VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

DATA STRUCTURE LAB RECORD

Submitted by

SHIVANI GAHLOT(1BM19CS150)

Under the Guidance of

Prof. LOHITH J.J
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 SHIVANI **GAHLOT (1BM19CS150)** 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 Visveswaraya 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
Prof. Prof. Sheelal VA
Assistant Professor
BMSCE, Bengaluru

Signature of the HOD
Dr. Umadevi V
Associate Prof.& Head, Dept. of CSE
BMSCE, Bengaluru

External Viva

Name of the Examiner | Signature with date

1._____ _____

2. _____ _____

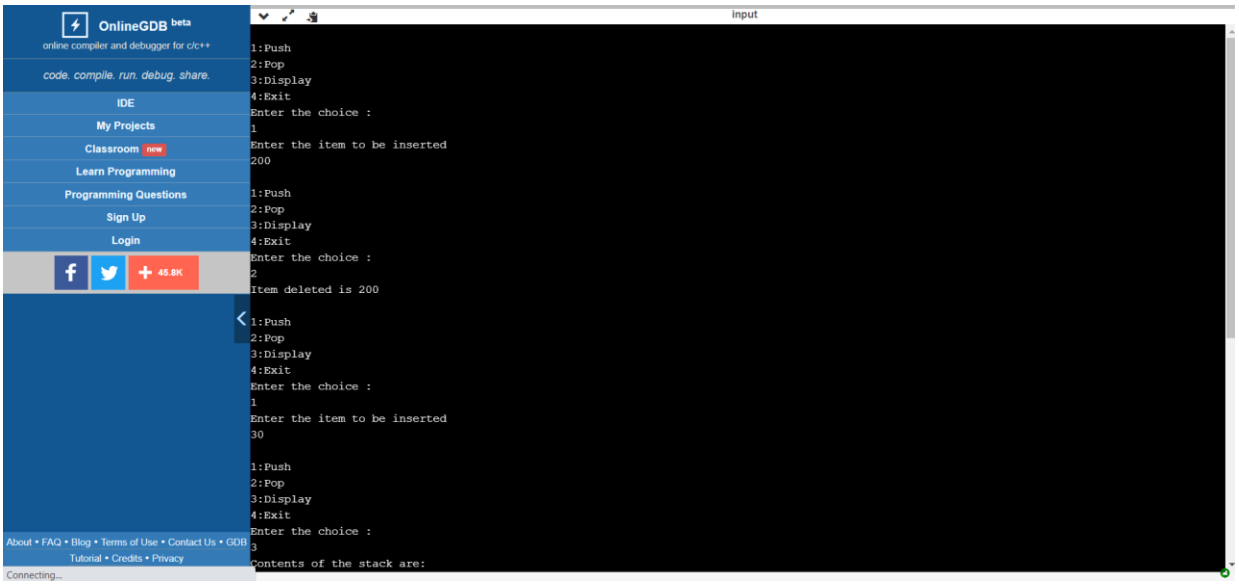
LAB PROGRAM-1: Write a program to simulate the working of stack using an array with the following: a) Push b) Pop c) Display The program should print appropriate messages for stack overflow, stack underflow.

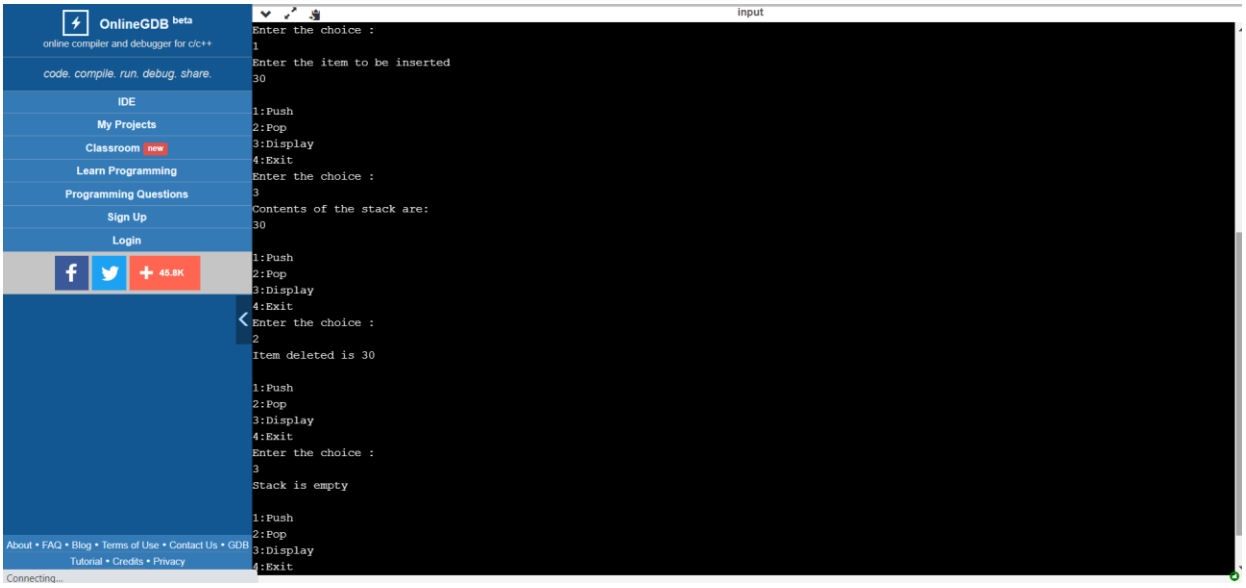
```
#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define STACK_SIZE 5

int top=-1;

int s [10];

int item;

void push ()

{

if (top == STACK_SIZE - 1)

{

printf("Stack Overflow \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 are: \n");
for(i=top;i>=0;i--)
{
printf("%d \n", s[i]);
}
}
void main()
{
int item_deleted, choice;
```

```
for(;;)
{
printf("\n1:Push \n2:Pop \n3:Display \n4: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);
}
}
getch();
```

}


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
Connecting...
input
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
1
Enter the item to be inserted
200
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
2
Item deleted is 200
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
1
Enter the item to be inserted
30
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
3
Contents of the stack are:


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
Connecting...
input
Enter the choice :
1
Enter the item to be inserted
30
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
3
Contents of the stack are:
30
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
2
Item deleted is 30
1:Push
2:Pop
3:Display
4:Exit
Enter the choice :
3
Stack is empty
1:Push
2:Pop
3:Display
4:Exit

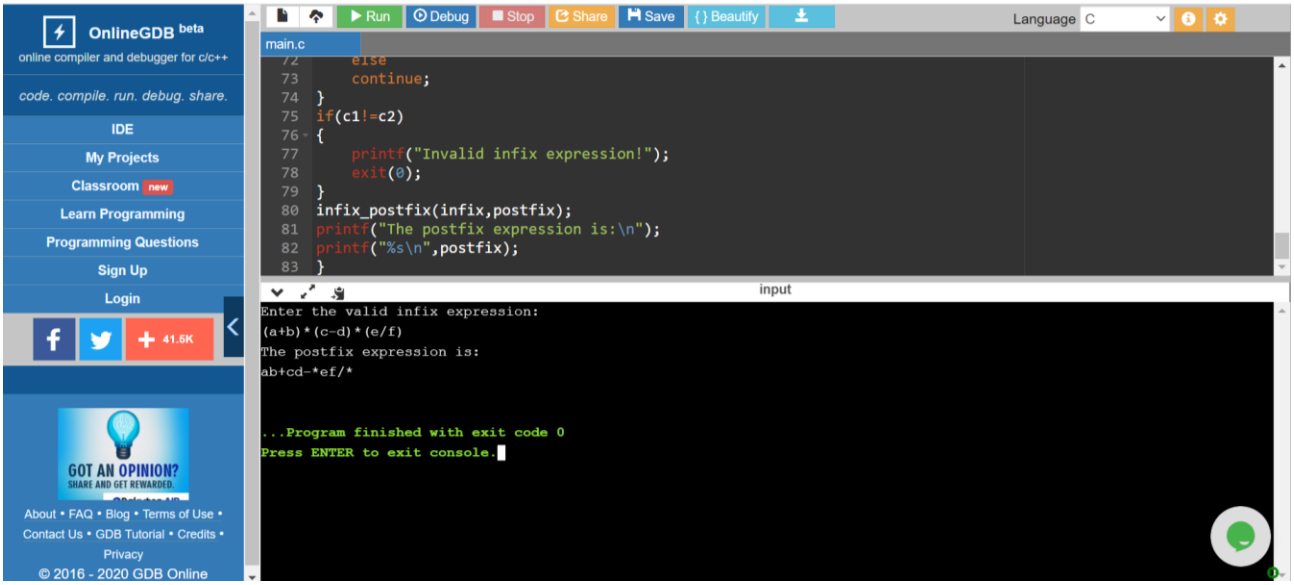
LAB PROGRAM-2: WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide).

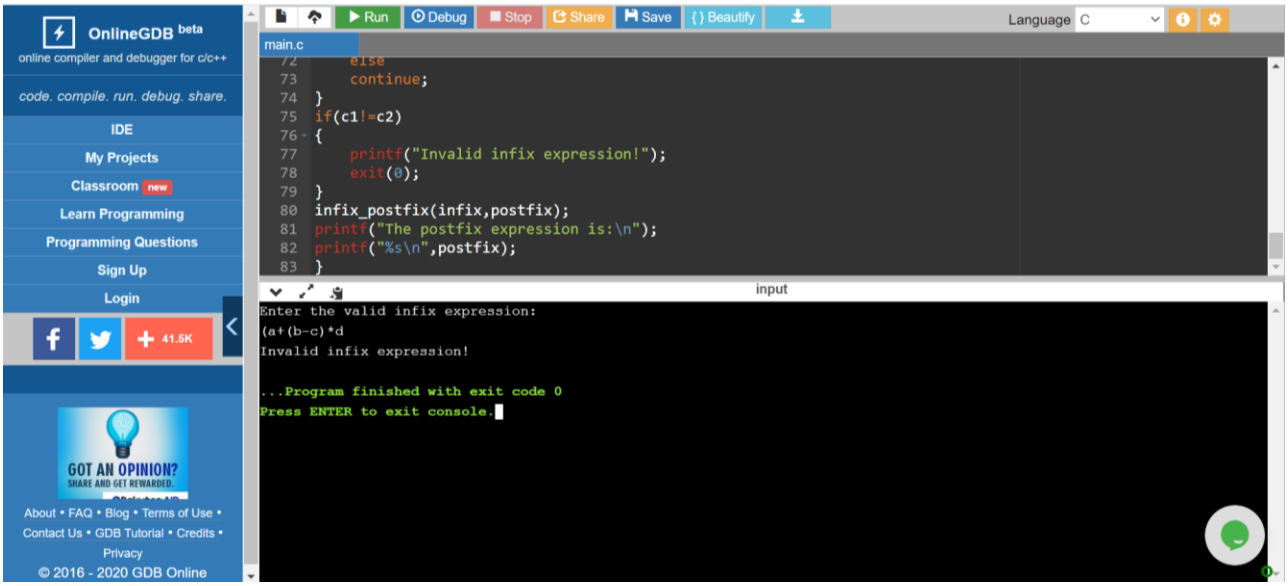
```
#include<stdio.h>

#include<stdlib.h>

#include<string.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,j,i;
char s[30];
char 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],postfix[20];
 int c1=0,c2=0;
printf("Enter the valid infix expression:\n");
```

```
scanf("%s",infix);
for(int k=0;k<strlen(infix);k++)
{
    if(infix[k]=='(')
    c1++;
    else if(infix[k]==')')
    c2++;
    else
    continue;
}
if(c1!=c2)
{
    printf("Invalid infix expression!");
    exit(0);
}
infix_postfix(infix,postfix);
printf("The postfix expression is:\n");
printf("%s\n",postfix);
}
```


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
41.5K
GOT AN OPINION?
SHARE AND GET REWARDED.
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
else
continue;
}
if(c1!=c2)
{
printf("Invalid infix expression!");
exit(0);
}
infix_postfix(infix,postfix);
printf("The postfix expression is:\n");
printf("%s\n",postfix);
}
input
Enter the valid infix expression:
(a+b)*(c-d)*(e/f)
The postfix expression is:
ab+cd-*ef/*
...Program finished with exit code 0
Press ENTER to exit console.

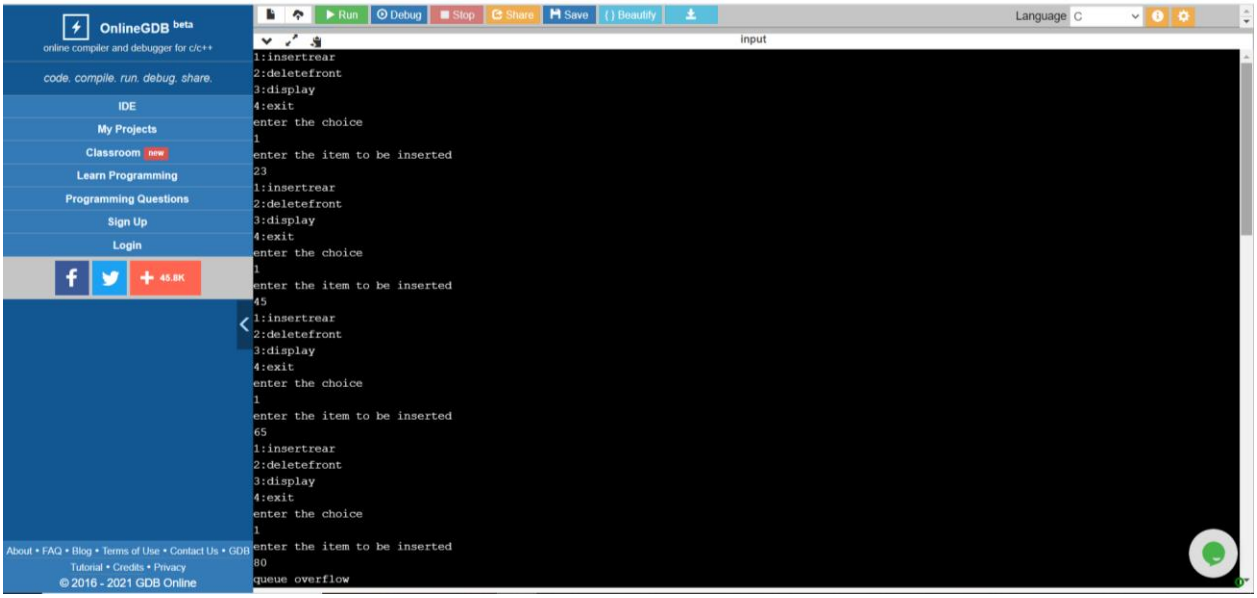

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
41.5K
GOT AN OPINION?
SHARE AND GET REWARDED.
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
else
continue;
}
if(c1!=c2)
{
printf("Invalid infix expression!");
exit(0);
}
infix_postfix(infix,postfix);
printf("The postfix expression is:\n");
printf("%s\n",postfix);
}
input
Enter the valid infix expression:
(a+(b-c)*d
Invalid infix expression!
...Program finished with exit code 0
Press ENTER to exit console.

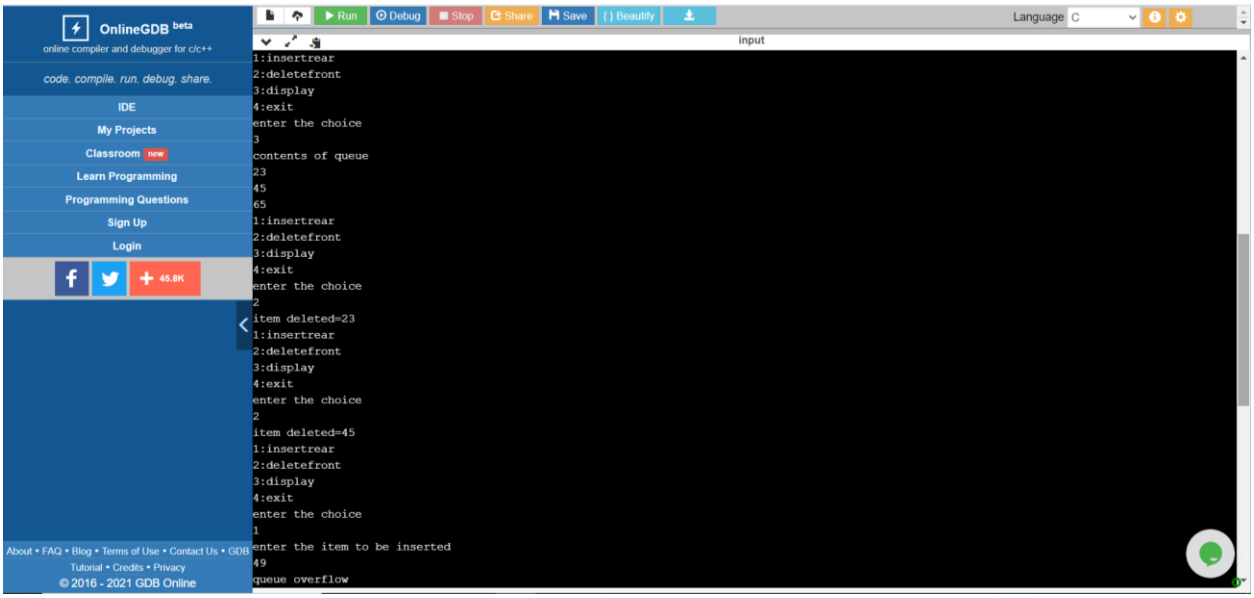
LAB PROGRAM-3: WAP to simulate the working of a queue of integers using an array. Provide the following operations a) Insert b) Delete c) Display The program should print appropriate messages for queue empty and queue overflow conditions.

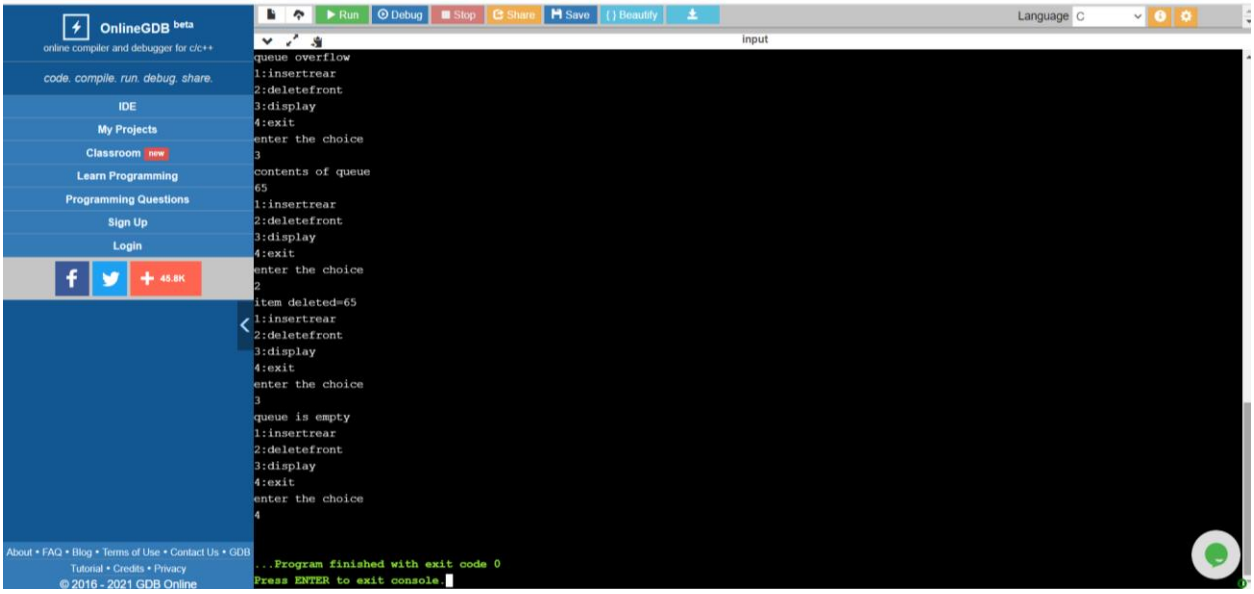
```
#include<stdio.h>

#include<stdlib.h>

#define QUE_SIZE 3

int item,front=0,rear=-1,q[10];

void insertrear()

{if(rear==QUE_SIZE-1)

{

printf("queue overflow\n");

return;

}

rear=rear+1;

q[rear]=item;

}int deletefront()

{if (front>rear)

{front=0;

rear=-1;

return -1;

}return q[front++];

}void displayQ()
```

```
{int i;
if (front>rear)
{
printf("queue is empty\n");
return;
}
printf("contents of queue\n");
for(i=front;i<=rear;i++)
{
printf("%d\n",q[i]);
}}
int main()
{
int choice;
for(;;)
{
printf("1:insertrear\n2:deletefront\n3:display\n4: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);
```

```
insertrear ();
break;
case 2:item=deletefront();
if(item==-1)
printf("queue is empty\n");
else
printf("item deleted=%d\n",item);
break;
case 3:displayQ();
break;
default:exit (0);

}

}
}
```


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
input
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
1
enter the item to be inserted
23
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
1
enter the item to be inserted
45
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
1
enter the item to be inserted
65
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
1
enter the item to be inserted
80
queue overflow


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
input
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
3
contents of queue
23
45
65
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
2
item deleted=23
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
2
item deleted=45
1:insertrear
2:deletefront
3:display
4:exit
enter the choice
1
enter the item to be inserted
49
queue overflow



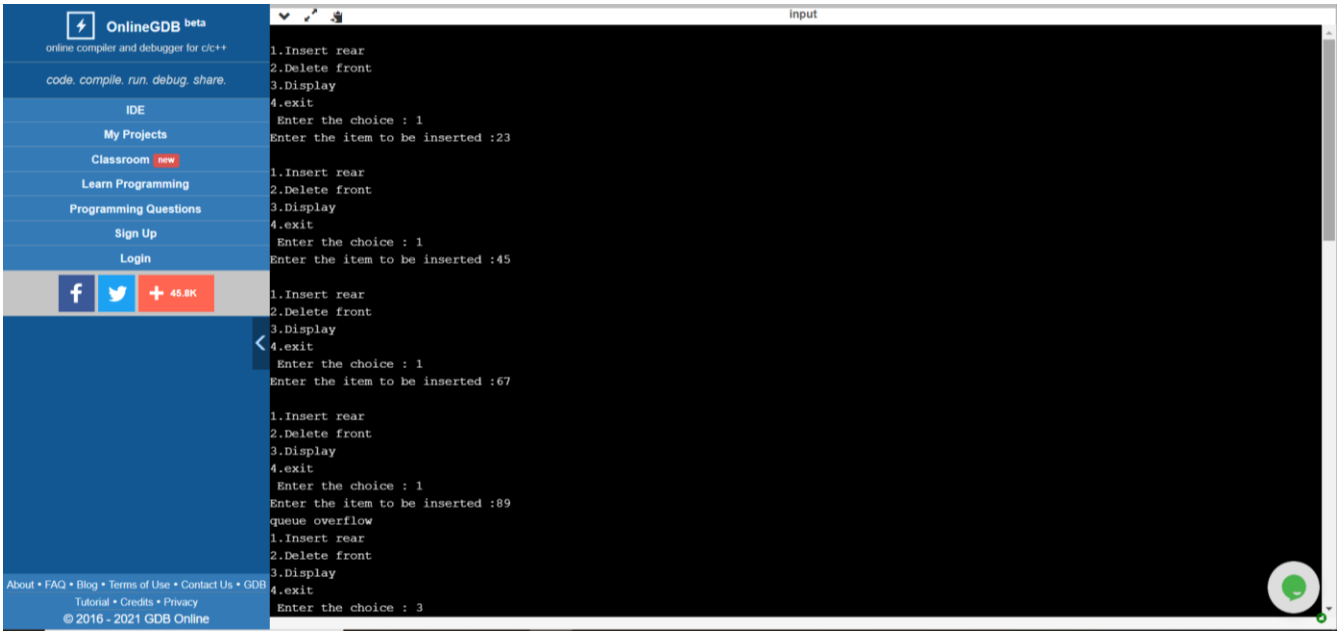
LAB PROGRAM-4: WAP to simulate the working of a circular queue of integers using an array. Provide the following operations. a) Insert b) Delete c) Display The program should print appropriate messages for queue empty and queue overflow conditions.

```
#include<stdio.h>

#include<stdlib.h>

#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 \n2.Delete front \n3.Display \n4.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);
}
}
}
```



OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K

input

```
 Enter the choice : 3
contents of queue
23
45
67
23

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 23

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 45

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
contents of queue
67
23

1.Insert rear
2.Delete front
3.Display
4.exit
```

About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K

input

```
23

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 67

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :74

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
contents of queue
74
45

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


...Program finished with exit code 0
Press ENTER to exit console.
```

About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online

LAB PROGRAM-5: WAP to Implement Singly Linked List with following operations a) a) Create a linked list. b) Insertion of a node at first position, at any position and at end of list. c) Display the contents of the linked list.

```
#include<stdio.h>

#include<stdlib.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;

}

int freenode(NODE x)
```

```
{
free(x);
return 0;
}
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 insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
```

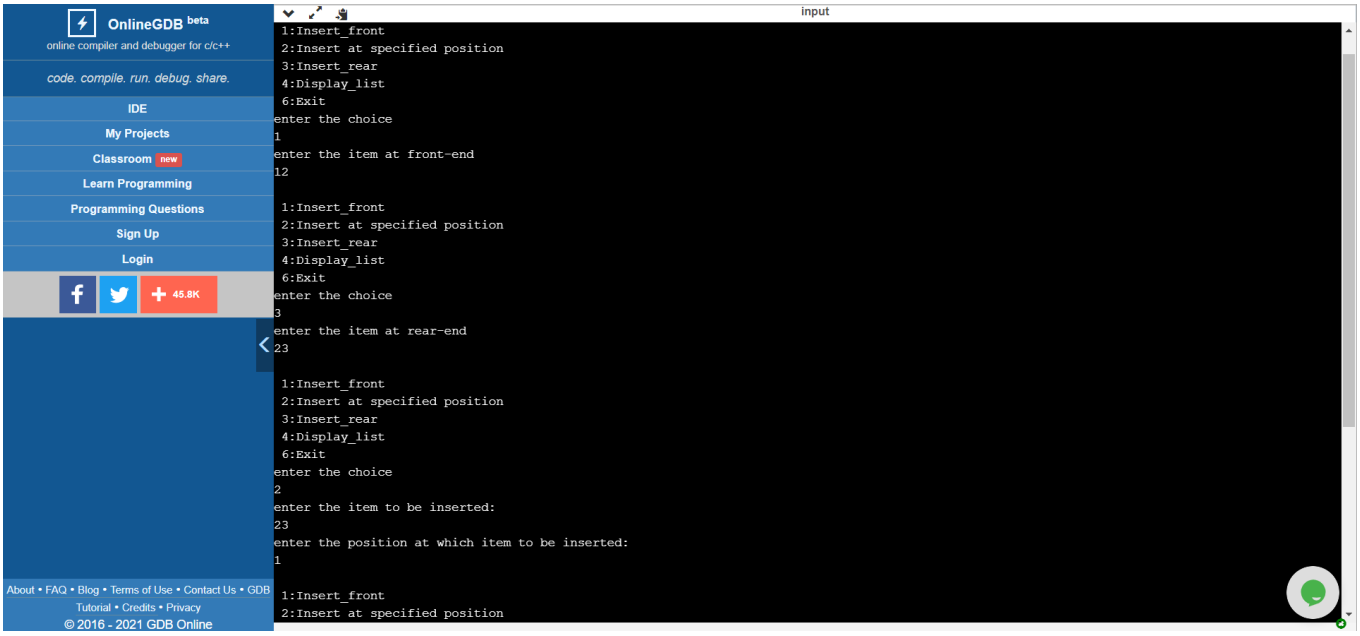
```
if(first==NULL)
return temp;
cur=first;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
return first;
}

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);
}}

NODE insert_pos( int item, int pos, NODE first)
{
NODE temp;
NODE prev,cur;
```

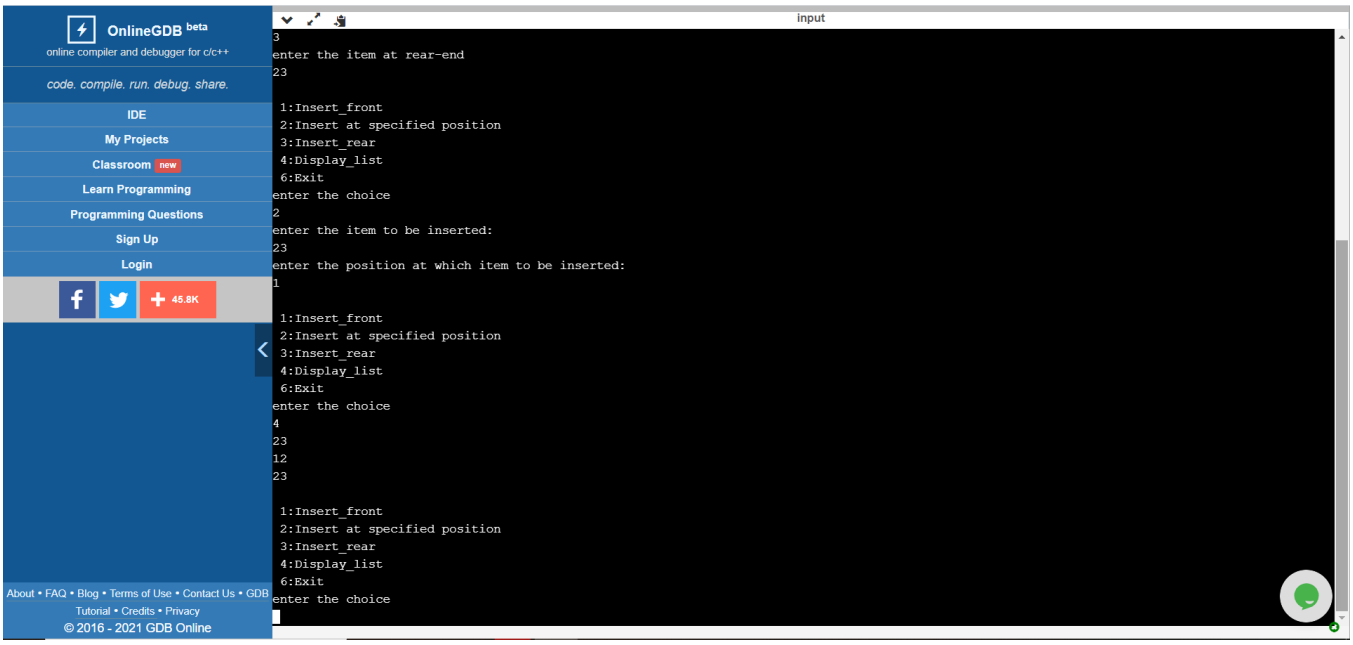
```
int count;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL && pos==1)
{
return temp;
}
if(first==NULL)
{
printf("invalid position \n");
return first;
}
if(pos==1)
{
temp->link=first;
return temp;
}
count=1;
prev=NULL;
cur=first;
while(cur!=NULL && count!=pos)
{
```

```
prev=cur;
cur=cur->link;
count++;
}
if(count==pos)
{
prev->link=temp;
temp->link=cur;
return first;
}
printf("invalid position \n");
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert_front\n 2:Insert at specified position \n 3:Insert_rear\n
4:Display_list\n 6: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:printf("enter the item to be inserted:\n");
      scanf("%d",&item);
      printf("enter the position at which item to be inserted:\n");
      scanf("%d",&pos);
      first=insert_pos(item,pos,first);
break;
case 3:printf("enter the item at rear-end\n");
scanf("%d",&item);
first=insert_rear(first,item);
break;
case 4:display(first);
break;
default:exit(0);
break;
}
}
```

```
return 0;
}
```


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice
1
enter the item at front-end
12
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice
3
enter the item at rear-end
23
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice
2
enter the item to be inserted:
23
enter the position at which item to be inserted:
1
1:Insert_front
2:Insert at specified position


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
3
enter the item at rear-end
23
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice
2
enter the item to be inserted:
23
enter the position at which item to be inserted:
1
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice
4
23
12
23
1:Insert_front
2:Insert at specified position
3:Insert_rear
4:Display_list
6:Exit
enter the choice

LAB PROGRAM-6: WAP to Implement Singly Linked List with following operations a) a) Create a linked list. b) Deletion of first element, specified element and last element in the list. c) Display the contents of the linked list.

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

```
}
int freenode(NODE x)
{
free(x);
return 0;
}
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)
```

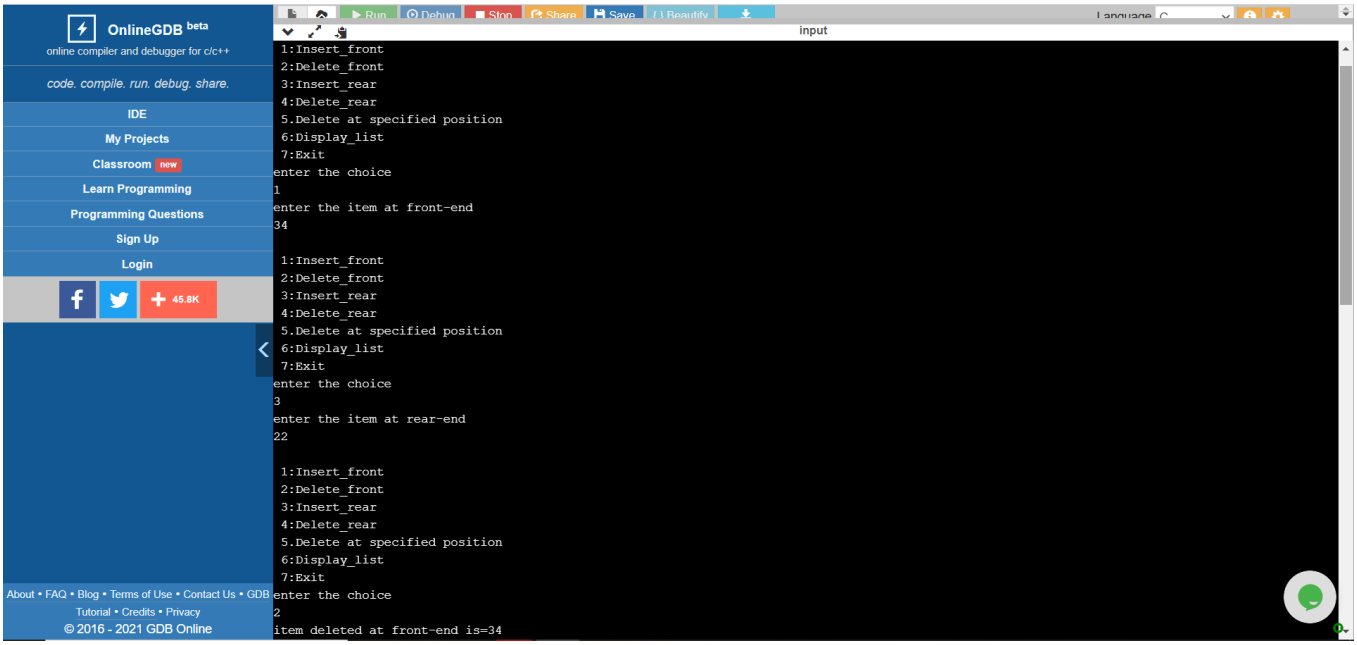
```
{
prev=cur;
cur=cur->link;
}
printf("iten deleted at rear-end is %d",cur->info);
free(cur);
prev->link=NULL;

return first;
}
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);
}
}
NODE delete_pos(int pos, NODE first)
```

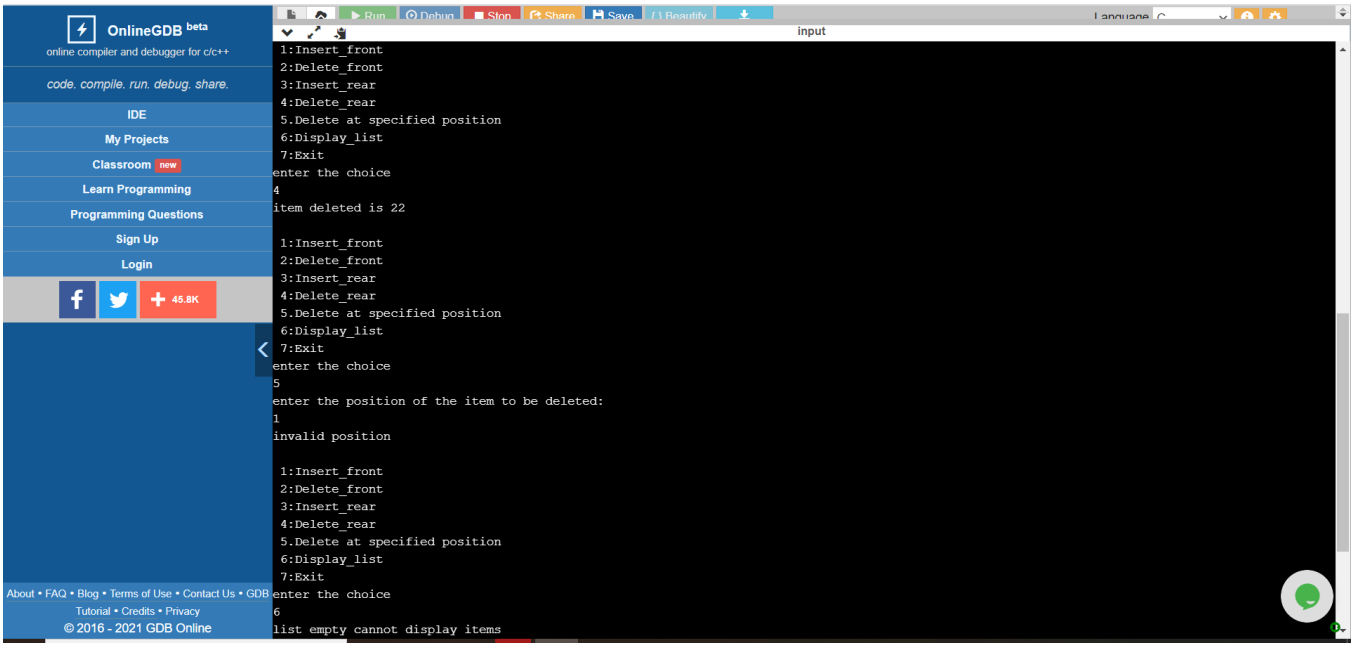
```
{
NODE cur;
NODE prev;
int count;
if(first==NULL || pos<=0)
{
printf("invalid position \n");
return NULL;
}
if (pos==1)
{
cur=first;
first=first->link;
freenode(cur);
return first;
}
prev=NULL;
cur=first;
count=1;
while(cur!=NULL)
{
```

```
if(count==pos) break;
prev=cur;
cur=cur->link;
count++;
}
if(count!=pos)
{
printf("invalid position \n");
return first;
}
if(count!=pos)
{
printf("invalid position specified \n");
return first;
}
prev->link=cur->link;
freenode(cur);
return first;
}
int main()
{
```

```
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert_front\n 2:Delete_front\n 3:Insert_rear\n
4:Delete_rear\n 5.Delete at specified position \n 6:Display_list\n
7: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 of the item to be deleted: \n");
        scanf("%d",&pos);
        first=delete_pos(pos,first);
    break;

case 6:display(first);
break;
default:exit(0);
break;}}
return 0;
}
```





LAB PROGRAM-7: WAP Implement Single Link List with following operations a) a) Sort the linked list. b) Reverse the linked list. c) Concatenation of two linked lists.

```
#include<stdio.h>

#include<conio.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("\nMemory is full\n");
        exit(0);
    }
    return 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 IF(NODE second,int item)
```

```
{
    NODE temp;
    temp=getnode();
    temp->info=item;
    temp->link=NULL;
    if(second==NULL)
        return temp;
    temp->link=second;
    second=temp;
    return second;
}

NODE IR(NODE second,int item)
{
    NODE temp,cur;
    temp=getnode();
    temp->info=item;
    temp->link=NULL;
    if(second==NULL)
        return temp;
    cur=second;
```

```
    while(cur->link!=NULL)
        cur=cur->link;
    cur->link=temp;
    return second;
}

NODE reverse(NODE first)
{
    NODE cur,temp;
    cur=NULL;
    while(first!=NULL)
    {
        temp=first;
        first=first->link;
        temp->link=cur;
        cur=temp;
    }
    return cur;
}

NODE ascending(NODE first)
{
```

```
    NODE prev=first;
    NODE cur=NULL;
int temp;
if(first== NULL)
{
   return 0;
}
    else
    {
    while(prev!= NULL)
    {
        cur = prev->link;
        while(cur!= NULL)
        {
                if(prev->info > cur->info)
                {
                    temp = prev->info;
                prev->info = cur->info;
                cur->info = temp;
            }
            cur = cur->link;
```

```
        }
        prev= prev->link;
    }
}
return first;
}

NODE descending(NODE first)
{
    NODE prev=first;
    NODE cur=NULL;
int temp;
if(first==NULL)
{
    return 0;
}
else
{
    while(prev!= NULL)
    {
        cur = prev->link;
```

```
        while(cur!= NULL)
        {
                if(prev->info < cur->info)
                {
                    temp = prev->info;
                        prev->info = cur->info;
                        cur->info = temp;
            }
            cur = cur->link;
        }
        prev= prev->link;
    }
    }
    return first;
}

NODE concatenate(NODE first,NODE second)
{
    NODE cur;
    if(first==NULL)
        return second;
```

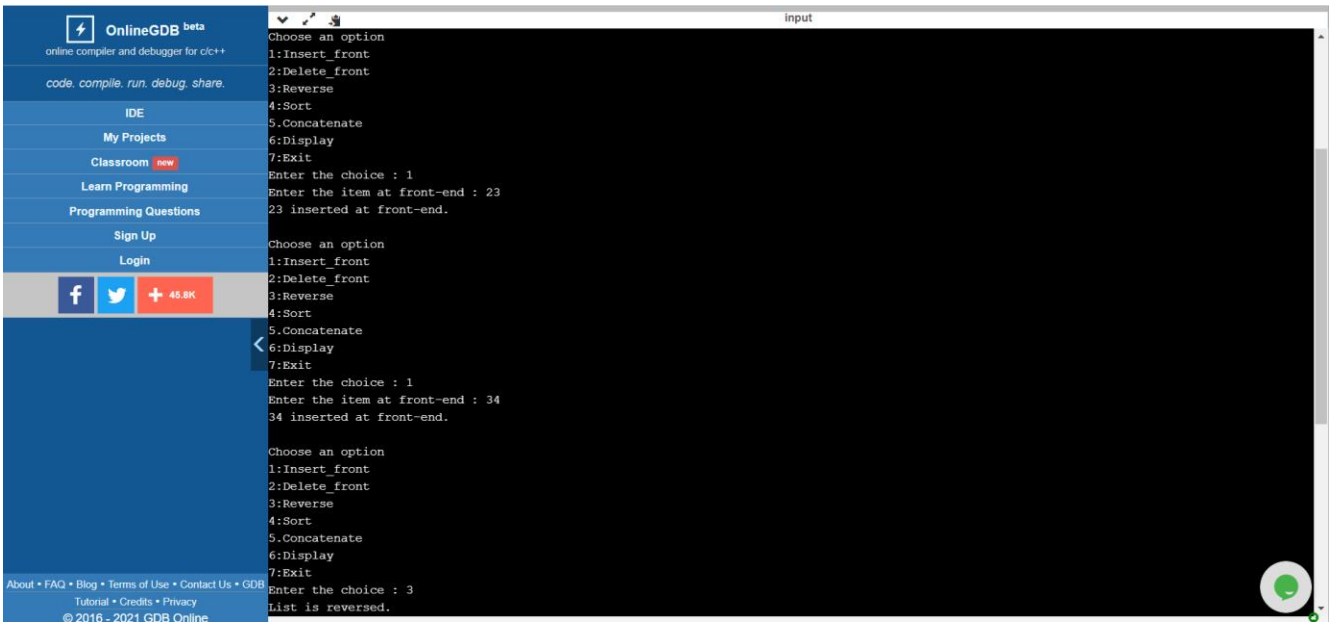
```
    if(second==NULL)
        return first;
    cur=first;
    while(cur->link!=NULL)
    {
        cur=cur->link;
    }
    cur->link=second;
    return first;
}

void display(NODE first)
{
    NODE temp;
    if(first==NULL)
        printf("List is empty. Cannot display items.\n");
    printf("List contents are : ");
    for(temp=first;temp!=NULL;temp=temp->link)
    {
        printf("\n%d",temp->info);
    }
```

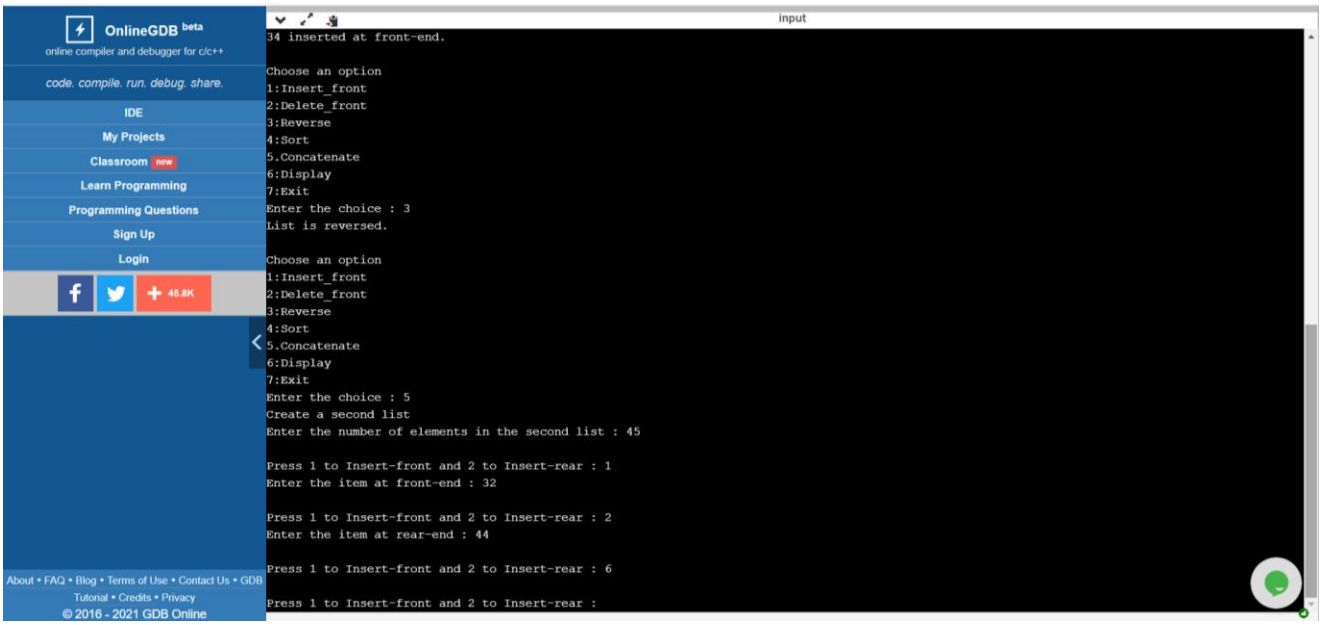
```
}

void main()
{
    int item,choice,pos,element,option,choice2,item1,num;
    NODE first=NULL;
    NODE second=NULL;
    for(;;)
    {
        printf("\n\nChoose an option");
        printf("\n1:Insert_front \n2:Delete_front \n3:Reverse
\n4:Sort \n5.Concatenate \n6:Display \n7:Exit\n");
        printf("Enter the choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("Enter the item at front-end : ");
                    scanf("%d",&item);
                    first=insert_front(first,item);
                    printf("%d inserted at front-end.",first->info);
                    break;
            case 2: first=delete_front(first);
```

```
                break;
        case 3: first=reverse(first);
                printf("List is reversed.");
                break;
        case 4: printf("Press 1 for Ascending-sort and 2 for
Descending-sort : ");
                scanf("%d",&option);
                if(option==1)
                {
                    first=ascending(first);
                    printf("List is sorted in ascending order.");
                }
                if(option==2)
                {
                    first=descending(first);
                    printf("List is sorted in descending order.");
                }
                break;
        case 5: printf("Create a second list\n");
                printf("Enter the number of elements in the second
list : ");
                scanf("%d",&num);
```

```
for(int i=1;i<=num;i++)
{
    printf("\nPress 1 to Insert-front and 2 to Insert-
rear : ");
    scanf("%d",&choice2);
    if(choice2==1)
    {
        printf("Enter the item at front-end : ");
            scanf("%d",&item1);
        second=IF(second,item1);
    }
    if(choice2==2)
    {
        printf("Enter the item at rear-end : ");
            scanf("%d",&item1);
        second=IR(second,item1);
    }
}
first=concatenate(first,second);
printf("\nThe two lists are concatenated.");
break;
```

```
            case 6: display(first);
                    break;
            default:exit(0);
                    break;
        }
    }
}
```





LAB PROGRAM-8: WAP to implement Stack & Queues using Linked Representation.

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

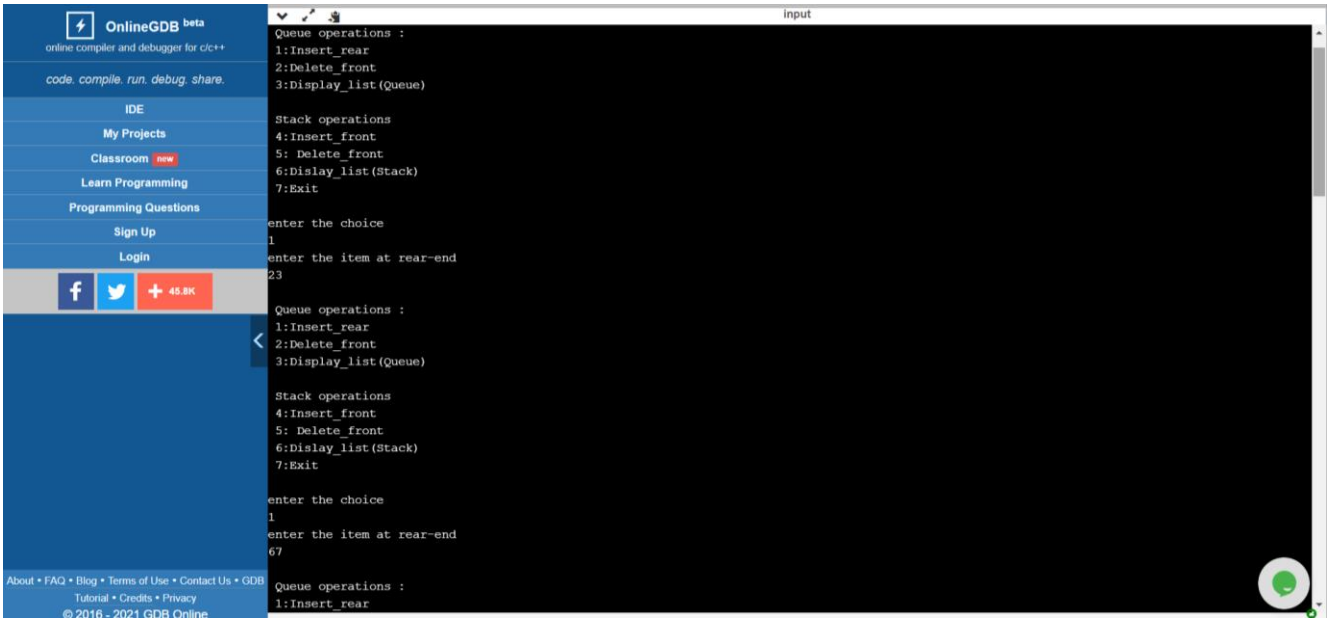
```
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
void display(NODE first)
{
NODE temp;
if(first==NULL)
printf("list empty cannot display items\n");
for(temp=first;temp!=NULL;temp=temp->link)
{
printf("%d \n",temp->info);
```

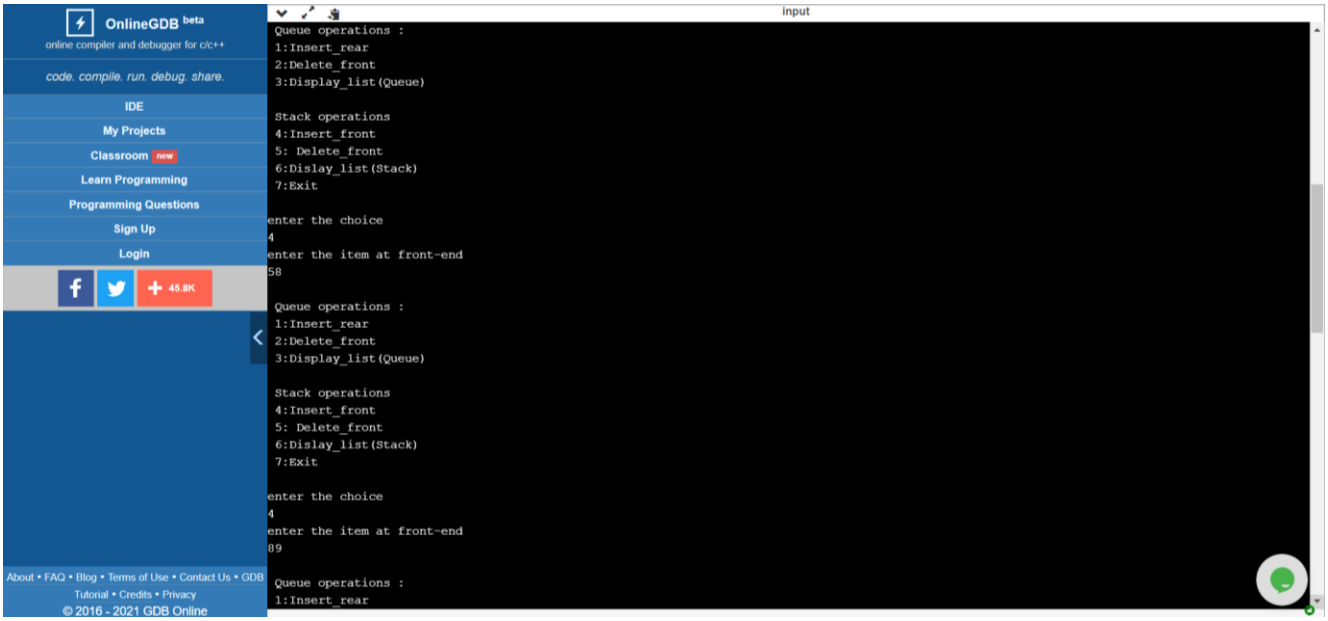
```
}
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}
NODE delete_front_s(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("stack is empty cannot delete\n");
return first;
}
temp=first;
```

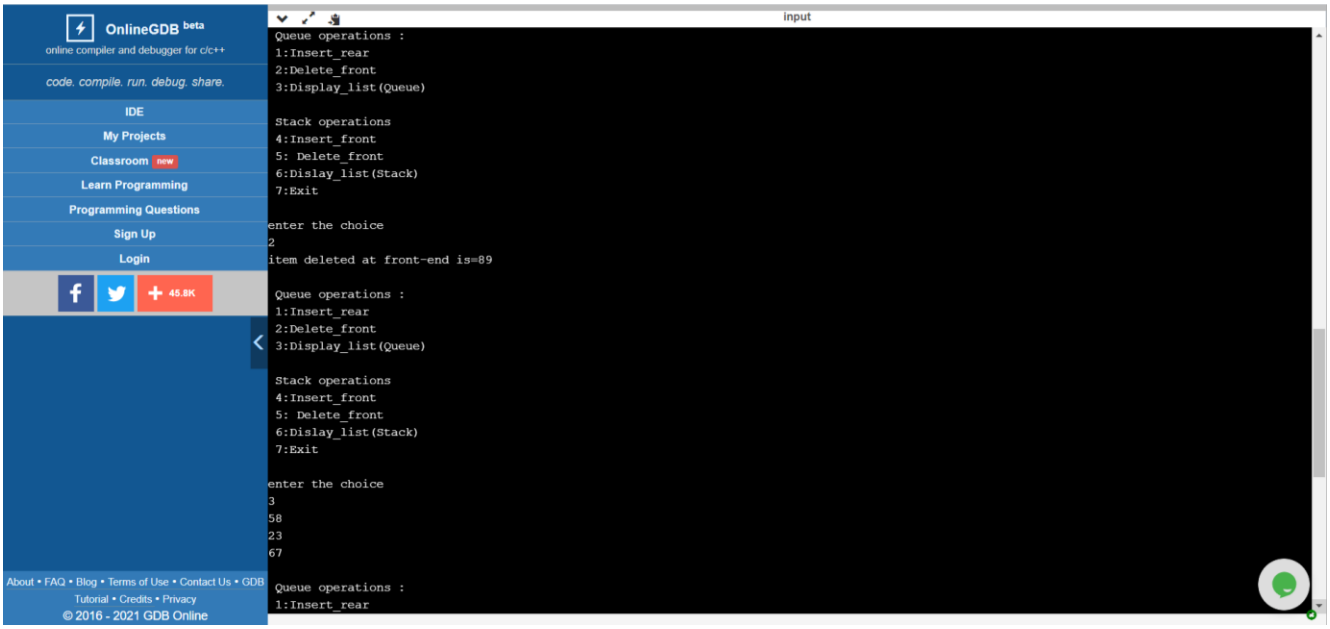
```
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
void display_s(NODE first)
{
NODE temp;
if(first==NULL)
printf("stack empty cannot display items\n");
for(temp=first;temp!=NULL;temp=temp->link)
{
printf("%d\n",temp->info);
}
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
```

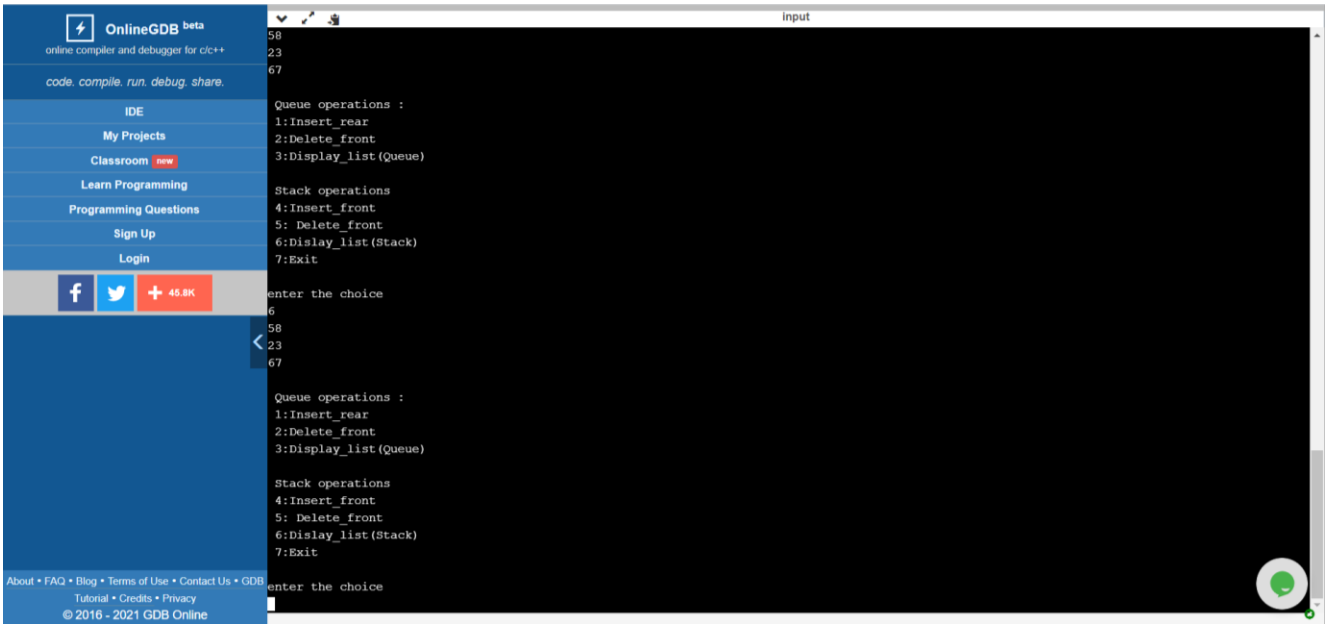
```
printf("\n Queue operations :\n 1:Insert_rear\n 2:Delete_front\n
3:Display_list(Queue)\n \n Stack operations \n 4:Insert_front\n 5:
Delete_front \n 6:Dislay_list(Stack)\n 7:Exit \n \n");

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;

case 4:printf("enter the item at front-end\n");

scanf("%d",&item);

first=insert_front(first,item);

break;

case 5:first=delete_front_s(first);

break;

case 6:display_s(first);

break;

default:exit(0);
```

```
break;
}}
return 0;
}
```









LAB PROGRAM-9: WAP Implement doubly link list with primitive operations a) a) Create a doubly linked list. b) Insert a new node to the left of the node. b) c) Delete the node based on a specific value. c) Display the contents of the list.

```
#include<stdio.h>

#include<stdlib.h>
```

```
struct node
{
 int info;
 struct node *llink;
 struct node *rlink;
 };
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 dinsert_front(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
cur=head->rlink;
head->rlink=temp;
temp->llink=head;
temp->rlink=cur;
cur->llink=temp;
return head;
}
NODE dinsert_rear(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
cur=head->llink;
head->llink=temp;
temp->rlink=head;
```

```
temp->llink=cur;
cur->rlink=temp;
return head;
}
NODE ddelete_front(NODE head)
{
NODE cur,next;
if(head->rlink==head)
{
printf("dq empty\n");
return head;
}
cur=head->rlink;
next=cur->rlink;
head->rlink=next;
next->llink=head;
printf("the node deleted is %d",cur->info);
freenode(cur);
return head;
}
NODE ddelete_rear(NODE head)
```

```
{
NODE cur,prev;
if(head->rlink==head)
{
printf("dq empty\n");
return head;
}
cur=head->llink;
prev=cur->llink;
head->llink=prev;
prev->rlink=head;
printf("the node deleted is %d",cur->info);
freenode(cur);
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 insert_rightpos(int item,NODE head)
{
NODE temp,cur,next;
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;
}
next=cur->rlink;
printf("enter towards right of %d=",item);
temp=getnode();
scanf("%d",&temp->info);
cur->rlink=temp;
temp->llink=cur;
next->llink=temp;
temp->rlink=next;
return head;
}
NODE search(NODE head,int item)
{
NODE temp,cur;
int flag=0;
if(head->rlink==head)
{
printf("list empty\n");
return head;
}
```

```
cur=head->rlink;
while(cur!=head)
{
if(item==cur->info)
{
    flag=1;
    break;
}
cur=cur->rlink;
}
if(cur==head)
printf("search unsuccessfull\n");
if(flag==1)
printf("search successfull\n");
}
NODE delete_all_key(int item,NODE head)
{
    NODE prev,cur,next;
    int count;
    if(head->rlink==head)
    {
```

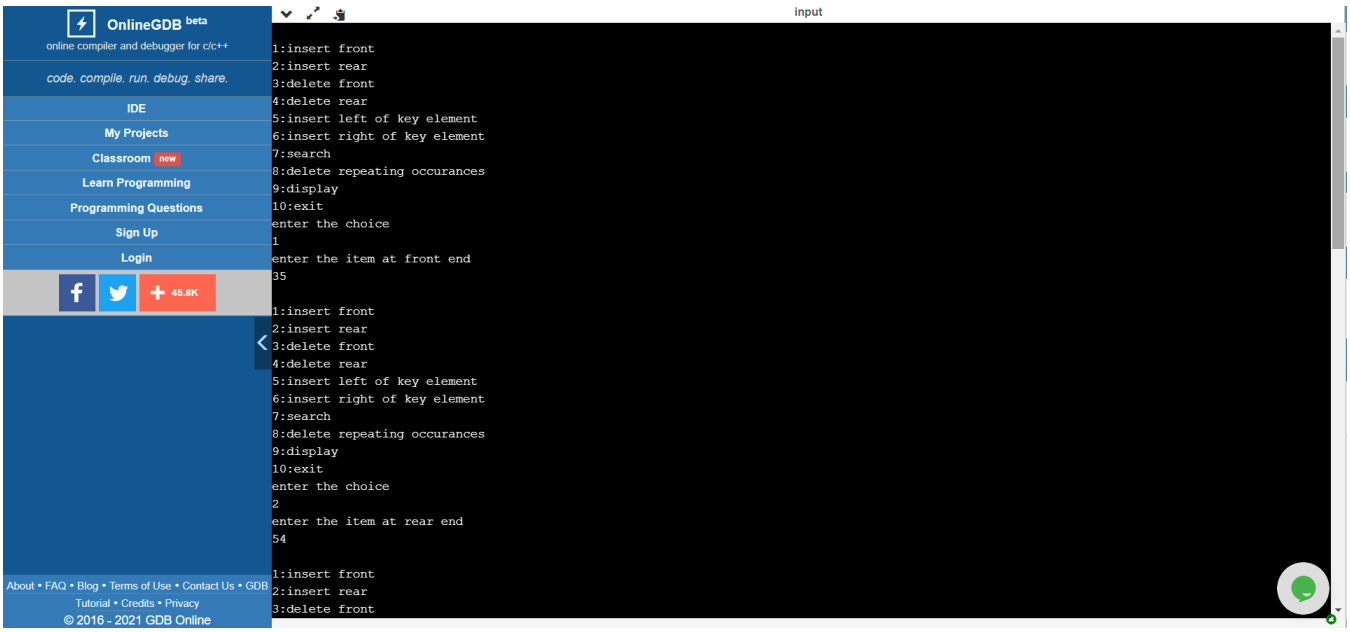
```
    printf("list empty\n");
    return head;
}
count=0;
cur=head->rlink;
while(cur!=head)
{
    if(item!=cur->info)
    cur=cur->rlink;
    else
    {
        count++;
        prev=cur->llink;
        next=cur->rlink;
        prev->rlink=next;
        next->llink=prev;
        freenode(cur);
        cur=next;
    }
}
if(count==0)
```

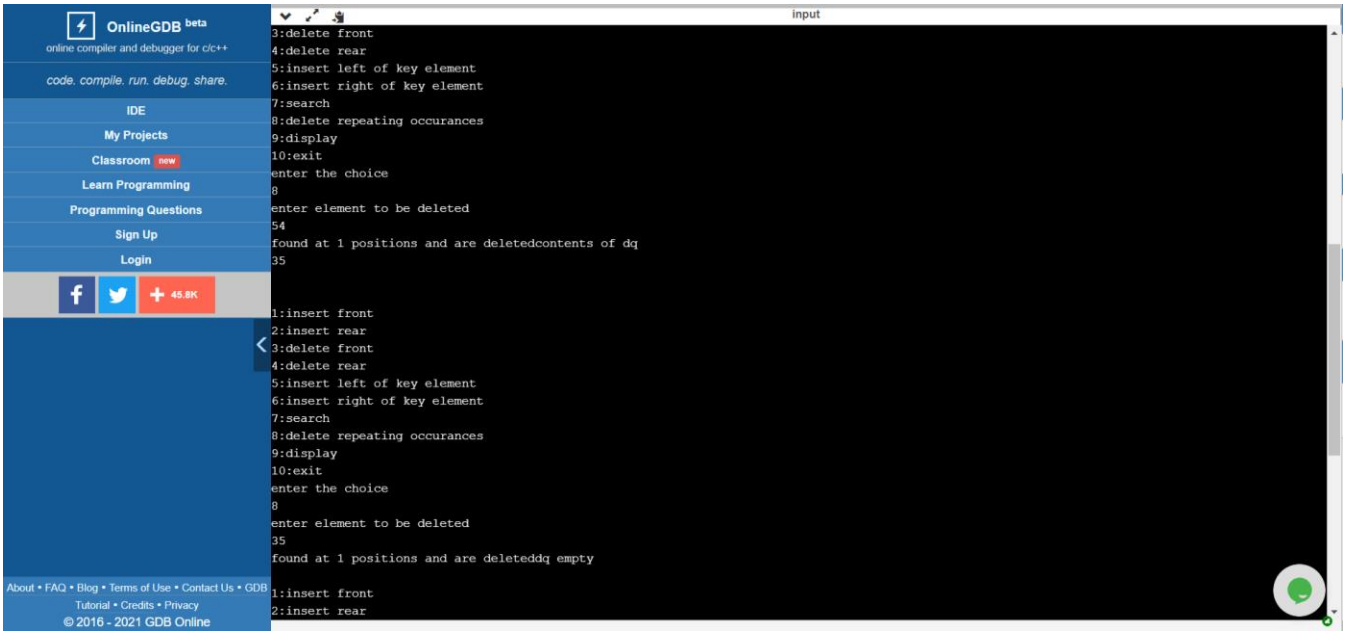
```
    printf("not found\n");
    else{
    printf("found at %d positions and are deleted",count);
      return head;
    }
}

void display(NODE head)
{
NODE temp;
if(head->rlink==head)
{
printf("dq empty\n");
return;
}
printf("contents of dq\n");
temp=head->rlink;
while(temp!=head)
{
printf("%d\n",temp->info);
temp=temp->rlink;
```

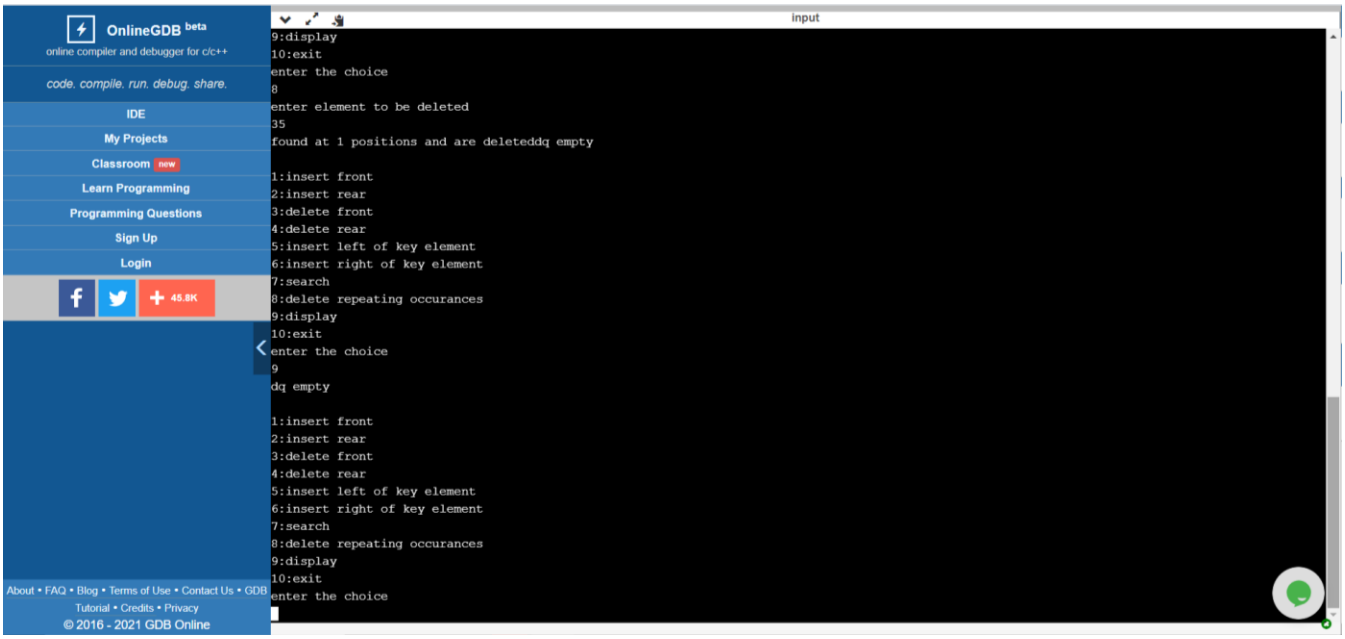
```
}
printf("\n");
}
void main()
{
NODE head,last;
int item, choice;
head=getnode();
head->rlink=head;
head->llink=head;

for(;;)
{
printf("\n1:insert front\n2:insert rear\n3:delete front\n4:delete
rear\n5:insert left of key element\n6:insert right of key
element\n7:search\n8:delete repeating
occurances\n9:display\n10: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);
last=dinsert_front(item,head);
break;
case 2: printf("enter the item at rear end\n");
scanf("%d",&item);
last=dinsert_rear(item,head);
break;
case 3:last=ddelete_front(head);
break;
case 4: last=ddelete_rear(head);
break;

case 5:
    printf("enter the key element\n");
    scanf("%d",&item);
    last=insert_leftpos(item,head);
    break;
 case 6:
    printf("enter the key element\n");
    scanf("%d",&item);
    last=insert_rightpos(item,head);
```

```
    break;
case 7:
        printf("enter the search element\n");
        scanf("%d",&item);
        search(head,item);
        break;
case 8: printf("enter element to be deleted\n");
        scanf("%d",&item);
        last=delete_all_key(item,head);
case 9: display(head);
break;
default:exit(0);
}
}
}
```


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
1:insert front
2:insert rear
3:delete front
4:delete rear
5:insert left of key element
6:insert right of key element
7:search
8:delete repeating occurances
9:display
10:exit
enter the choice
1
enter the item at front end
35
1:insert front
2:insert rear
3:delete front
4:delete rear
5:insert left of key element
6:insert right of key element
7:search
8:delete repeating occurances
9:display
10:exit
enter the choice
2
enter the item at rear end
54
1:insert front
2:insert rear
3:delete front


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
3:delete front
4:delete rear
5:insert left of key element
6:insert right of key element
7:search
8:delete repeating occurances
9:display
10:exit
enter the choice
8
enter element to be deleted
54
found at 1 positions and are deletedcontents of dq
35
1:insert front
2:insert rear
3:delete front
4:delete rear
5:insert left of key element
6:insert right of key element
7:search
8:delete repeating occurances
9:display
10:exit
enter the choice
8
enter element to be deleted
35
found at 1 positions and are deleteddq empty
1:insert front
2:insert rear



LAB 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, preorder and post order c) To display the elements in the tree.

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct node

{

int info;

struct node*llink;

struct node*rlink;

};

typedef struct node*NODE;

NODE getnode()

{
```

```
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("Memory not available!");
exit(0);
}
return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert(int item,NODE root)
{
NODE temp,cur,prev;
char direction[10];
int i;
temp=getnode();
temp->info=item;
temp->llink=NULL;
temp->rlink=NULL;
if(root==NULL)
```

```
return temp;
printf("Give direction to insert..\n");
scanf("%s",direction);
prev=NULL;
cur=root;
for(i=0;i<strlen(direction)&&cur!=NULL;i++)
{
prev=cur;
if(direction[i]=='l')
cur=cur->llink;
else
cur=cur->rlink;
}
if(cur!=NULL||i!=strlen(direction))
{
printf("Insertion not possible\n");
freenode(temp);
return(root);
}
if(cur==NULL)
{
if(direction[i-1]=='l')
prev->llink=temp;
```

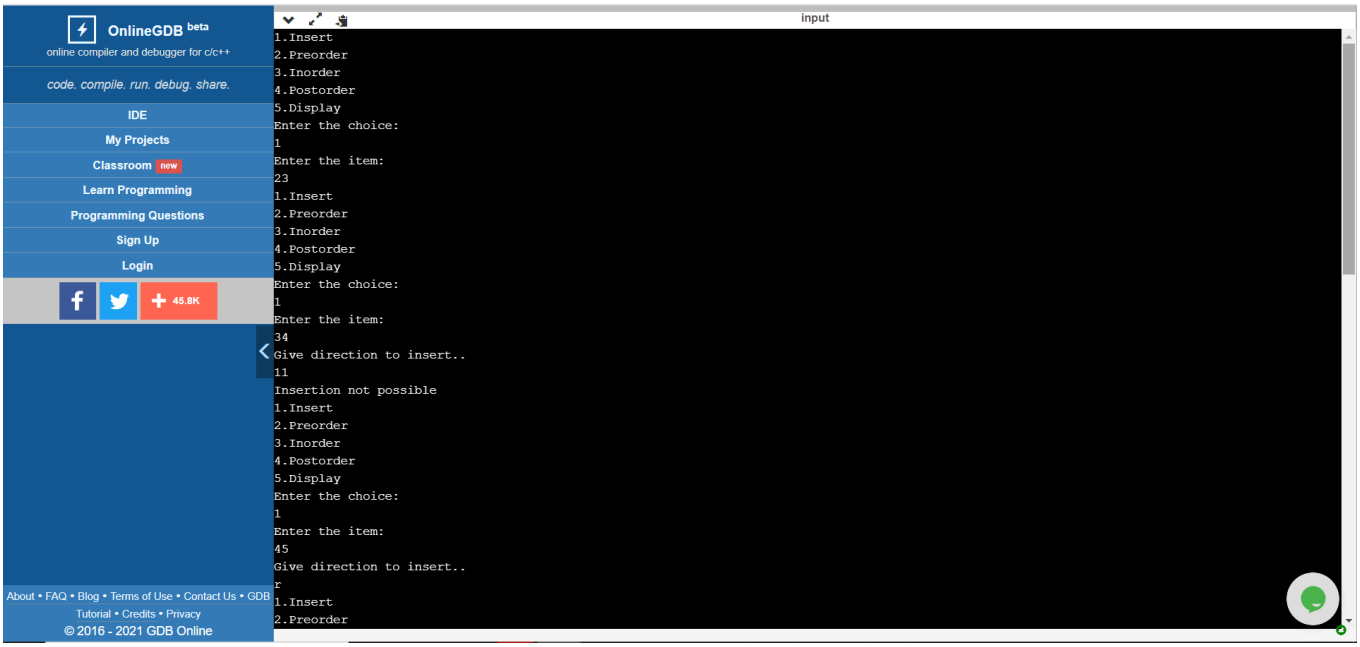
```
else
prev->rlink=temp;
}
return(root);
}
void preorder(NODE root)
{
if(root!=NULL)
{
printf("the item is %d\n",root->info);
preorder(root->llink);
preorder(root->rlink);
}
}
void inorder(NODE root)
{
if(root!=NULL)
{
inorder(root->llink);
printf("The item is%d\n",root->info);
inorder(root->rlink);
}
}
```

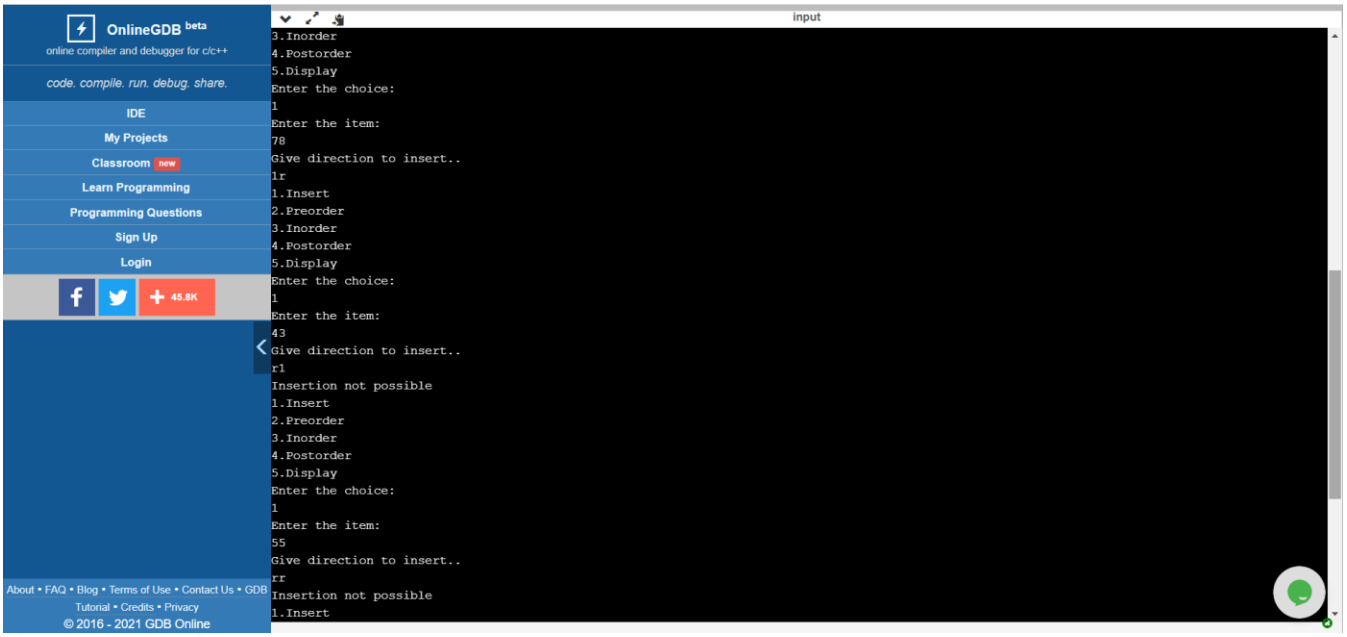
```
void postorder(NODE root)
{
if (root!=NULL)
{
postorder(root->llink);
postorder(root->rlink);
printf("The item is%d\n",root->info);
}
}
void display(NODE root,int i)
{
int j;
if(root!=NULL)
{
display(root->rlink,i+1);
for (j=1;j<=i;j++)
printf("  ");
printf("%d\n",root->info);
display(root->llink,i+1);
}
}

int main()
```

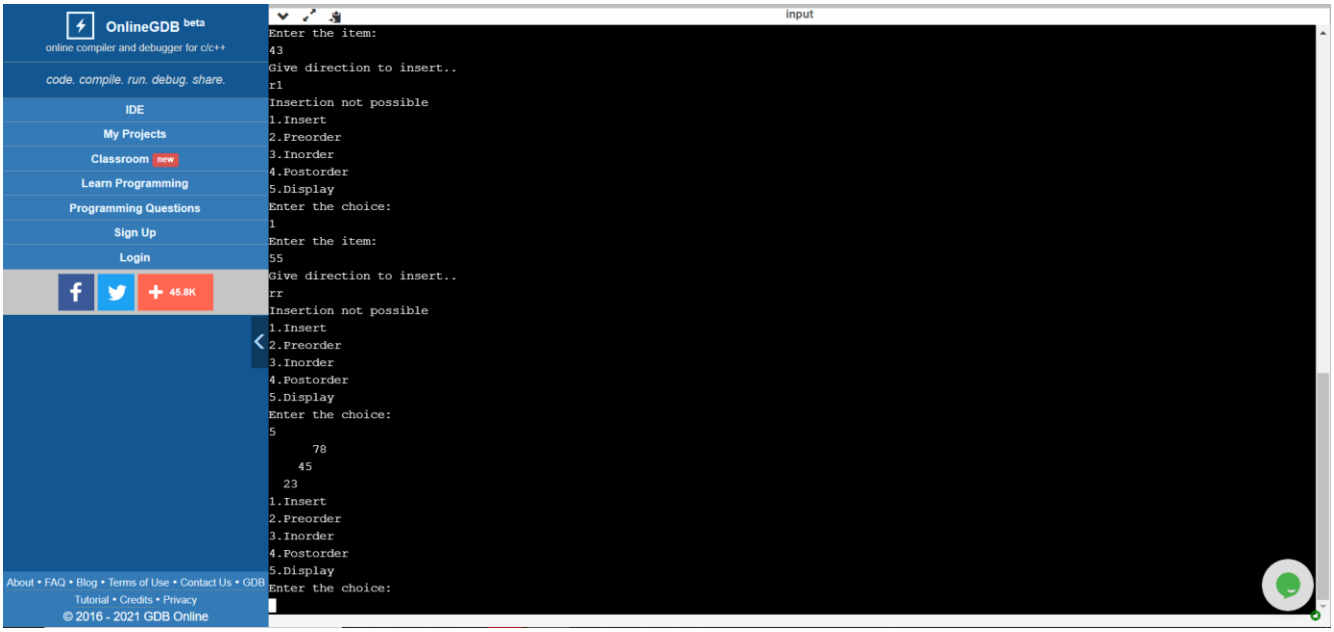
```
{
NODE root=NULL;
int choice,i,item;

for(;;)
{
printf("1.Insert\n2.Preorder\n3.Inorder\n4.Postorder\n5.Display\n");
printf("Enter the choice:\n");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("Enter the item:\n");
scanf("%d",&item);
root=insert(item,root);
break;
case 2: if(root==NULL)
{
 printf("Tree is empty!");
}
else
{
 printf("Given tree is..");
 display(root,1);
```

```
 printf("The preorder traversal is:\n");
 preorder(root);
}
break;
case 3:if(root==NULL)
   {
printf("Tree is empty");
   }
   else
   {
printf("Given tree is..");
display(root,1);
printf("The inorder traversal is \n");
inorder(root);
}
   break;
case 4:if (root==NULL)
{
printf("Tree is empty");
 }
   else
   {
printf("Given tree is..");
```

```
display(root,1);
printf("The postorder traversal is \n");
postorder(root);
   }
  break;
case 5:display(root,1);
   break;
default:printf("Invalid choice entered.\n");
        exit(0);
}
}
return 0;
}
```




OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
78
Give direction to insert..
lr
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
43
Give direction to insert..
r1
Insertion not possible
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
55
Give direction to insert..
rr
Insertion not possible
1.Insert


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
45.8K
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2021 GDB Online
input
Enter the item:
43
Give direction to insert..
r1
Insertion not possible
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
55
Give direction to insert..
rr
Insertion not possible
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
5
78
45
23
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice: