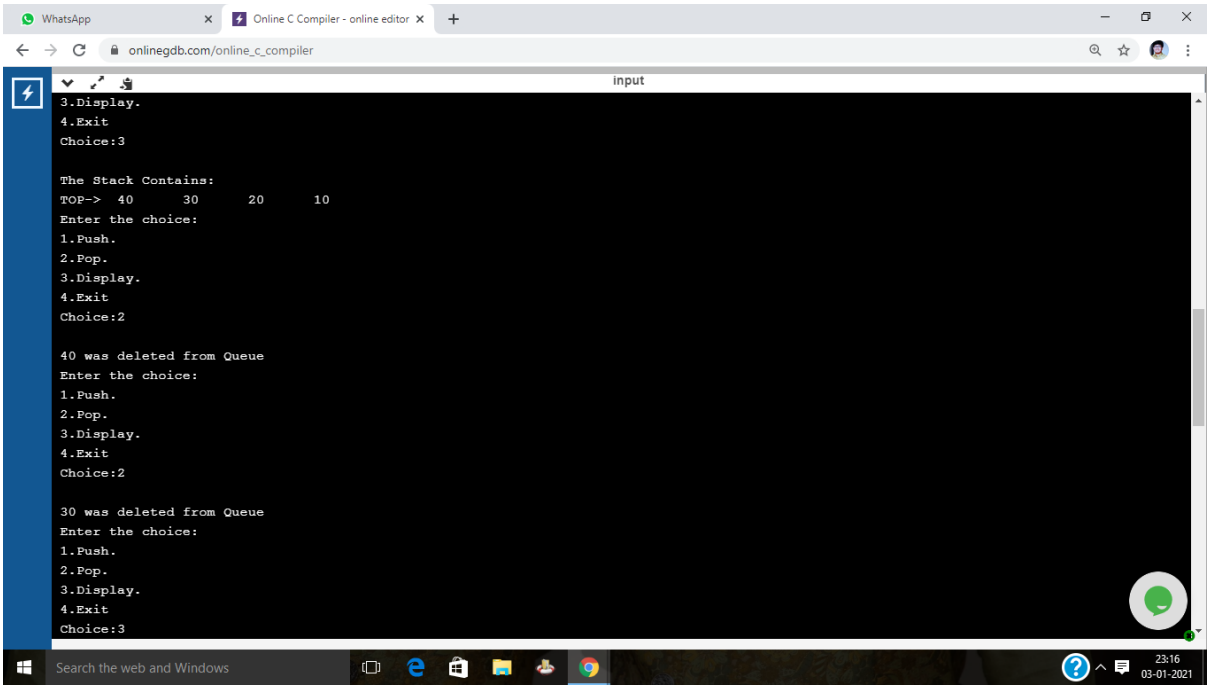
```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node *top=NULL;
                    struct node
                    {
                        int data;
                        struct node *next;
```

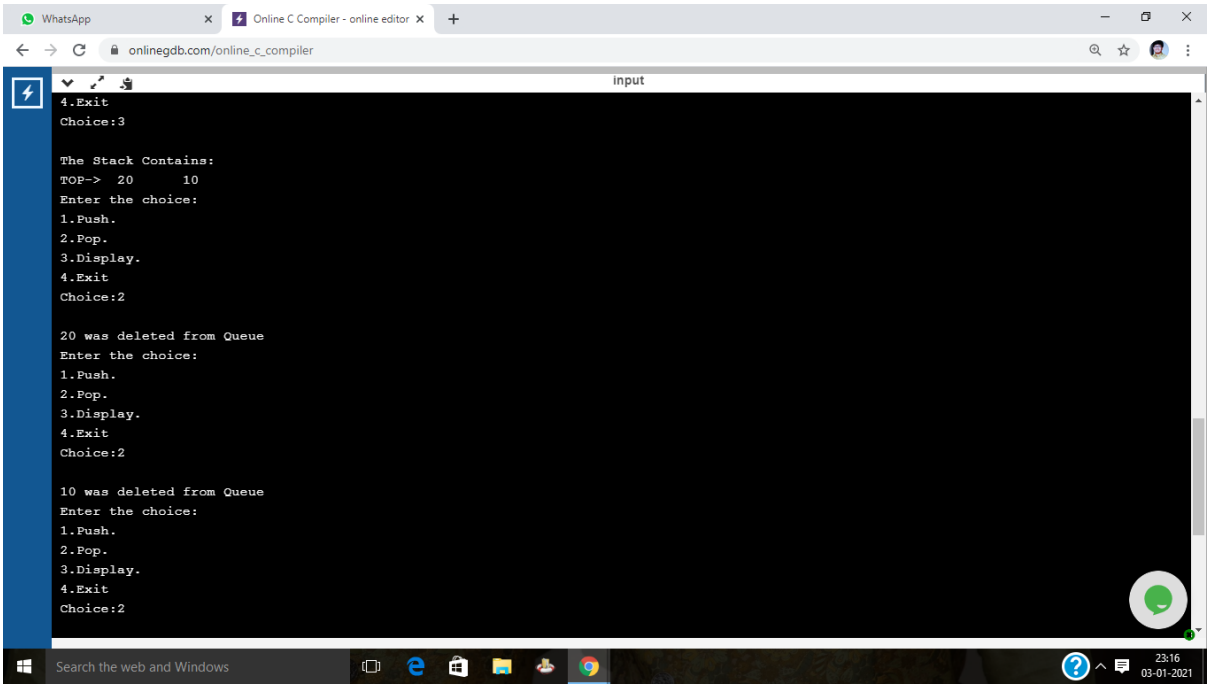
```
};
void stack();
void push(int);
int pop();
void display();
int main()
{
    stack();
    return 1;
}
void stack()
{
    int choice=0,ele=0;
    do
    {
        printf("\n Enter the choice:\n 1.Push.\n 2.Pop.\n 3.Display.\n 4.Exit\n Choice:");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
                printf("\n Enter the element to Push: ");
                scanf("%d",&ele);
                push(ele);
                break;
        case 2:
                ele=pop();
                printf("\n %d was deleted from Queue ",ele);
                break;
        case 3:
                display();
                break;
        case 4:
                exit(0);
        default:
                printf("\n Input Error Try Again! ");
                stack();
        }
    }while(1);
}
void push(int ele)
{
    struct node *newnode;
    newnode =(struct node *) malloc (sizeof(struct node));
    newnode->data=ele;
    newnode->next=NULL;
    if(top==NULL)
    {
        top=newnode;
    }
    else
    {
        newnode->next=top;
        top=newnode;
    }
    printf("\n Element %d was inserted!\n ",ele);
}
int pop()
{
     int ele;
     struct node *temp;
     if(top==NULL)
     {
         printf("\n Stack UnderFlow, The Stack is empty! \n");
         stack();
```

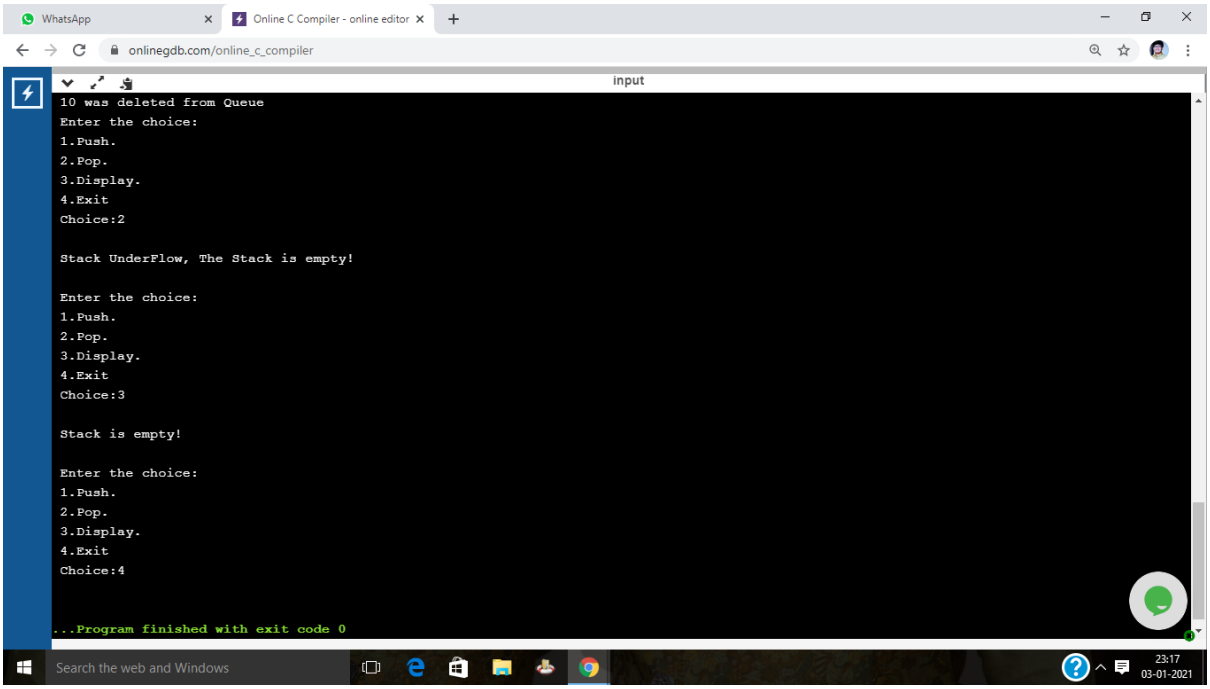
```
        }
        ele=top->data;
        temp=top;
        if(top->next==NULL)
        {
            top=NULL;
        }
        else
        {
           top=top->next;
        }
        free(temp);
        return ele;
}
void display()
{
    struct node *i;
    if(top==NULL)
     {
         printf("\n Stack is empty!\n ");
         stack();
     }
    printf("\n The Stack Contains:\n TOP->");
    for(i=top;i!=NULL;i=i->next)
       printf("\t%d ",i->data);
}
```




WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:1
Enter the element to Push: 30
Element 30 was inserted!
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:1
Enter the element to Push: 40
Element 40 was inserted!
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:3
Search the web and Windows
23:15
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
3.Display.
4.Exit
Choice:3
The Stack Contains:
TOP-> 40 30 20 10
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:2
40 was deleted from Queue
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:2
30 was deleted from Queue
Enter the choice:
1.Push.
2.Pop.
3.Display.
4.Exit
Choice:3
Search the web and Windows
23:16
03-01-2021



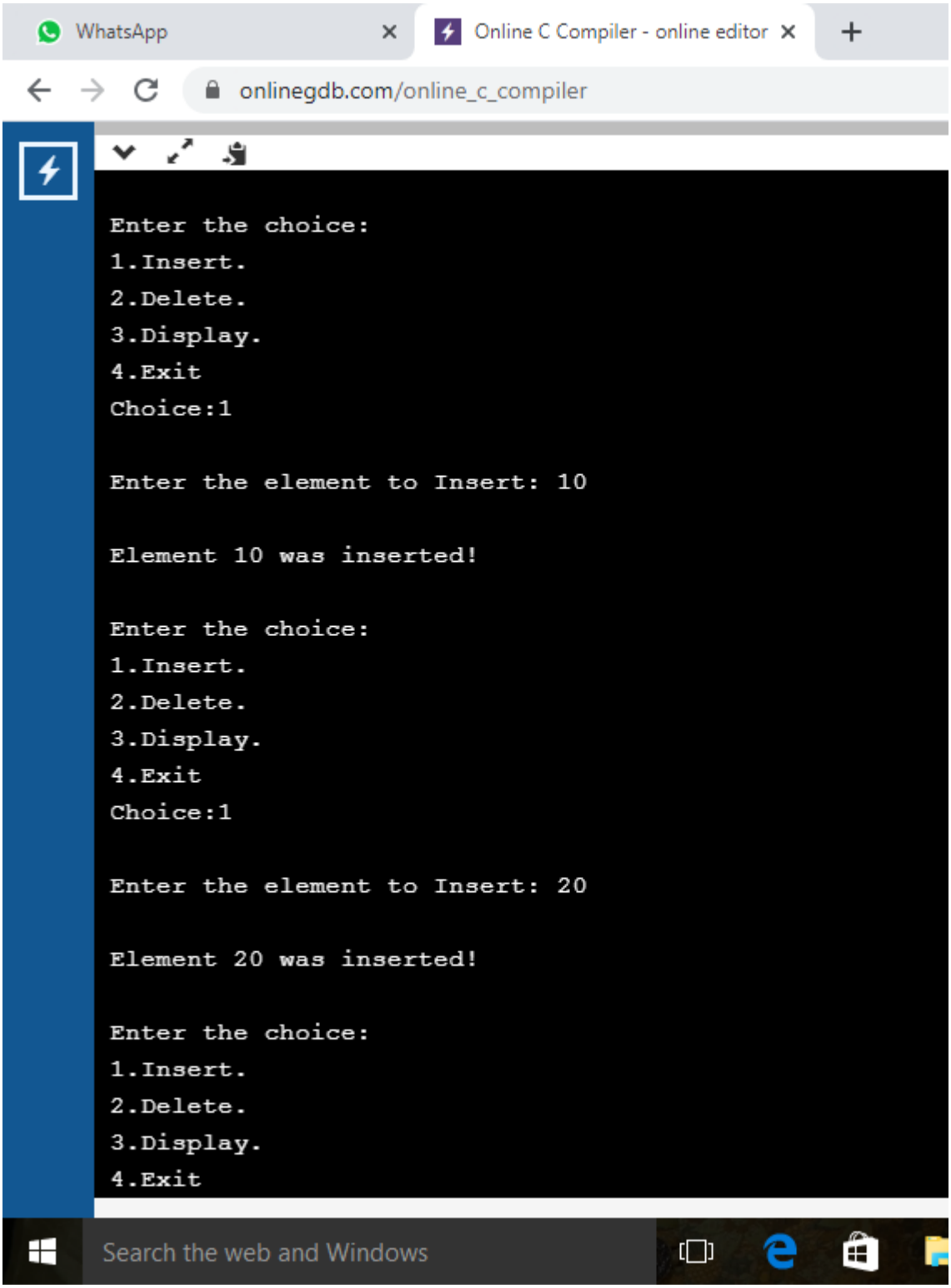


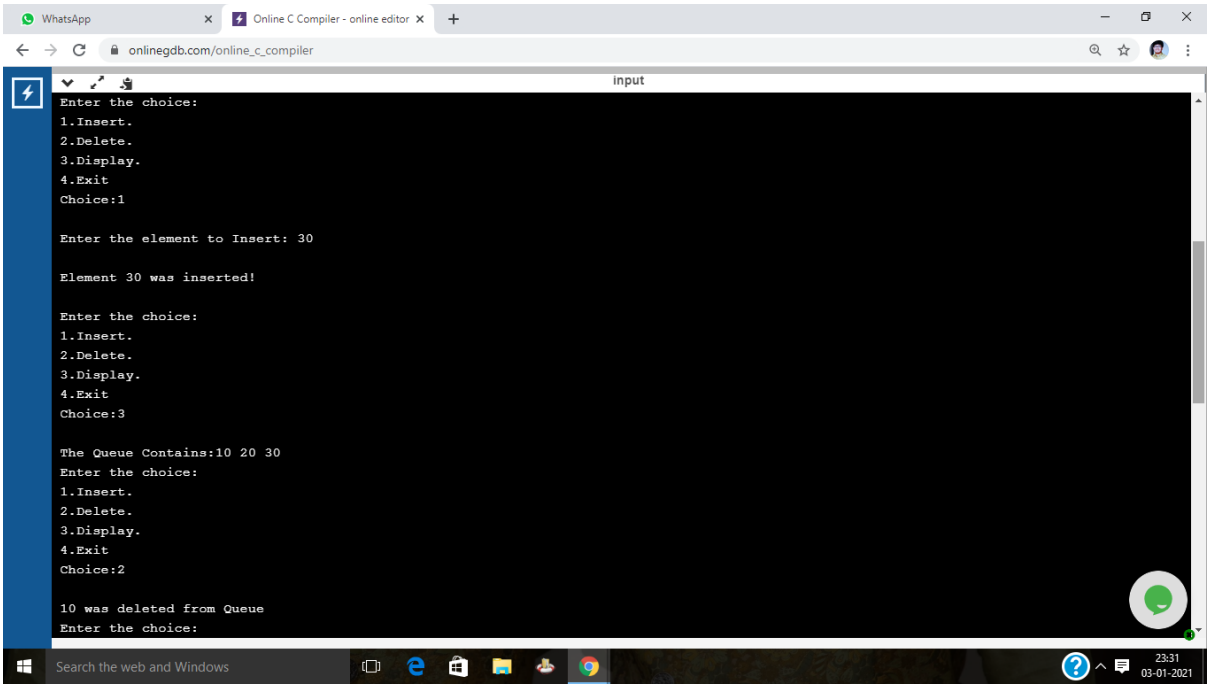
QUEUE :

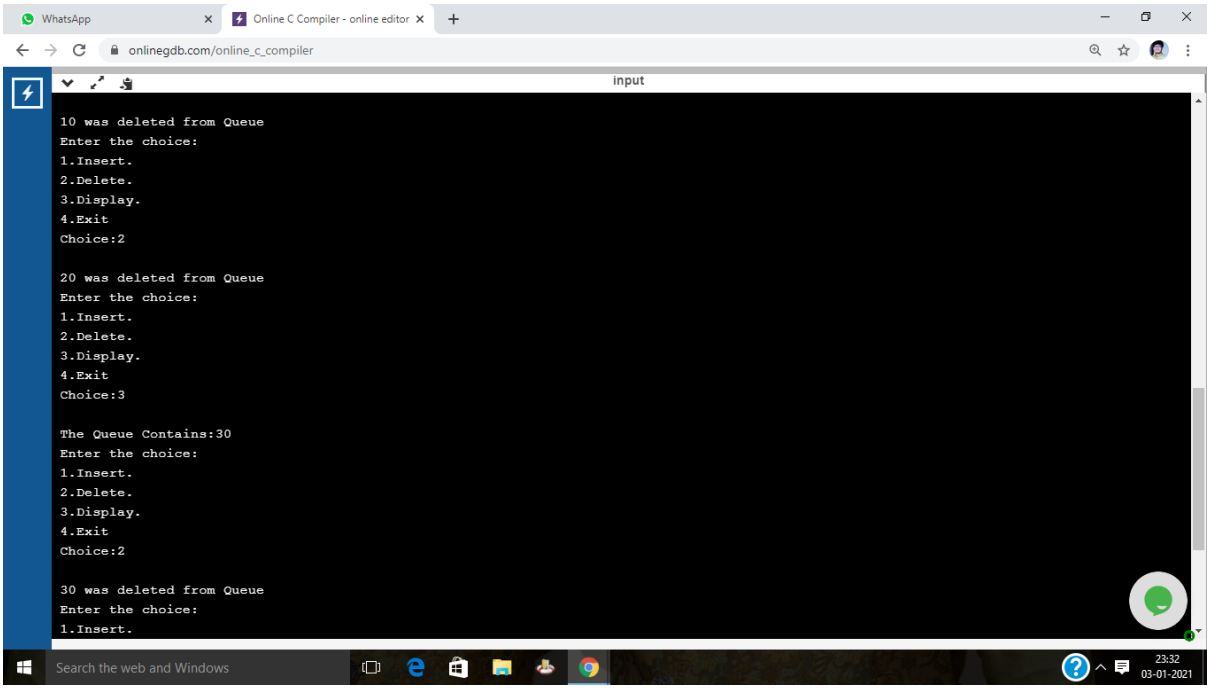
```
#inclu
de<std
io.h>
          #include<stdlib.h>
          struct node *front=NULL, *rear=NULL;
          struct node
          {
              int data;
              struct node *next;
          };
          void queue();
          void insNode(int);
```

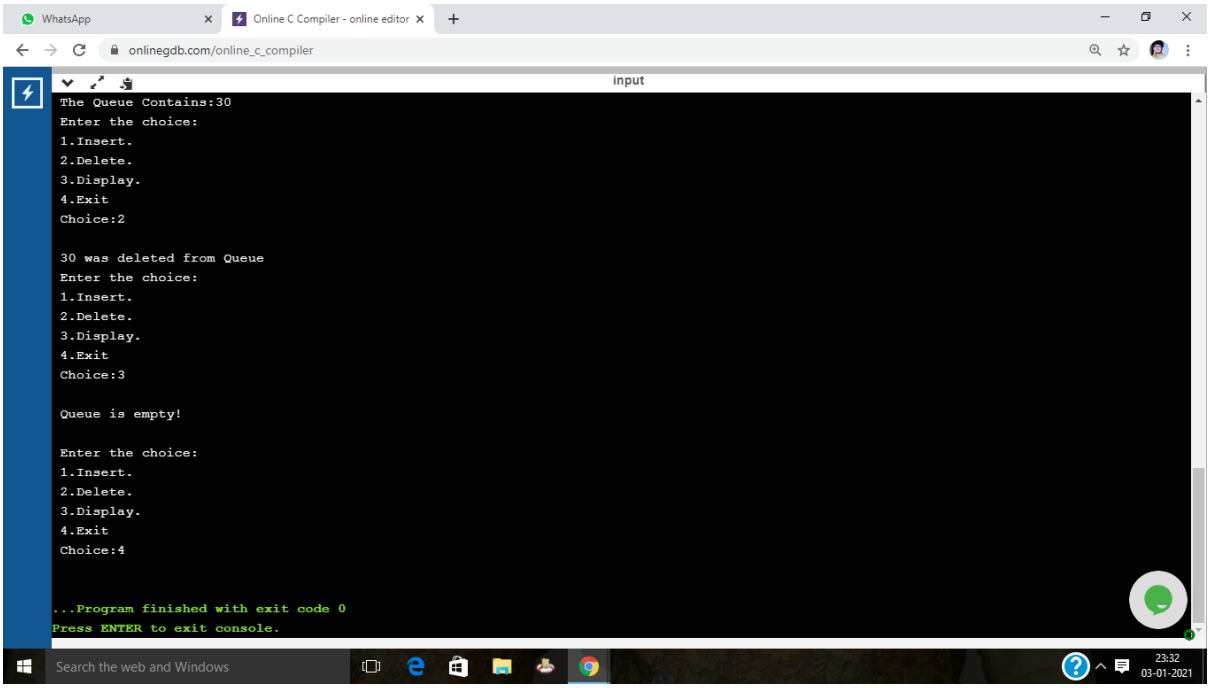
```
int delNode();
void display();
int main()
{
    queue();
    return 1;
}
void queue()
{
    int choice=0,ele=0;
    do
    {
        printf("\n Enter the choice:\n 1.Insert.\n 2.Delete.\n 3.Display.\n 4.Exit\n Choice:");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
                printf("\n Enter the element to Insert: ");
                scanf("%d",&ele);
                insNode(ele);
                break;
        case 2:
                ele=delNode();
                printf("\n %d was deleted from Queue ",ele);
                break;
        case 3:
                display();
                break;
        case 4:
                exit(0);
        default:
                printf("\n Input Error Try Again! ");
                queue();
        }
    }while(1);
}
void insNode(int ele)
{
    struct node *newnode;
    newnode =(struct node *) malloc (sizeof(struct node));
    newnode->data=ele;
    newnode->next=NULL;
    if(rear==NULL)
    {
        front=newnode;
        rear=newnode;
    }
    else
    {
        rear->next=newnode;
        rear=newnode;
    }
    printf("\n Element %d was inserted!\n ",ele);
}
int delNode()
{
     int ele;
     struct node *temp;
     if(front==NULL)
     {
         printf("\n Queue UnderFlow, The queue is empty! \n");
         queue();
     }
     ele=front->data;
```

```
     temp=front;
     if(front==rear)
     {
          front=NULL;
          rear=NULL;
     }
     else
     {
        front=front->next;
     }
     free(temp);
     return ele;
}
void display()
{
    struct node *i;
    if(front==NULL && rear==NULL)
     {
         printf("\n Queue is empty!\n ");
         queue();
     }
    printf("\n The Queue Contains:");
    for(i=front;i!=NULL;i=i->next)
        printf("%d ",i->data);
}
```


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:1
Enter the element to Insert: 10
Element 10 was inserted!
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:1
Enter the element to Insert: 20
Element 20 was inserted!
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Search the web and Windows


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:1
Enter the element to Insert: 30
Element 30 was inserted!
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:3
The Queue Contains:10 20 30
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:2
10 was deleted from Queue
Enter the choice:
Search the web and Windows
23:31
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
10 was deleted from Queue
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:2
20 was deleted from Queue
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:3
The Queue Contains:30
Enter the choice:
1.Insert.
2.Delete.
3.Display.
4.Exit
Choice:2
30 was deleted from Queue
Enter the choice:
1.Insert.
Search the web and Windows
23:32
03-01-2021

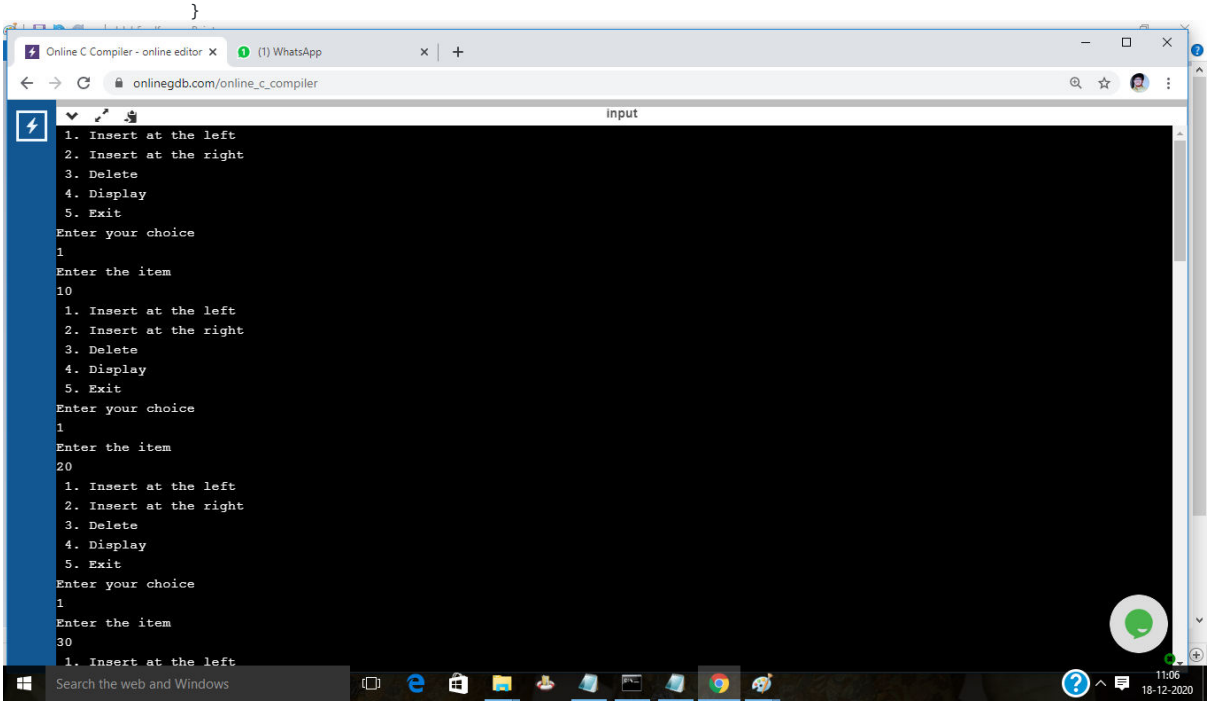


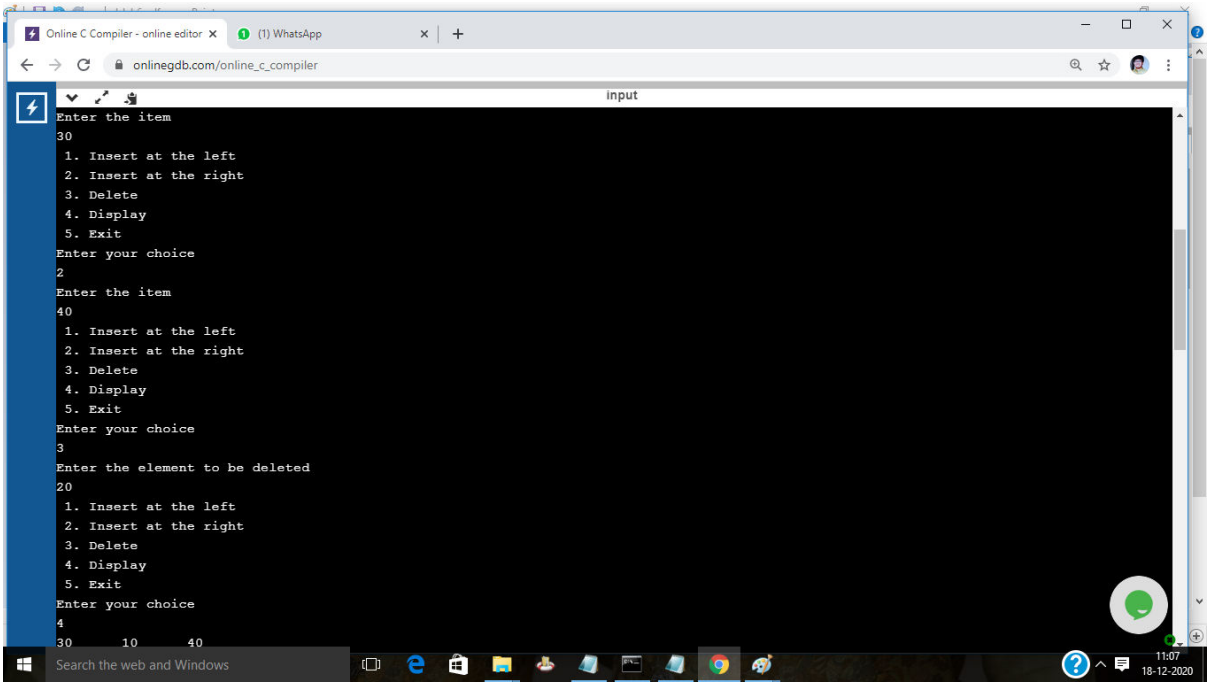
LAB PROGRAM – 9

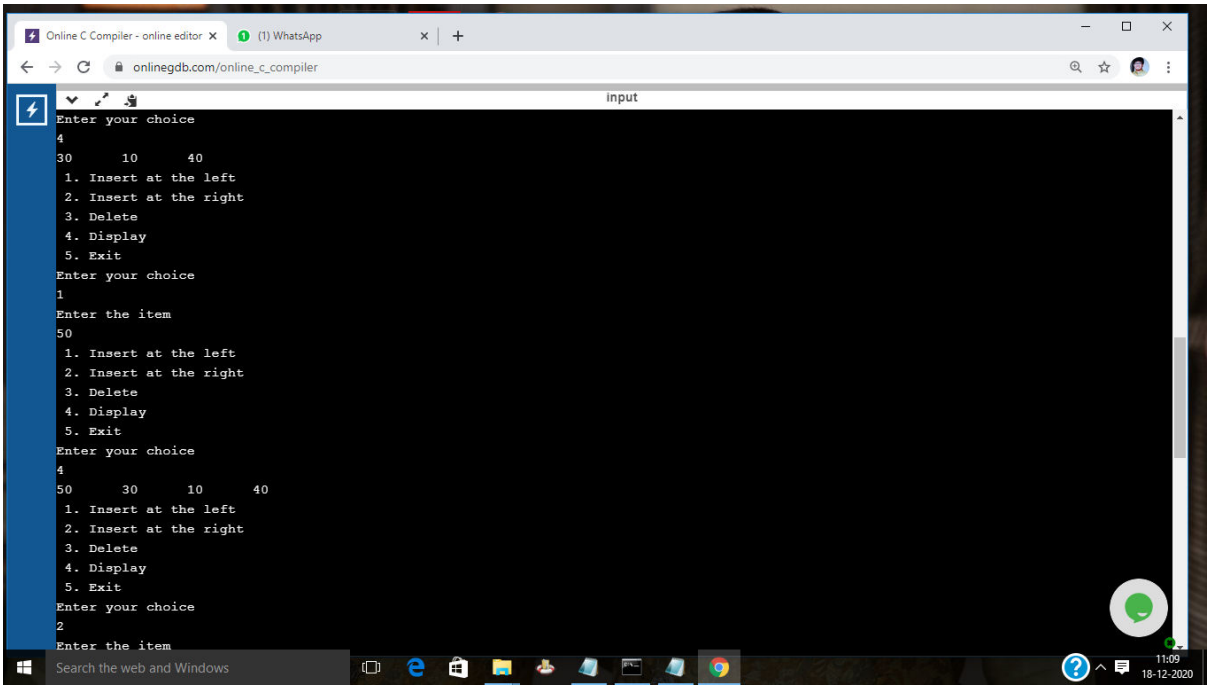
```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                    {
                              int data;
                              struct node *next;
                              struct node *prev;
                    };
                    struct node *head=NULL;
                    void insert_left()
                    {
                               struct node *new_node;
                               new_node=(struct node*)malloc(sizeof(struct node));
                               printf("Enter the item\n");
                               scanf("%d",&new_node->data);
                               new_node->next=NULL;
                               new_node->prev=NULL;
                               if(head==NULL)
                               {
                                                  head=new_node;
                               }
                               else
                               {
                                                  new_node->next=head;
                                                  head->prev=new_node;
                                                  head=new_node;
                               }
                    }
                    void insert_right()
                    {
                              struct node *new_node,*temp;
```

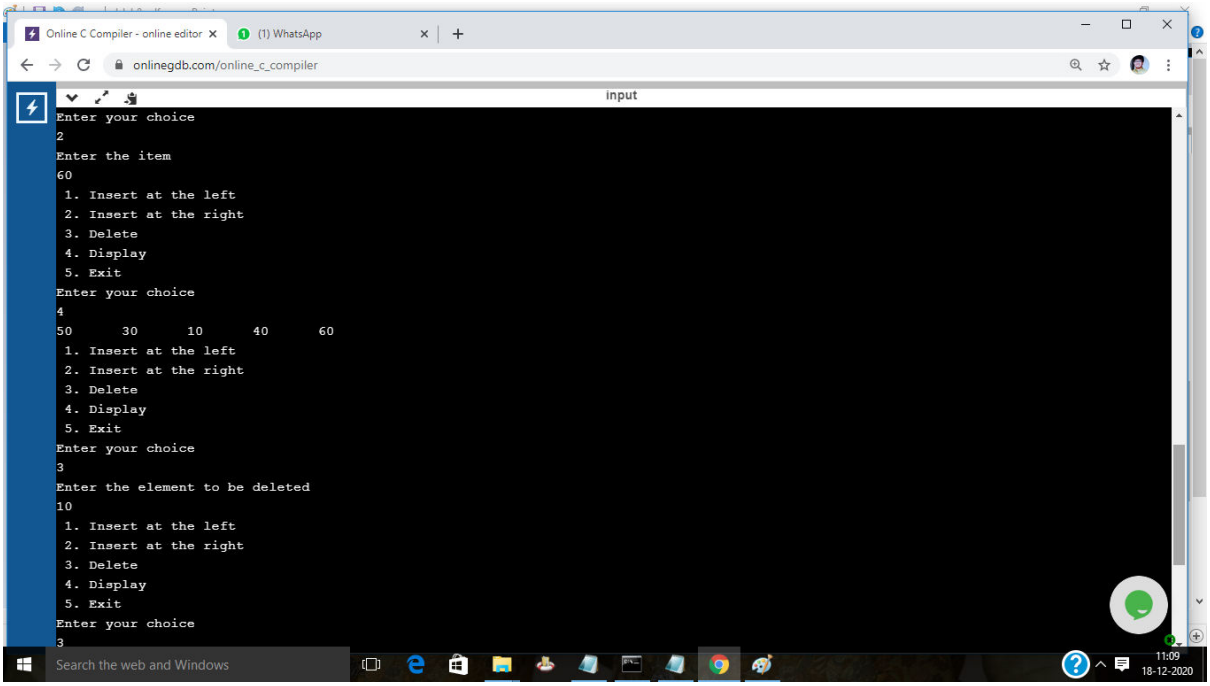
```
          new_node=(struct node*)malloc(sizeof(struct node));
          printf("Enter the item\n");
          scanf("%d",&new_node->data);
          new_node->next=NULL;
          new_node->prev=NULL;
          if(head==NULL)
          {
                    head=new_node;
          }
          else
          {
                     temp=head;
                     while(temp->next!=NULL)
                     temp=temp->next;
                     temp->next=new_node;
                     new_node->prev=temp;
          }
}
void del()
{
          struct node *temp;
          int ele;
    if(head==NULL)
    {
        printf("Empty List \n");
        return;
    }
          printf("Enter the element to be deleted\n");
          scanf("%d",&ele);
          temp=head;
          while(temp->data!=ele)
          {
                    temp=temp->next;
                    if(temp==NULL)
                    {
                     printf("Element is not in the list\n");
                     break;
                    }
           }
           if(temp==head)
           {
                     head=head->next;
           }
           else if(temp->next==NULL)
           {
                              temp=temp->prev;
                              temp->next=NULL;
           }
           else
           {
                     temp->prev->next=temp->next;
                     temp->next->prev=temp->prev;
           }
}
void display()
{
           struct node *ptr;
           ptr=head;
           while(ptr!=NULL)
           {
                     printf("%d\t",ptr->data);
                     ptr=ptr->next;
           }
           printf("\n");
```

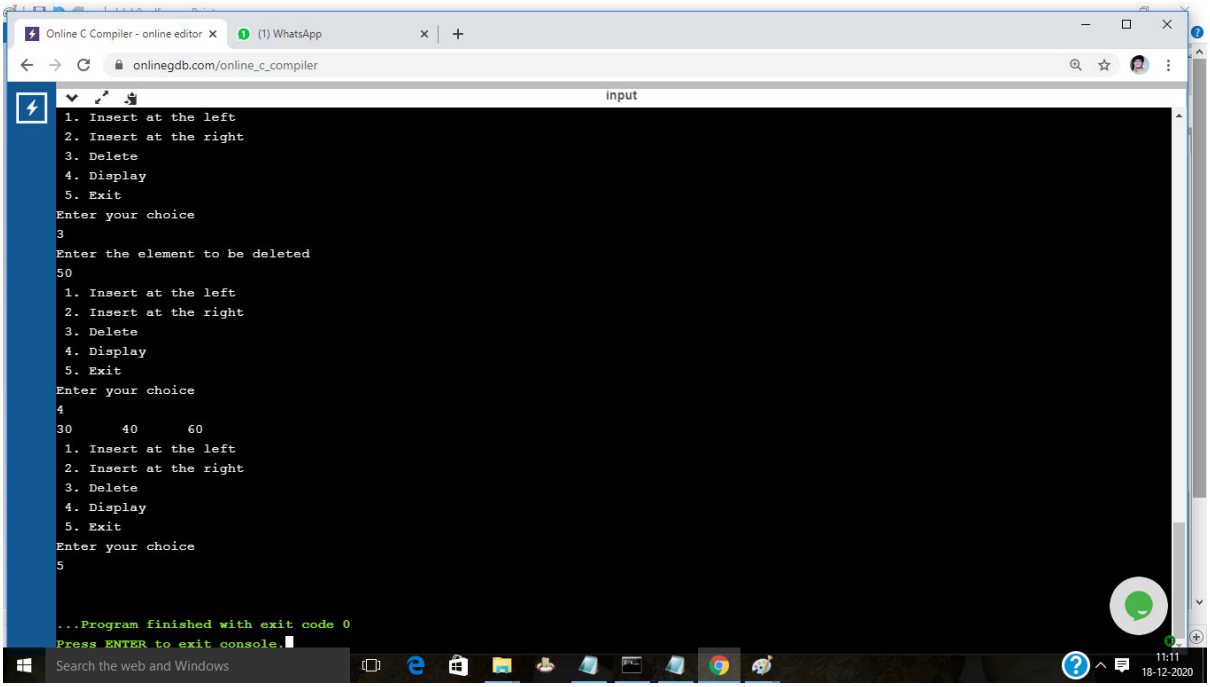
```
}
int main()
{
        int choice;

         while(1)
         {
                  printf(" 1. Insert at the left \n");
                  printf(" 2. Insert at the right \n");
                  printf(" 3. Delete \n");
                  printf(" 4. Display\n");
                  printf(" 5. Exit\n");
                  printf("Enter your choice\n");
                  scanf("%d",&choice);
                  switch(choice)
                  {
                           case 1: insert_left(); break;
                           case 2: insert_right();break;
                           case 3: del(); break;
                           case 4: display(); break;
                           case 5: exit(0);
                  }
         }
}
```




Online C Compiler - online editor
(1) WhatsApp
onlinegdb.com/online_c_compiler
input
Enter the item
30
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
2
Enter the item
40
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
3
Enter the element to be deleted
20
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
4
30 10 40
Search the web and Windows
11:07
18-12-2020


Online C Compiler - online editor
(1) WhatsApp
onlinegdb.com/online_c_compiler
input
Enter your choice
4
30 10 40
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
1
Enter the item
50
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
4
50 30 10 40
1. Insert at the left
2. Insert at the right
3. Delete
4. Display
5. Exit
Enter your choice
2
Enter the item
Search the web and Windows
11:09
18-12-2020



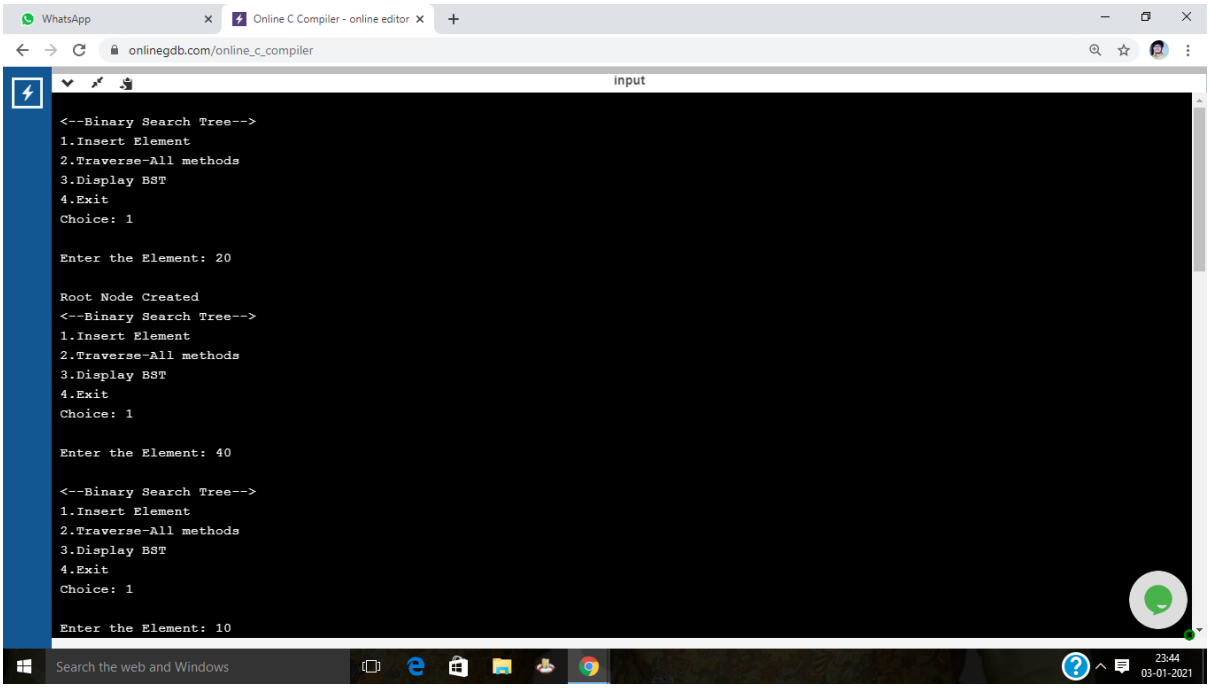


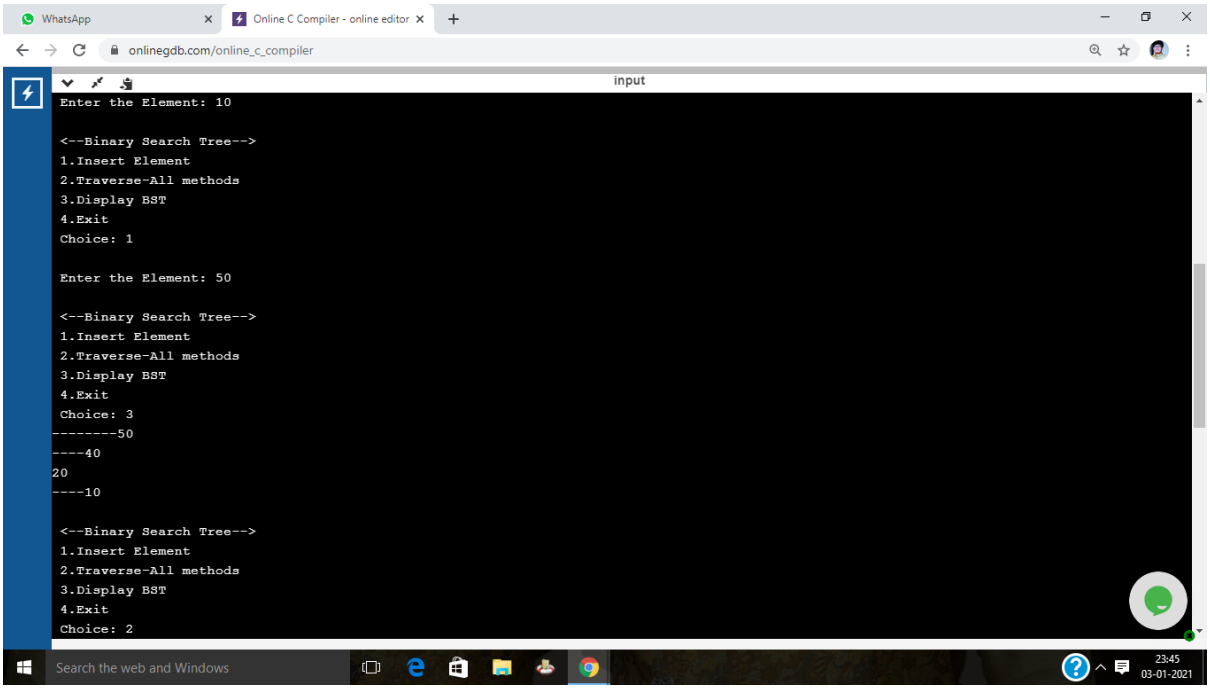
LAB PROGRAM -10

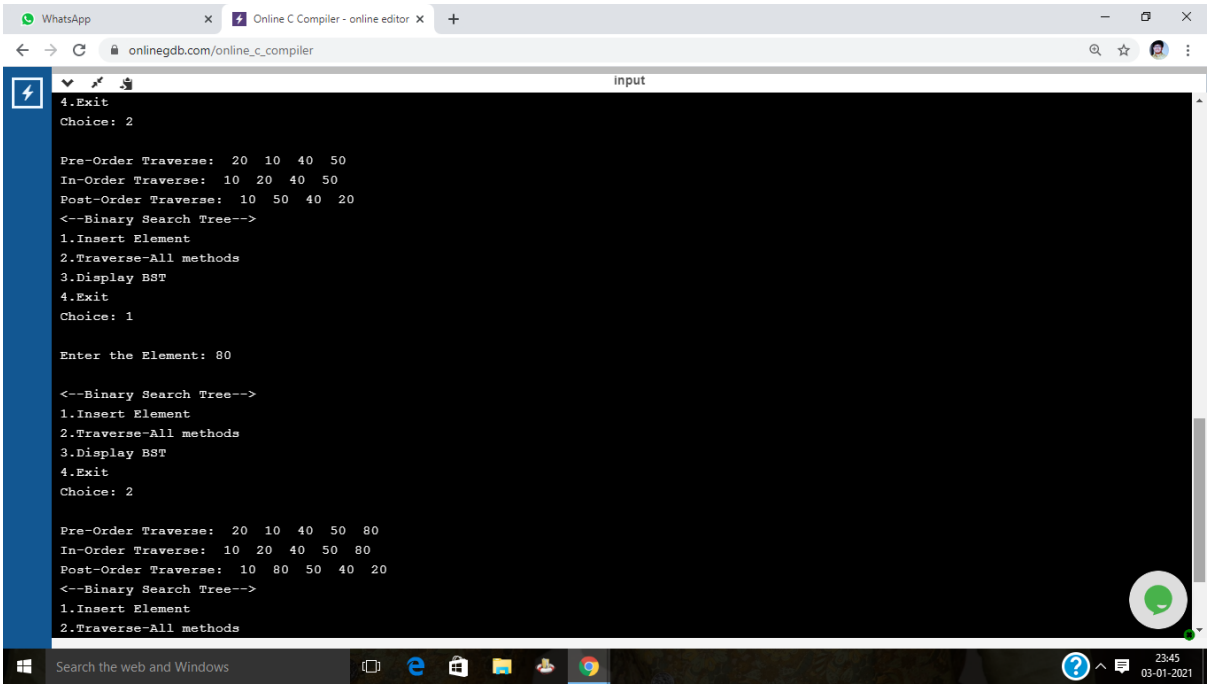
```
#include<stdio.h>
                    #include<stdlib.h>
                    typedef struct node
                    {
                        int data;
                        struct node *left;
                        struct node *right;
                    }Node;
                    void tree();
```

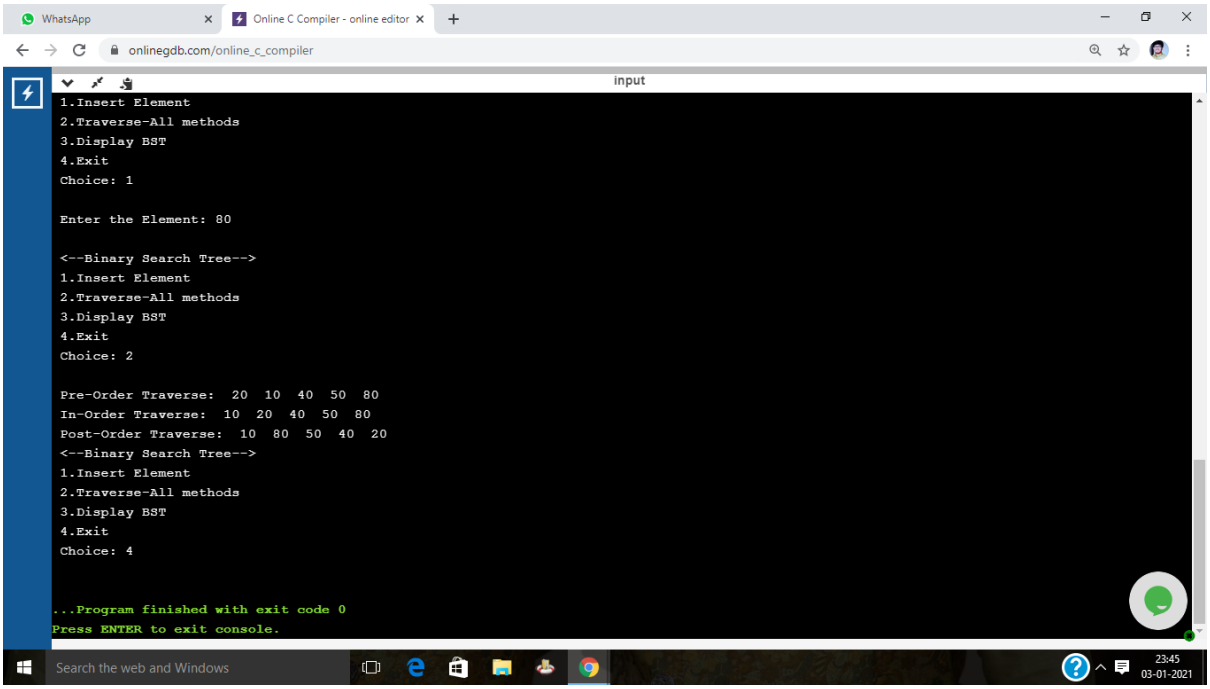
```
Node * create();
Node *insert(Node *,Node *);
void traverse();
void preOrder(Node *);
void inOrder(Node *);
void postOrder(Node *);
void display(Node *,int);
 Node *root;
int main()
{
    tree();
    return 0;
}
void tree()
{
    int choice;
    printf("\n <--Binary Search Tree-->\n 1.Insert Element\n 2.Traverse-All methods\n 3.Display BST\n
4.Exit\n Choice: ");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1: insert(root,create()); break;
        case 2: traverse(); break;
        case 3: if(root==NULL)
                printf("\n Tree is Empty!");
                else
                display(root,0);
                break;
        case 4: exit(0);break;
        default: printf("\n Error Choice !\n ");
                tree();
    }
    tree();
}
Node * create()
{
    Node* newnode=(Node *)malloc(sizeof(Node));
    printf("\n Enter the Element: ");
    scanf("%d",&newnode->data);
    newnode->left=NULL;
    newnode->right=NULL;
    return newnode;
}
Node * insert(Node *Root,Node *newNode)
{
    if(root==NULL)
    {
        root=newNode;
        printf("\n Root Node Created ");
    }
    else
    {
    if(newNode->data>Root->data)
    {
        if(Root->right==NULL)
        {
            Root->right=newNode;
        }
        else
        insert(Root->right,newNode);
    }
    else
    if(newNode->data<Root->data)
    {
```

```
        if(Root->left==NULL)
        {
            Root->left=newNode;
        }
        else
        insert(Root->left,newNode);
    }
    }
}
void traverse()
{
    if(root==NULL)
    {
        printf("\n The Tree Is Empty! ");
        return;
    }
    printf("\n Pre-Order Traverse: ");
    preOrder(root);
    printf("\n In-Order Traverse: ");
    inOrder(root);
    printf("\n Post-Order Traverse: ");
    postOrder(root);
}
void preOrder(Node *Root)
{
    if(Root!=NULL){
    printf(" %d ",Root->data);
    preOrder(Root->left);
    preOrder(Root->right);
    }
}
void inOrder(Node *Root)
{
    if(Root!=NULL){
    inOrder(Root->left);
    printf(" %d ",Root->data);
    inOrder(Root->right);
    }
}
void postOrder(Node *Root)
{
    if(Root!=NULL){
    postOrder(Root->left);
    postOrder(Root->right);
    printf(" %d ",Root->data);
    }
}
void display(Node* root,int i)
{
  int j;
  if(root!=NULL)
  {
    display(root->right,i+1);
    for(j=0;j<i;j++)
      printf("----");
    printf("%d\n",root->data);
    display(root->left,i+1);
  }
}
```


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 20
Root Node Created
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 40
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 10
Search the web and Windows
23:44
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
Enter the Element: 10
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 50
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 3
--------50
----40
20
----10
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 2
Search the web and Windows
23:45
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
4.Exit
Choice: 2
Pre-Order Traverse: 20 10 40 50
In-Order Traverse: 10 20 40 50
Post-Order Traverse: 10 50 40 20
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 80
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 2
Pre-Order Traverse: 20 10 40 50 80
In-Order Traverse: 10 20 40 50 80
Post-Order Traverse: 10 80 50 40 20
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
Search the web and Windows
23:45
03-01-2021


WhatsApp
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
input
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 1
Enter the Element: 80
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 2
Pre-Order Traverse: 20 10 40 50 80
In-Order Traverse: 10 20 40 50 80
Post-Order Traverse: 10 80 50 40 20
<--Binary Search Tree-->
1.Insert Element
2.Traverse-All methods
3.Display BST
4.Exit
Choice: 4
...Program finished with exit code 0
Press ENTER to exit console.
Search the web and Windows
23:45
03-01-2021