**Term work**

**on**

**Data Structure with C**

**(PCS 302)**

**2021-22**

**Submitted to: Submitted by:**

Mr. Rakesh Patra ANSHIKA MAHESHWARI

Assistant Professor University Roll. No: 2018807 GEHU, Dehradun Class Roll. No: 07 / Section:A

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GRAPHIC ERA HILL UNIVERSITY, DEHRADUN**

**ACKNOWLEDGMENT**

I would like to particularly thank my Data Science using Data Structure with C Lab Faculty **Mr. Rakesh Patra** for his patience, support and encouragement throughout the completion of this Term work.

At last but not the least I greatly indebted to all other persons who directly or indirectly helped me during this course.

**ANSHIKA MAHESHWARI**

**University. Roll No.- 2018187**

**B.Tech CSE-A-III Sem**

**Session: 2021-22**

**GEHU, Dehradun**



Table of Contents

|  |  |  |
| --- | --- | --- |
| **Program No.** | **Program Name** | **Page No** |
| **1.** | Write a the C program to create an array by inserting N elements in it then find second non repeating element from the array. |  |
| **2.** | Write a the C program to create an array by inserting N elements in it then find third repeating element from the array. |  |
| **3.** | Write a C program Create a Dynamic array and then Reverse the array using recursion and then finally print the array. |  |
| 4. | Write a C Program implement STACK using array in menu driven form |  |
| **5.** | Write a C Program to Convert Infix to Postfix Expression using Stack. |  |
| **6.** | Write a C Program to create singly linked list by adding nodes in the right hand side and delete alternate node from the list and then print the final list. |  |
| **7.** | Write a C Program implement STACK using Link List in menu driven form. |  |
| **8.** | Write a C Program implement QUEUE using Link List in menu driven form. |  |
| **9.** | Write a C Program implement priority QUEUE using array in menu driven form. |  |
| **10.** | Write a C Program implement QUEUE using array in menu driven form. |  |
| **11.** | Write a C program to Evaluate Postfix Expression using Stack |  |
| **12.** | Write a C program to create TWO singly linked list L1 and L2 and sort both the list and finally merge both the list such that L2 comes after L1.[ use double pointer] |  |
| **13.** | Write C program to create a doubly link list by adding the node right hand side and then check list is in palindrome form or not. |  |
| **14.** | Write a C program to create a circular link list by adding the nodes in right hand side and then print the list. |  |

1. Write a C program to create an array by inserting N elements in it then find second non repeating element from the array.

**Source Code :**

#include<stdio.h>

int main()

{

int n, c = 0;

printf("Enter the size of array : ");

scanf("%d", &n);

int a[n];

printf("Enter the elements : \n");

for(int i=0; i<n; i++)

{

scanf("%d", &a[i]);

}

for(int i=0;i<n; i++)

{

int f=0;

for(int j=0;j<n; j++)

{

if(a[i]==a[j] &&i!=j)

{

f=1;

break;

}

}

if(f==0)

{

c++;

}

if(c==2)

{

printf("Second non repeating element is : %d", a[i]);

break;

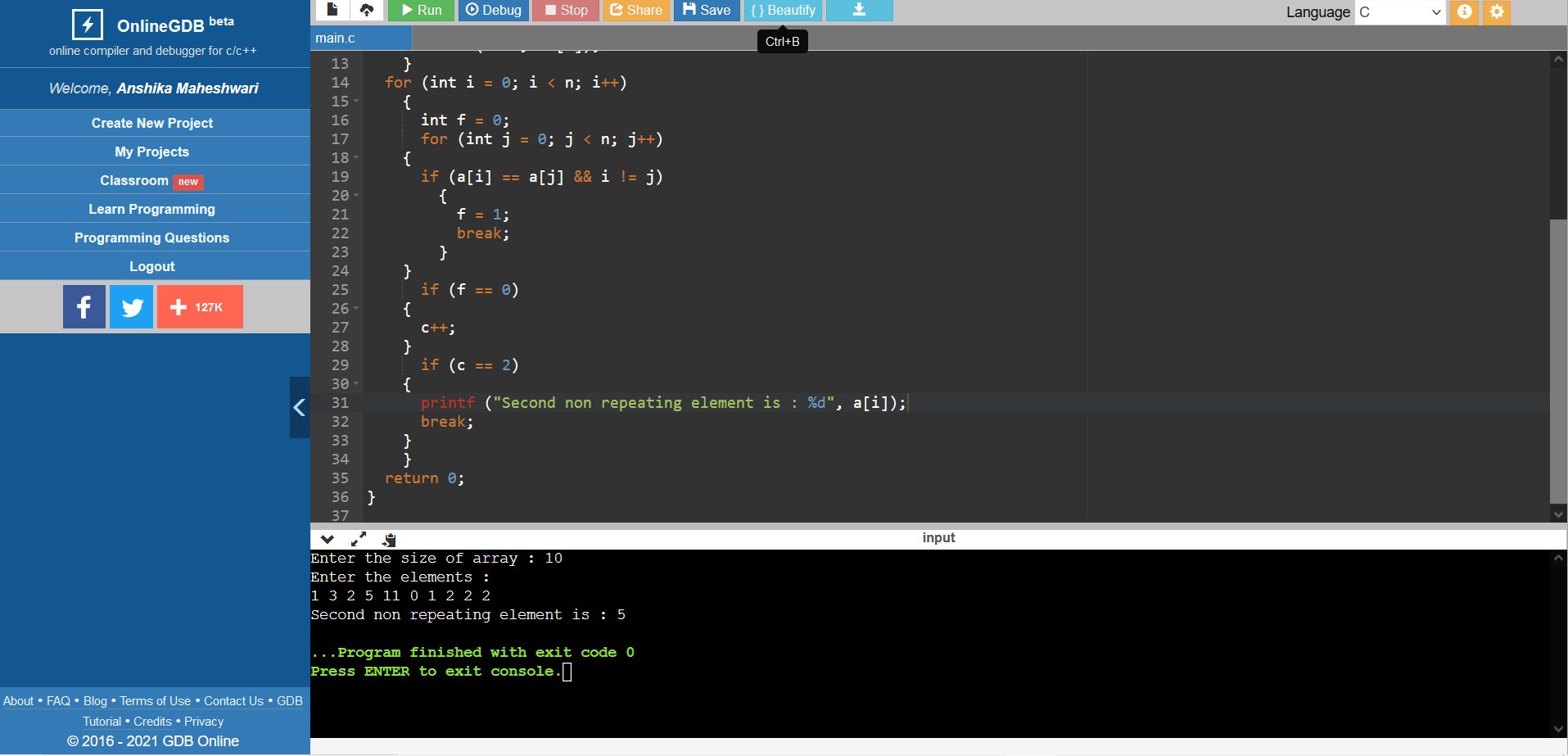
}

}

return 0;

}

**Output**



2.Write a the C program to create an array by inserting  N elements in it then find third repeating element from the array.

**Source Code :**

#include<stdio.h>

int thirdrepeat(int a[],int n,int k)

{int c=0;

int in;

int arr[n];

for(int i=0;i<n;i++)

{

for(int j=i+1;j<n;j++)

{

if(a[i]==a[j])

{

arr[i]=-1;

a[j]=0;

c=1;

}

if(c==0)

{

arr[i]=a[i];

}

}

}

for(int i=0;i<n;i++)

{

if(arr[i]==-1)

{

in=i;

k--;

if(k==0)

return in;

}

}

}

int main()

{

int n,k;

scanf("%d%d",&n,&k);

int a[n];

for(int i=0;i<n;i++)

{

scanf("%d",&a[i]);

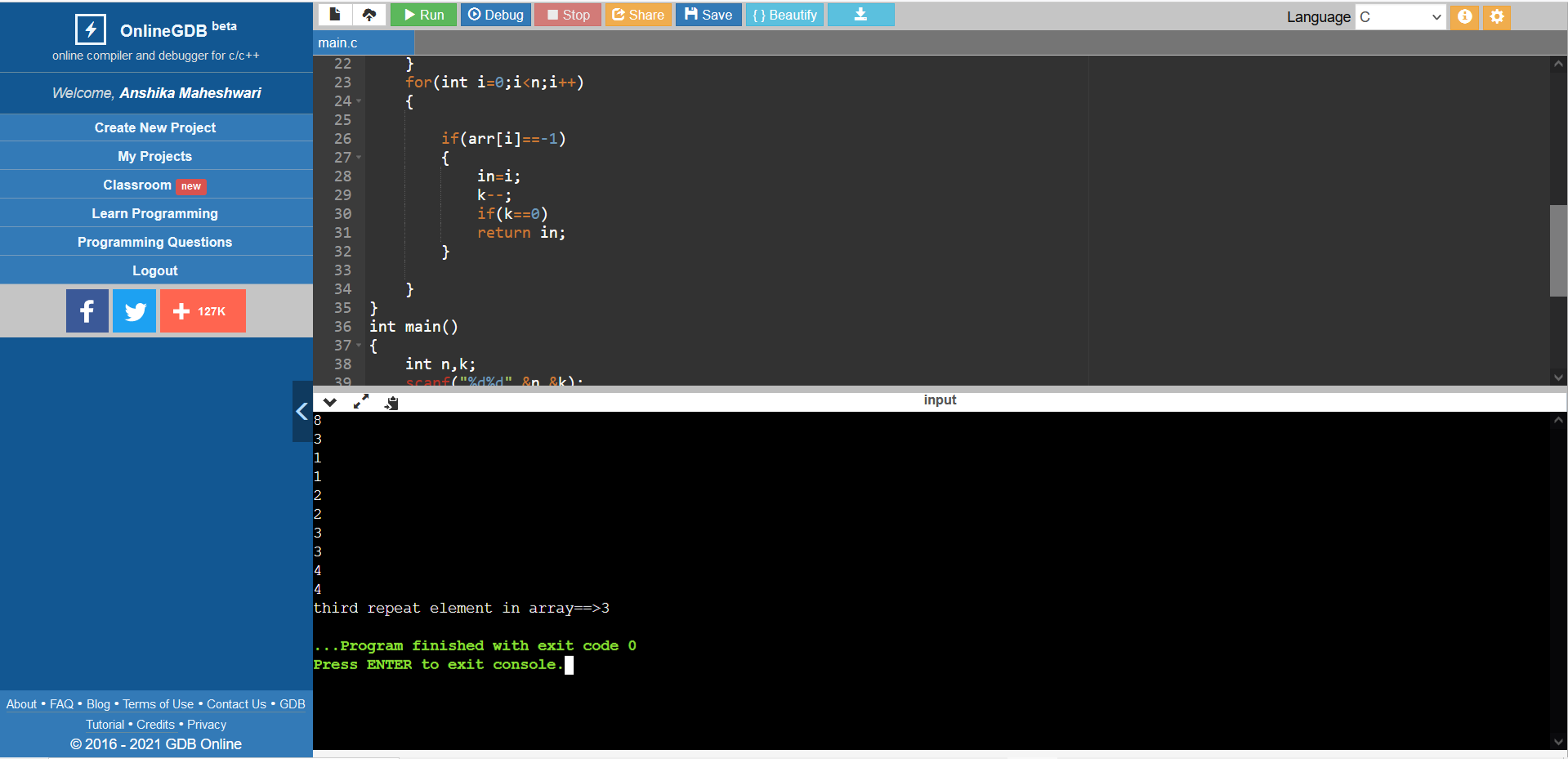
}

int ans=thirdrepeat(a,n,k);

printf("third repeat element in array==>%d",a[ans]);

}

**Output**



3.Write a C program create a Dynamic array and then Reverse the array using recursion and then finally print the array.

**Source Code :**

#include<stdio.h>

#include<stdlib.h>

void swap(int \*a, int \*b)

{

\*a = \*a+\*b;

\*b = \*a-\*b;

\*a = \*a-\*b;

}

void reverse(int \*arr,int n)

{

int i,j = n-1;

for(i = 0; i<n/2; i++)

{

swap(&arr[i],&arr[j]);

j--;

}

}

int main()

{

int n;

printf("Enter Size of the array : ");

scanf("%d",&n);

int \*arr = malloc(n \* sizeof(int));

printf("Enter elements in the array : \n");

for(int i = 0; i<n; i++)

scanf("%d",&arr[i]);

reverse(arr,n);

printf("\nReverse Array is : \n");

for(int i = 0; i<n; i++)

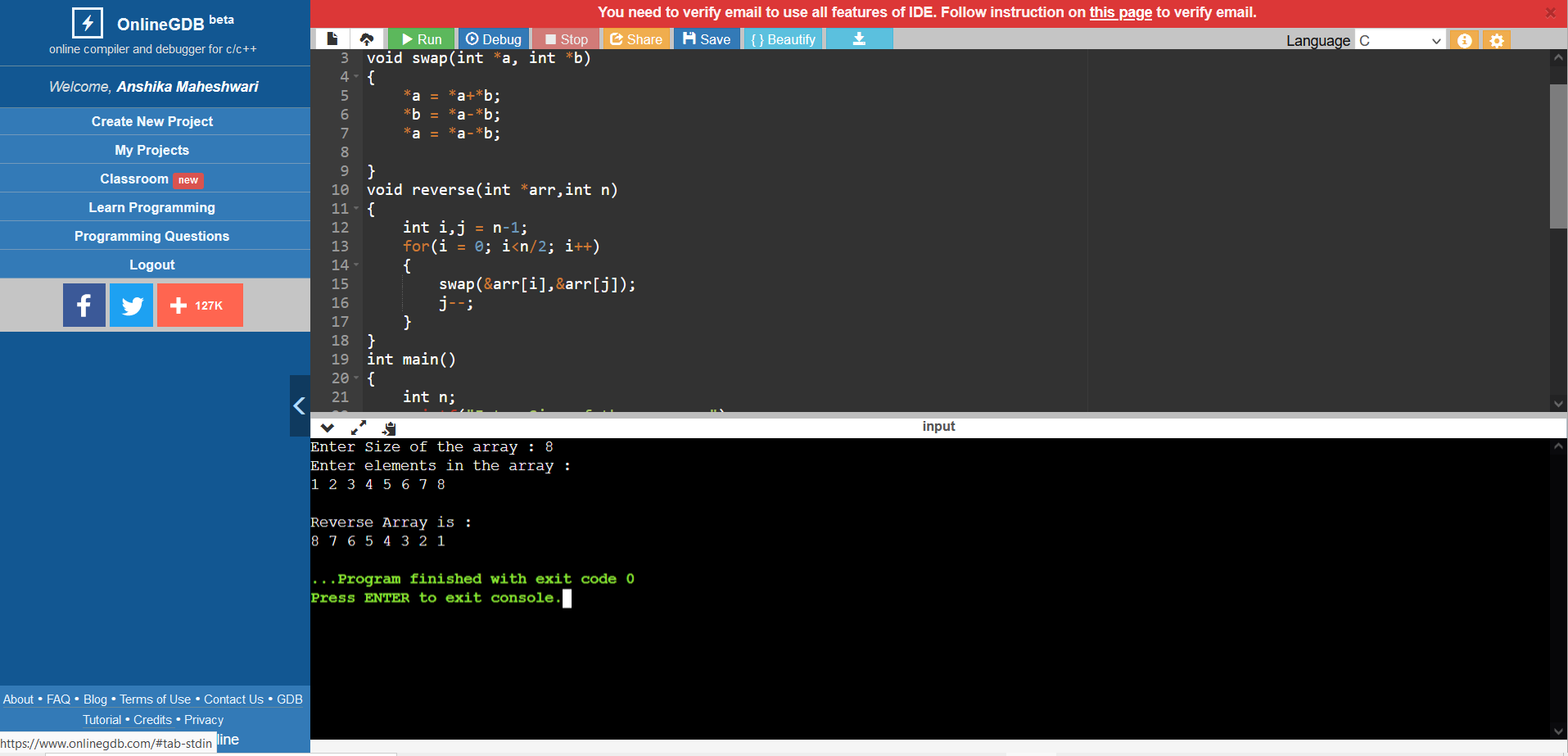
printf("%d ",arr[i]);

free(arr);

return 0;

}

**Output**



4. Write a C program implement STACK using array in menu driven form.

**Source Code :**

|  |
| --- |
| #include<stdlib.h> |
|  |

|  |
| --- |
| #include<stdio.h> |
|  |

|  |
| --- |
| #define MAX 10 |
|  |

|  |
| --- |
| int stack[MAX], top=-1; |
|  |

|  |
| --- |
| void push(int n) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if(top==MAX-1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| printf("\nInsufficient space ! Stack is full\n"); |
|  |

|  |
| --- |
| return; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| top++; |
|  |

|  |
| --- |
| stack[top]=n; |
|  |

|  |
| --- |
| printf("Element added successfully."); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void pop() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if(top==-1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| printf("Stack is empty !"); |
|  |

|  |
| --- |
| return ; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| int pop\_element=stack[top]; |
|  |

|  |
| --- |
| top--; |
|  |

|  |
| --- |
| printf("Deleted element %d from stack!\n",pop\_element); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void display() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if(top==-1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| printf("Stack is empty !"); |
|  |

|  |
| --- |
| return ; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| printf("\n"); |
|  |

|  |
| --- |
| for(int i=top;i>=0;i--) |
|  |

|  |
| --- |
| printf("%d\t",stack[i]); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void peek() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| if(top==-1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| printf("Stack is empty !"); |
|  |

|  |
| --- |
| return ; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| printf("\nTop element is : %d\n",stack[top]); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| int main() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| int choice, item; |
|  |

|  |
| --- |
| while(1) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| printf(" MENU "); |
|  |

|  |
| --- |
| printf("\n1.Push element to stack\t\t2.Pop element from stack\n"); |
|  |

|  |
| --- |
| printf("3.Display the stack\t\t\t4.Peek\t\t\t5.Exit\n"); |
|  |

|  |
| --- |
| printf("Enter your choice : "); |
|  |

|  |
| --- |
| scanf("%d",&choice); |
|  |

|  |
| --- |
| switch(choice) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| case 1 : printf("\nEnter element : "); |
|  |

|  |
| --- |
| scanf("%d",&item); |
|  |

|  |
| --- |
| push(item); |
|  |

|  |
| --- |
| break; |
|  |

|  |
| --- |
| case 2 : printf("\n"); |
|  |

|  |
| --- |
| pop(); |
|  |

|  |
| --- |
| break; |
|  |

|  |
| --- |
| case 3 : printf("\nDisplaying the stack ...\n"); |
|  |

|  |
| --- |
| display(); |
|  |

|  |
| --- |
| break; |
|  |

|  |
| --- |
| case 4 : printf("\nDisplaying the top of the stack...\n"); |
|  |

|  |
| --- |
| peek(); |
|  |

|  |
| --- |
| break; |
|  |

|  |
| --- |
| case 5 : printf("\nexiting...\n"); |
|  |

|  |
| --- |
| exit(0); |
|  |

|  |
| --- |
| default: printf("\nINVALID CHOICE !!!\n"); |
|  |

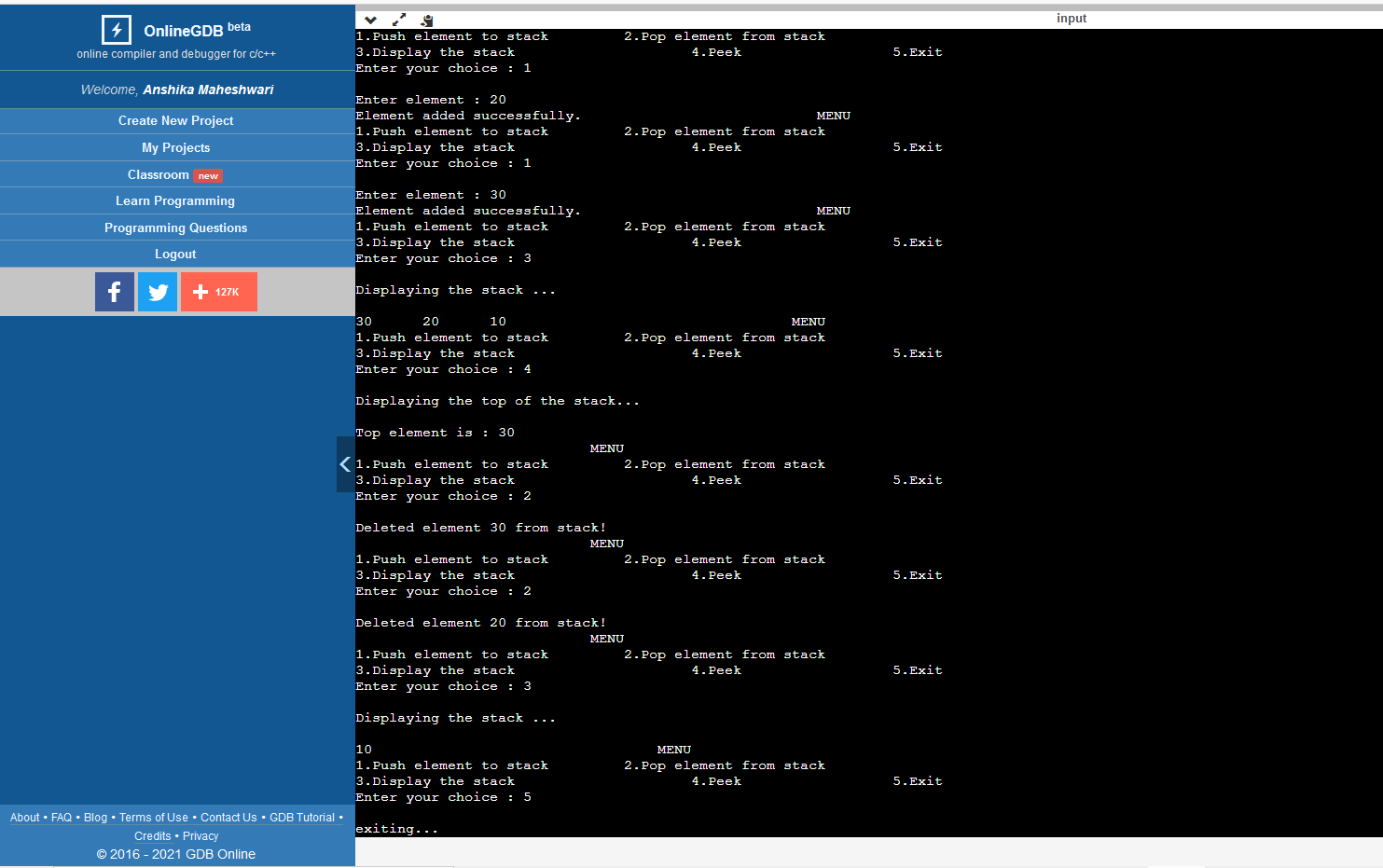
|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| return 0; |
|  |

}

**Output**



5. Write a C program to convert Infix to Postfix Expression using Stack.

**Source Code :**

#include<stdio.h>

#include<string.h>

int stack[1000];

int top=-1;

void push(char ch)

{

top++;

stack[top]=ch;

}

void pop()

{

if(top==-1)

printf("underflow");

else top--;

}

int precedence(char ch)

{

if(ch=='^')

return 3;

if(ch=='\*' || ch=='/')

{

return 2;

}

if(ch=='+' || ch=='-')

{

return 1;

}

else return -1;

}

int main()

{

char str[100];

scanf("%s\n",&str);

int l=strlen(str);

printf("%d",l);

printf(" %s\n",str);

int c=0;

for(int i=0;i<l;i++)

{ if(str[i]>=97 && str[i]<=122) {

printf("%c",str[i]);

}

else if(top==-1 && c==0)

{

c=1;

push(str[i]);

}

else if(str[i]=='(')

{

push(str[i]);

}

else if(str[i]==')')

{

while(stack[top]!='(')

{

printf("%c",stack[top]);

pop();

}

pop();

}

else if(precedence(stack[top])==precedence(str[i]))

{

printf("%c",stack[top]);

pop();

push(str[i]);

}

else if(precedence(stack[top])<precedence(str[i]))

{

push(str[i]);

}

else {

while(precedence(stack[top])>precedence(str[i]))

{

printf("%c",stack[top]);

pop();

}

push(str[i]);

}

}

while(top!=-1){

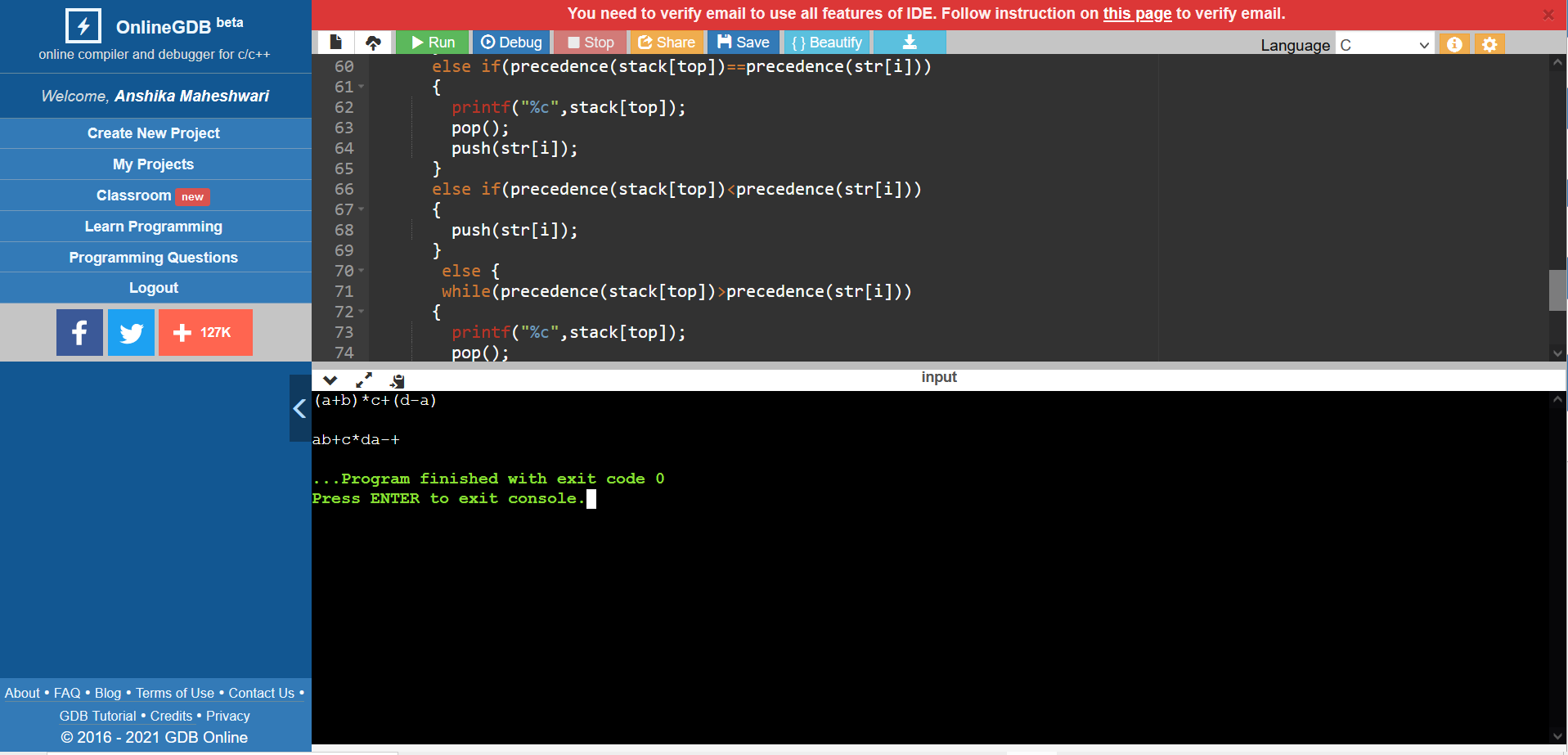
printf("%c",stack[top]);

pop();

}

}

**Output**



6. Write a C Program to create singly linked list by adding nodes in the right hand side and delete alternate node from the list and then print the final list.

**Source Code :**

#include<stdio.h>

#include<stdlib.h>

struct Node

{

int data;

struct Node \*next;

};

void deleteAlt(struct Node \*head)

{

if (head == NULL)

return;

struct Node \*prev = head;

struct Node \*node = head->next;

while (prev != NULL && node != NULL)

{

prev->next = node->next;

free(node);

prev = prev->next;

if (prev != NULL)

node = prev->next;

}

}

void push(struct Node\*\* head\_ref, int new\_data)

{

struct Node\* new\_node = (struct Node\*) malloc(sizeof(struct Node));

new\_node->data = new\_data;

new\_node->next = (\*head\_ref);

(\*head\_ref) = new\_node;

}

void printList(struct Node \*node)

{

while (node != NULL)

{

printf("%d ", node->data);

node = node->next;

}

}

int main()

{

struct Node\* head = NULL;

push(&head, 10);

push(&head, 9);

push(&head, 8);

push(&head, 7);

push(&head, 6);

push(&head, 5);

push(&head, 4);

push(&head, 3);

push(&head, 2);

push(&head, 1);

printf("List before deleting Alternate Nodes : \n");

printList(head);

deleteAlt(head);

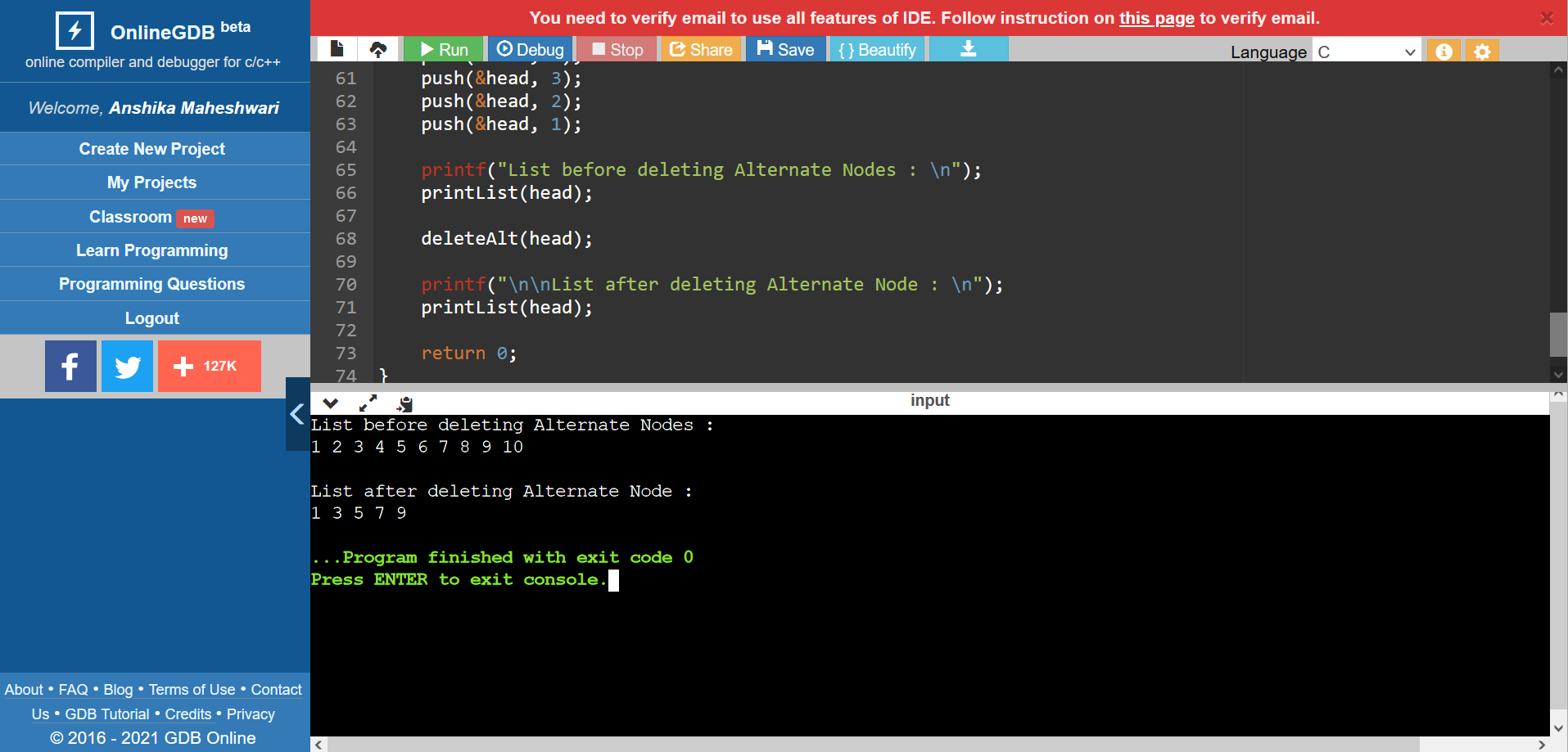
printf("\n\nList after deleting Alternate Node : \n");

printList(head);

return 0;

}

**Output**



7. Write a C Program implement STACK using Linked List in menu driven form.

**Source Code :**

#include<stdio.h>

#include<stdlib.h>

struct Node

{

int data;

struct Node \*link;

};

int isEmpty(struct Node\* top)

{

if(top==NULL)

return 1;

else

return 0;

}

int isFull( struct Node\* top)

{

struct Node\* temp= (struct Node\*)malloc(sizeof(struct Node));

if(temp==NULL){

return 1;

}

else

return 0;

}

struct Node\* push(struct Node \*top, int data)

{

if(top==NULL)

{

top=(struct Node\*)malloc(sizeof(struct Node));

top->data=data;

top->link=NULL;

}

else{

struct Node\* ptr=(struct Node\*)malloc(sizeof(struct Node));

ptr->data=data;

ptr->link=top;

top=ptr;

return top;

}

}

int pop(struct Node\*\* top)

{

if(isEmpty(\*top))

{

printf("Stack is empty! No element can be poppped!!");

}

else

{

struct Node\* temp= \*top;

int x= temp->data;

\*top = (\*top) -> link;

free(temp);

return x;

}

}

void display(struct Node\* ptr)

{

while(ptr!=NULL)

{

printf("%d\t", ptr->data);

ptr=ptr->link;

}

}

int peek( struct Node\* top)

{

return top->data;

}

int main()

{

struct Node\* top=NULL;

int n, choice;

while(1)

{

printf(" MENU ");

printf("\n1.Push element to stack\t\t2.Pop element from stack\n");

printf("3.Display the stack\t\t\t4.Peek\t\t\t5.Exit\n");

printf("Enter your choice : ");

scanf("%d",&choice);

switch(choice)

{

case 1 :printf("\nEnter element : ");

scanf("%d",&n);

top = push(top,n);

break;

case 2 :printf("\n");

int num = pop(&top);

printf("\nDeleted %d from stack\n",num);

break;

case 3 :printf("\nDisplaying the stack ...\n");

display(top);

break;

case 4 :printf("\nDisplaying the top of the stack...");

printf("%d ",peek(top));

break;

case 5 :printf("\nexiting...\n");

exit(0);

default: printf("\nINVALID CHOICE !!!\n");

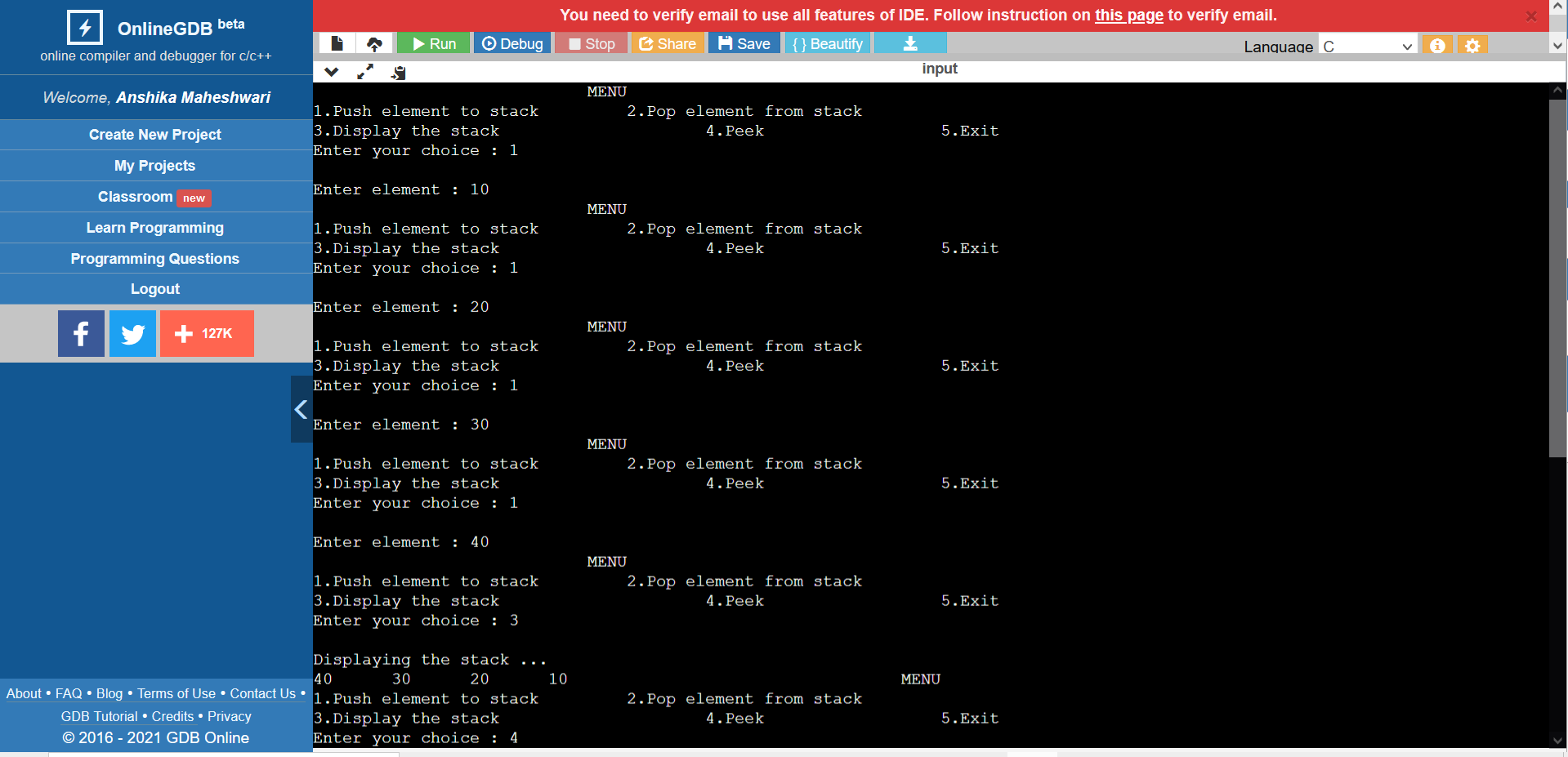
}

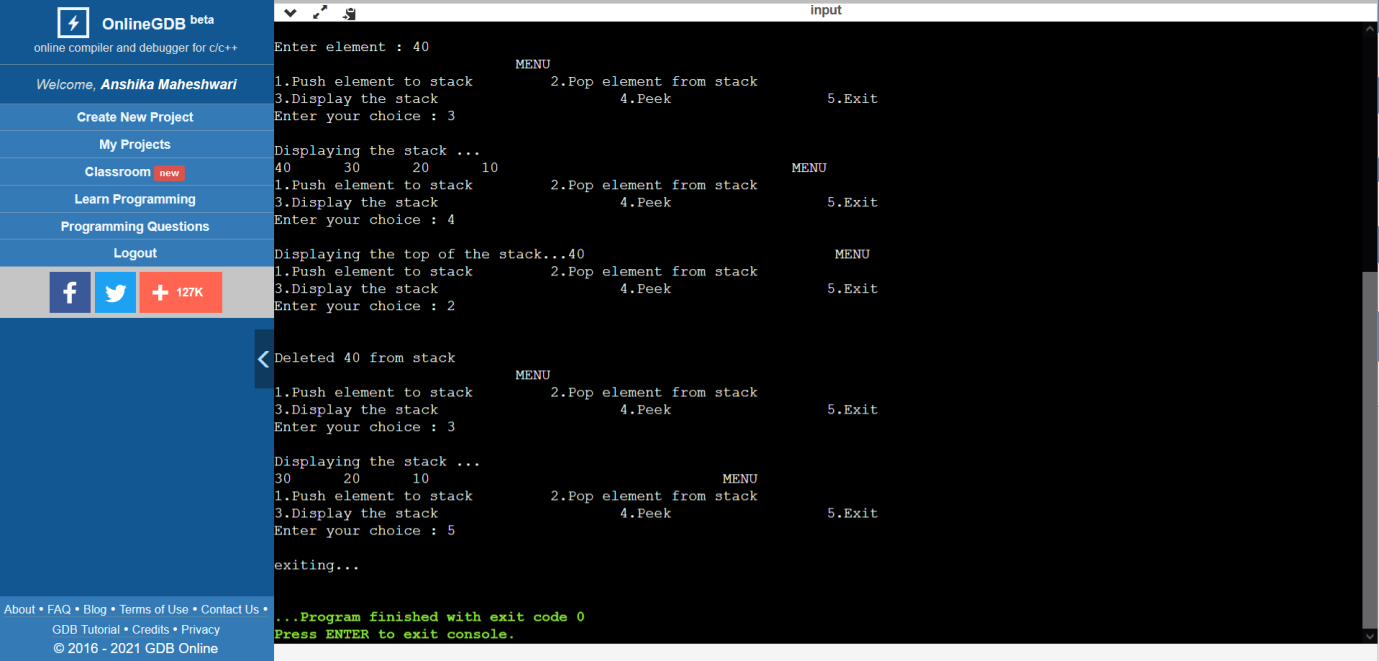
}

return 0;

}

**Output**

****



8. Write a C Program implement QUEUE using Linked List in menu driven form.

**Source Code :**

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*front;

struct node \*rear;

void insert();

void delete();

void display();

void main ()

{

int choice;

while(choice != 4)

{

printf("\n1.insert an element\n2.Delete an element\n3.Display the queue\n4.Exit\n");

printf("Enter your choice : ");

scanf("%d",& choice);

switch(choice)

{

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

printf("Exiting the program!\n");

break;

default:

printf("\nEnter valid choice : ");

}

}

}

void insert()

{

struct node \*ptr;

int item;

ptr = (struct node \*) malloc (sizeof(struct node));

if(ptr == NULL)

{

printf("\nOVERFLOW\n");

return;

}

else

{

printf("\nEnter value : ");

scanf("%d",&item);

ptr -> data = item;

if(front == NULL)

{

front = ptr;

rear = ptr;

front -> next = NULL;

rear -> next = NULL;

}

else

{

rear -> next = ptr;

rear = ptr;

rear->next = NULL;

}

}

}

void delete ()

{

struct node \*ptr;

if(front == NULL)

{

printf("\nUNDERFLOW\n");

return;

}

else

{

ptr = front;

front = front -> next;

free(ptr);

}

}

void display()

{

struct node \*ptr;

ptr = front;

if(front == NULL)

{

printf("\nQueue is Empty!\n");

}

else

{ printf("\nPrinting values ...\n");

while(ptr != NULL)

{

printf("\n%d\n",ptr -> data);

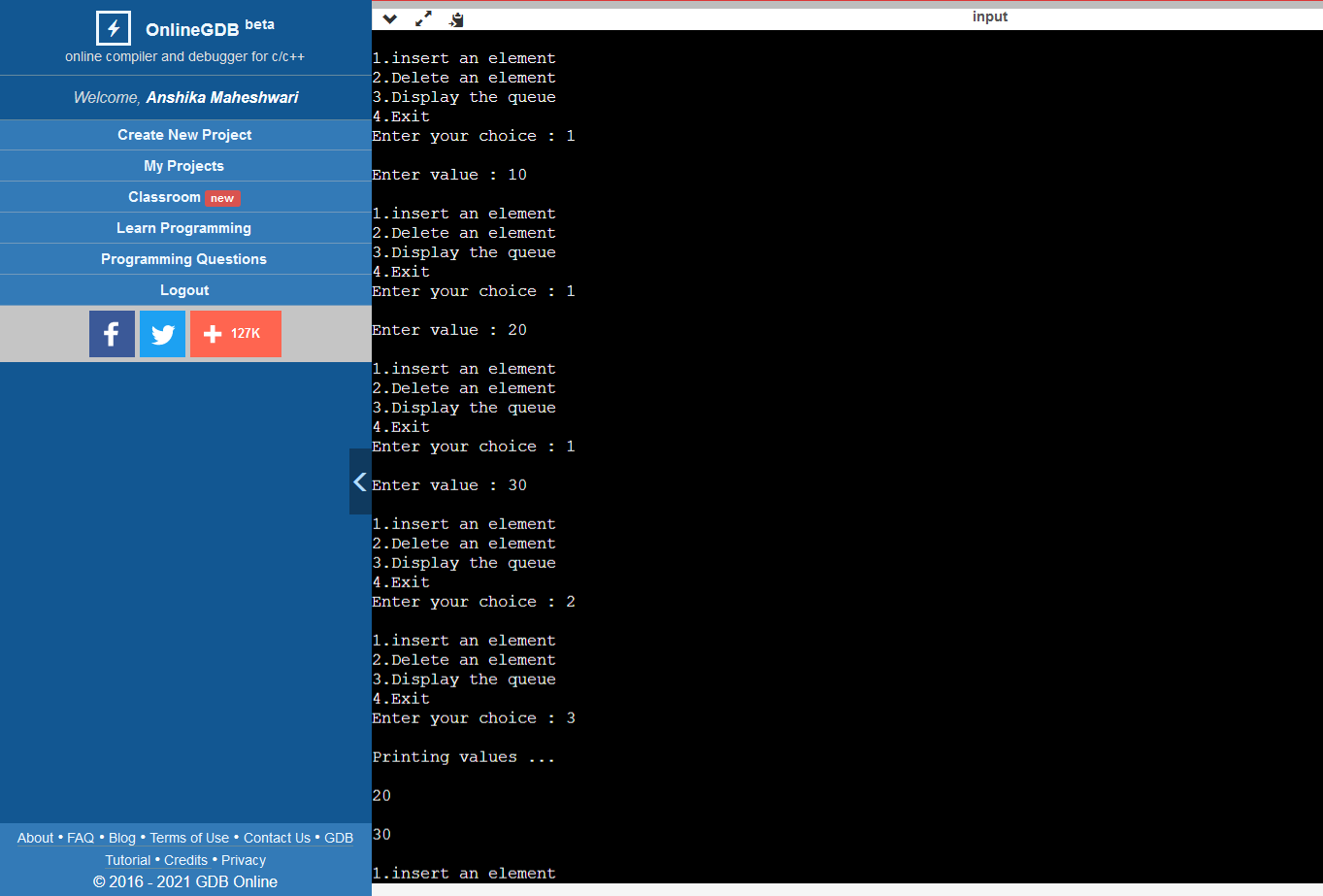
ptr = ptr -> next;

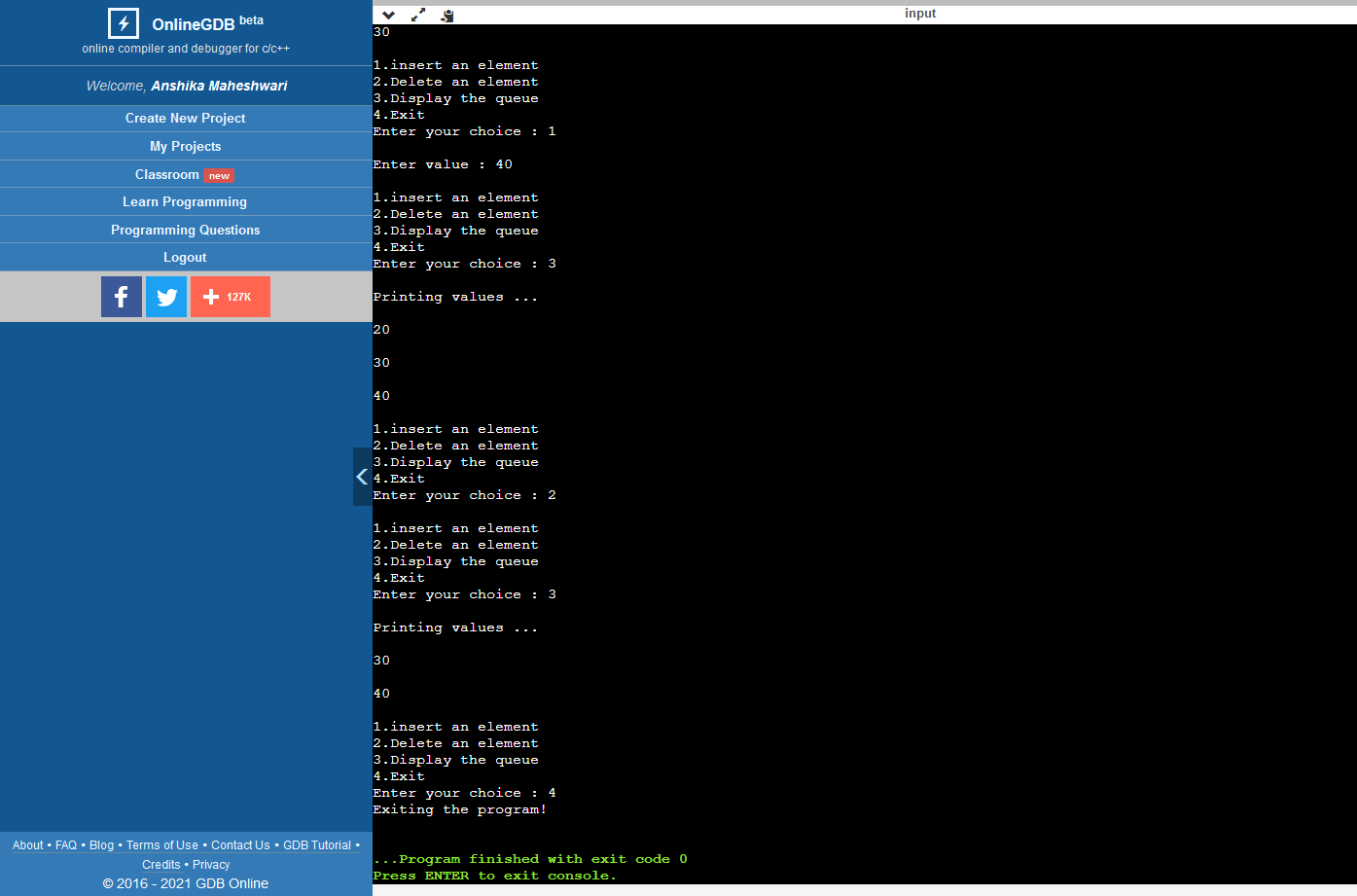
}

}

}

**Output**





9. Write a C Program implement priority QUEUE using array in menu driven form.

**Source Code :**

#include<stdio.h>

#define N 20

int Q[N],Pr[N];

int r = -1,f = -1;

void enqueue(int data,int p)

{

int i;

if((f==0)&&(r==N-1))

printf("Queue is full");

else

{

if(f==-1)

{

f = r = 0;

Q[r] = data;

Pr[r] = p;

}

else if(r == N-1)

{

for(i=f;i<=r;i++) { Q[i-f] = Q[i]; Pr[i-f] = Pr[i]; r = r-f; f = 0; for(i = r;i>f;i--)

{

if(p>Pr[i])

{

Q[i+1] = Q[i];

Pr[i+1] = Pr[i];

}

else

break;

Q[i+1] = data;

Pr[i+1] = p;

r++;

}

}

}

else

{

for(i = r;i>=f;i--)

{

if(p>Pr[i])

{

Q[i+1] = Q[i];

Pr[i+1] = Pr[i];

}

else

break;

}

Q[i+1] = data;

Pr[i+1] = p;

r++;

}

}

}

void print()

{

int i;

for(i=f;i<=r;i++)

{

printf("\nElement = %d\tPriority = %d",Q[i],Pr[i]);

}

}

int dequeue()

{

if(f == -1)

{

printf("Queue is Empty");

}

else

{

printf("deleted Element = %d\t Its Priority = %d",Q[f],Pr[f]);

if(f==r)

f = r = -1;

else

f++;

}

}

int main()

{

int opt,n,i,data,p;

printf("Enter Your Choice:-");

do{

printf("\n\n1 for Insert the Data in Queue\n2 for show the Data in Queue \n3 for Delete the data from the Queue\n0 for Exit");

scanf("%d",&opt);

switch(opt){

case 1:

printf("\nEnter the number of data");

scanf("%d",&n);

printf("\nEnter your data and Priority of data");

i=0;

while(i<n){

scanf("%d %d",&data,&p);

enqueue(data,p);

i++;

}

break;

case 2:

print();

break;

case 3:

dequeue();

break;

case 0:

break;

default:

printf("\nIncorrect Choice");

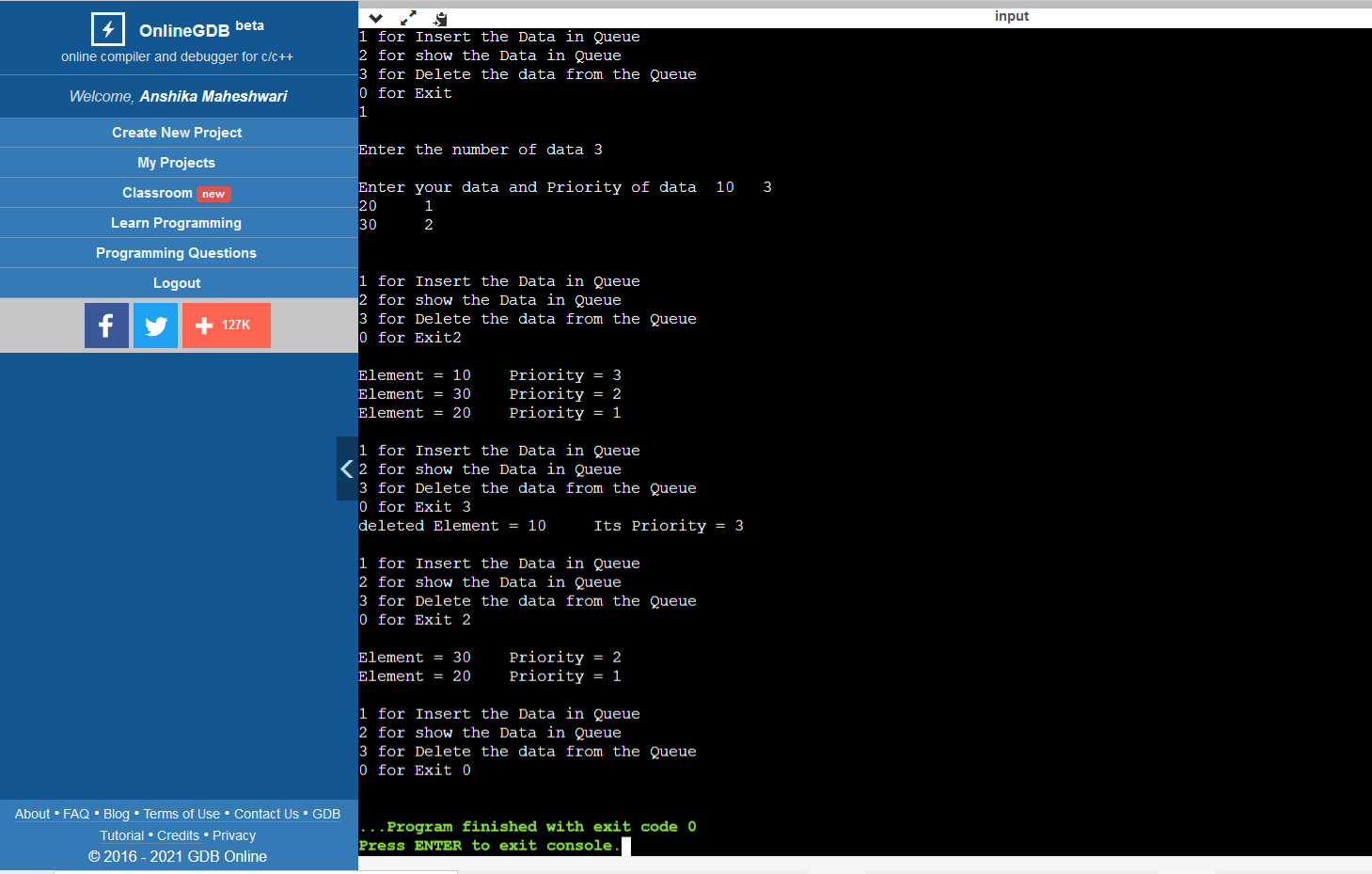
}

}while(opt!=0);

return 0;

}

**Output**



10. Write a C program implement QUEUE using array in menu driven form.

**Source Code :**

#include<stdio.h>

#define size 10

int queue[size];

int front=-1,rare=-1;

void enqueue(int x)

{

if( rare<=size-1)

{

rare++;

queue[rare]=x;

}

else {

printf("queue is full\n");

}

}

void dequeue()

{

if(front>rare)

{

printf("queue is empty\n");

}

else if(front==-1)

{

front++;

printf("dequeued element is==>%d\n",queue[front]);

}

else{

printf("dequeued element is==>%d\n",queue[front]);

front++;

}

}

void peek()

{

if(front==-1 && rare==-1)

{

printf("queue is empty\n");

}

int temp=front+1;

printf("%d\n",queue[temp]);

}

void display()

{

if(rare==-1)

{

printf("queue is empty\n");

} int temp=front+1;

while(temp<rare+1)

{

printf("%d ",queue[temp]);

temp++;

}

}

int main()

{

while(1)

{

int n;

printf("enter your choice\n");

printf("1.enqueue 2.dequeue 3.peek 4.display\n");

scanf("%d",&n);

switch(n)

{

case 1: printf("enter element\n");

int x;

scanf("%d",&x);

enqueue(x);

break;

case 2:dequeue();

break;

case 3: peek();

break;

case 4: display();

break;

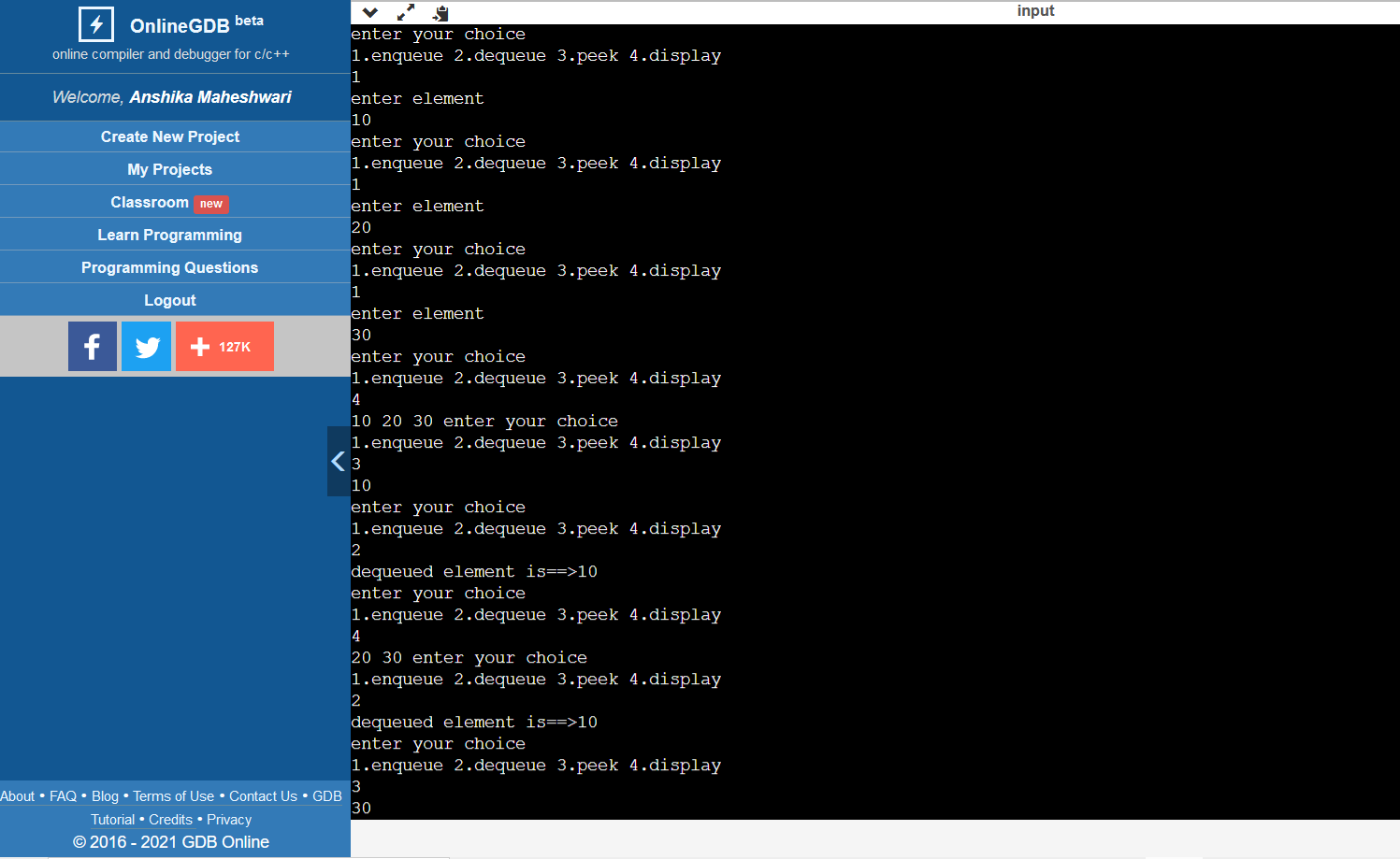
default :return 0;

}

}

}

**Output**



11. Write a C program to evaluate Postfix Expression using Stack.

**Source Code :**

#include<stdio.h>

#include<ctype.h>

int stack[20];

int top = -1;

void push(int x)

{

stack[++top] = x;

}

int pop()

{

return stack[top--];

}

int main()

{

char exp[20];

char \*e;

int n1,n2,n3,num;

printf("Enter the expression : ");

scanf("%s", exp);

e = exp;

while(\*e != '\0')

{

if(isdigit(\*e))

{

num = \*e - 48;

push(num);

}

else

{

n1 = pop();

n2 = pop();

switch(\*e)

{

case '+':

{

n3 = n1 + n2;

break;

}

case '-':

{

n3 = n2 - n1;

break;

}

case '\*':

{

n3 = n1 \* n2;

break;

}

case '/':

{

n3 = n2 / n1;

break;

}

}

push(n3);

}

e++;

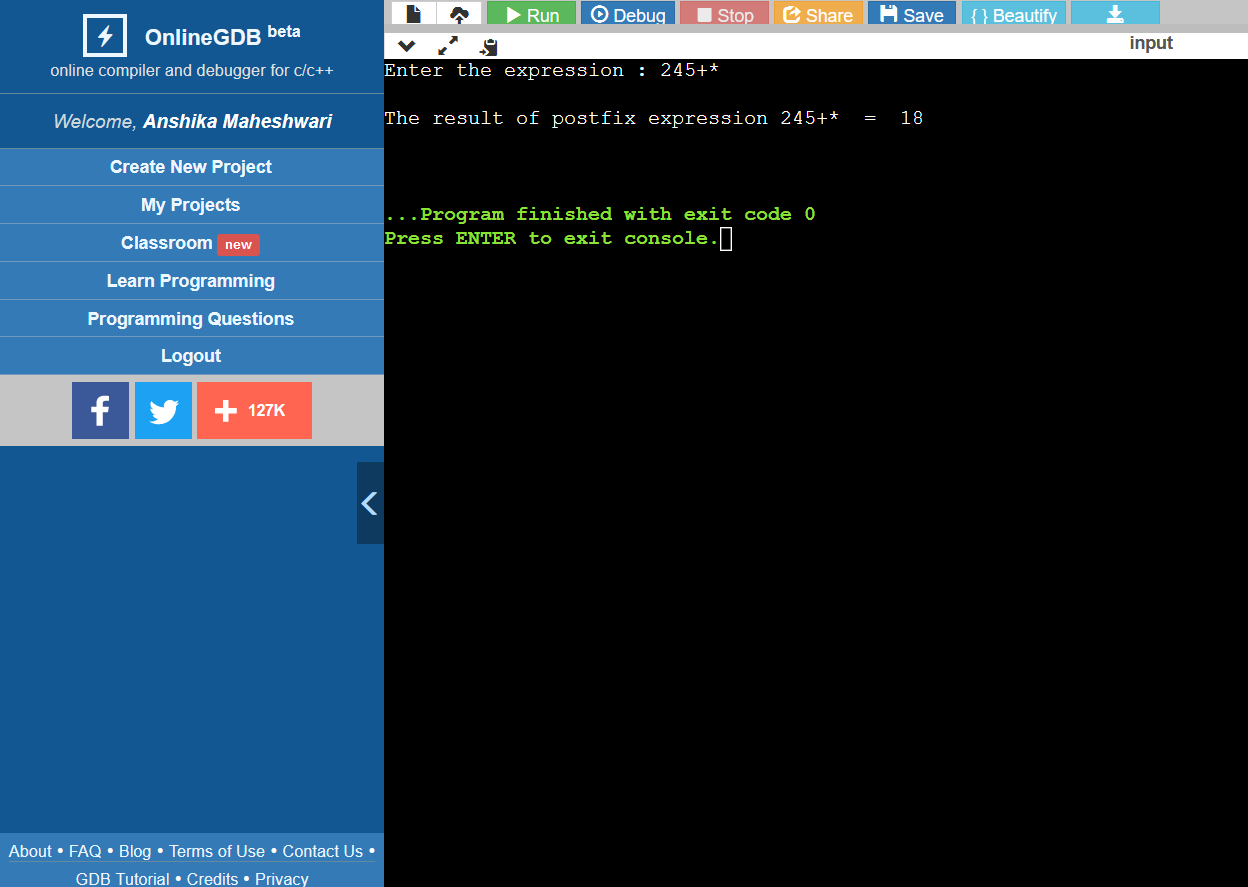
}

printf("\nThe result of postfix expression %s = %d\n\n", exp, pop());

return 0;

}

**Output**

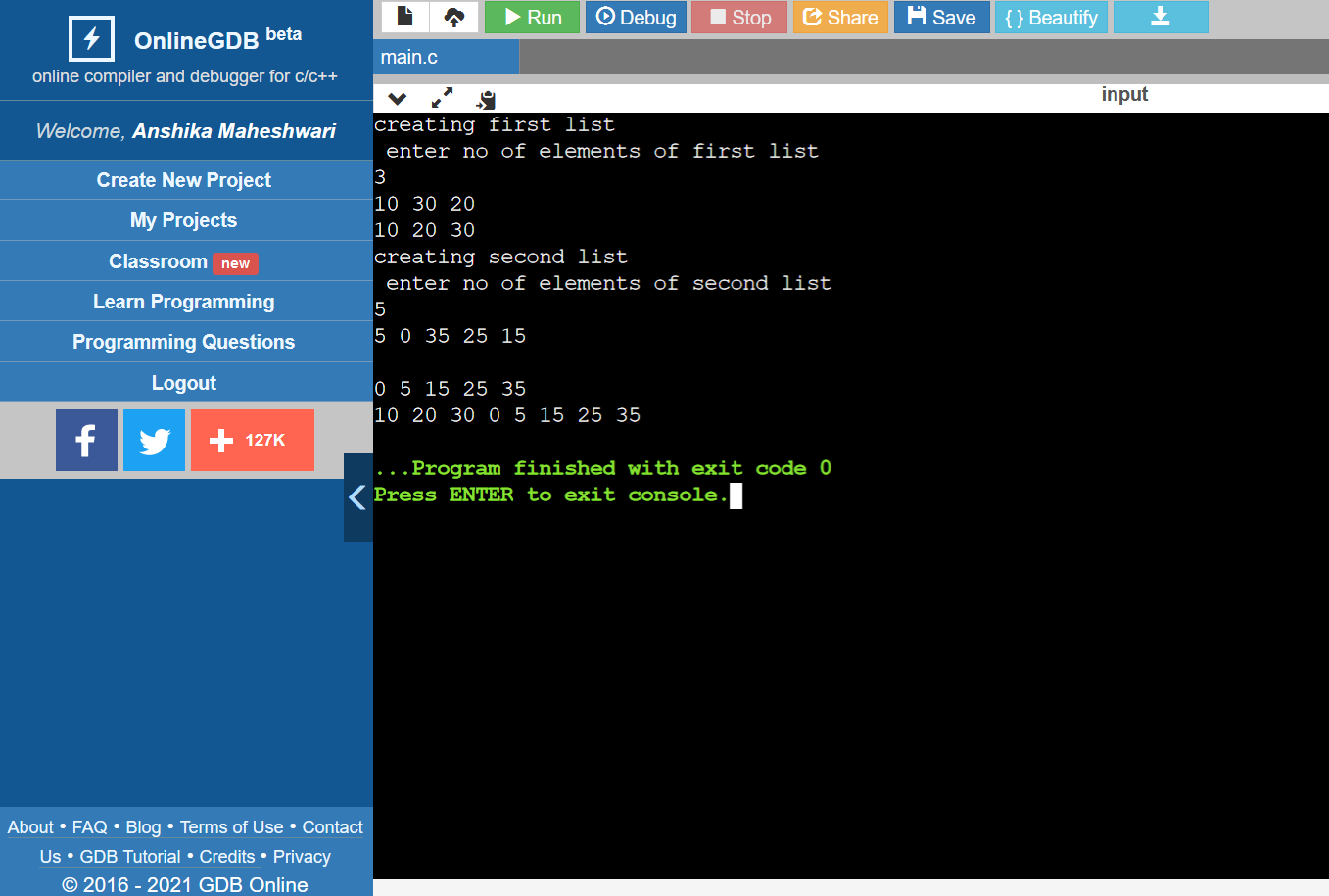


12. Write a C program to create TWO singly linked list L1 and L2 and sort both the list and finally merge both the list such that L2 comes after L1.[ use double pointer]

**Source Code :**

|  |
| --- |
| #include<stdio.h> |
| #include<stdlib.h> |
| struct node { |
| int data; |
| struct node\* next; |
| }; |
| void create(struct node\*\*head,int data) |
| { |
| struct node\* temp=(struct node\*)malloc(sizeof(struct node)); |
| temp->data=data; |
| temp->next=NULL; |
| struct node\* ptr,\*ptr1; |
| if(\*head==NULL) |
| { |
| \*head=temp; |
| ptr=temp; |
| ptr1=temp; |
| } |
| else if(data>ptr->data) |
| { |
| ptr->next=temp; |
| ptr=temp; |
| } |
| else if(data<ptr1->data) |
| { |
| temp->next=ptr1; |
| ptr1=temp; |
| \*head=temp; |
| } |
| else{ |
| struct node\* temp1=\*head; |
| while(temp1!=NULL) |
| { |
| if(temp1->data<data && temp1->next->data>data) |
| { |
| temp->next=temp1->next; |
| temp1->next=temp; |
| break; |
| } |
| temp1=temp1->next; |
| } |
| } |
| } |
| void print\_node(struct node\* head) |
| { |
| while(head!=NULL) |
| { |
| printf("%d ",head->data); |
| head=head->next; |
| } |
| } |
| void merge(struct node\*\* head,struct node\*ptr) |
| { |
| struct node \*temp=\*head; |
| while(temp->next!=NULL) |
| { |
| temp=temp->next; |
| } |
| temp->next=ptr; |
| } |
| int main() |
| { |
| struct node\* head=NULL; |
| struct node\* ptr=NULL; |
| struct node\* temp; |
| printf("creating first list\n enter no of elements of first list\n"); |
| int n; |
| scanf("%d",&n); |
| while(n--) |
| { |
| int data; |
| scanf("%d",&data); |
| create(&head,data); |
| } |
| print\_node(head); |
| printf("\ncreating second list\n enter no of elements of second list\n"); |
| int k; |
| scanf("%d",&k); |
| while(k--) |
| { |
| int data; |
| scanf("%d",&data); |
| create(&ptr,data); |
| } |
| print\_node(ptr); |
| merge(&head,ptr); |
| print\_node(head); |
| } |

**Output**

****

13. Write C program to create a doubly link list by adding the node right hand side and then check list is in palindrome form or not.

**Source Code :**

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node\* next;

struct Node\* prev;};

void create(struct Node\*\*head,struct Node\*\*temp)

{

struct Node\*ptr;

int choice=1;

while(choice)

{

ptr=(struct Node\*)malloc(sizeof(struct Node));

printf("Enter the data ");

scanf("%d",&ptr->data);

ptr->next=NULL;

ptr->prev=NULL;

if(\*head==NULL && \*temp==NULL)

\*head=\*temp=ptr;

else

{

(\*temp)->next=ptr;

ptr->prev=\*temp;

\*temp=ptr;

}

printf("\n Enter 1 to continue or 0 to finish entry");

scanf("%d",&choice);

}}

void check\_palindrome(struct Node\*\*head)

{

struct Node \*temp=(\*head);

while(temp->next!=NULL)

temp=temp->next;

while(\*head!=temp && (\*head)->prev != temp->next)

{

if((\*head)->data!=temp->data)

{

\*head=(\*head)->next;

temp=temp->prev;

}

else

{

printf("List is not palindrome");

return;

}

}

printf("List is a palindrome");

}

void Display(struct Node \*head)

{

while (head != NULL)

{

printf("%d ", head->data);

head = head->next;

}

}

int main()

{

struct Node \*head=NULL,\*temp=NULL;

create(&head,&temp);

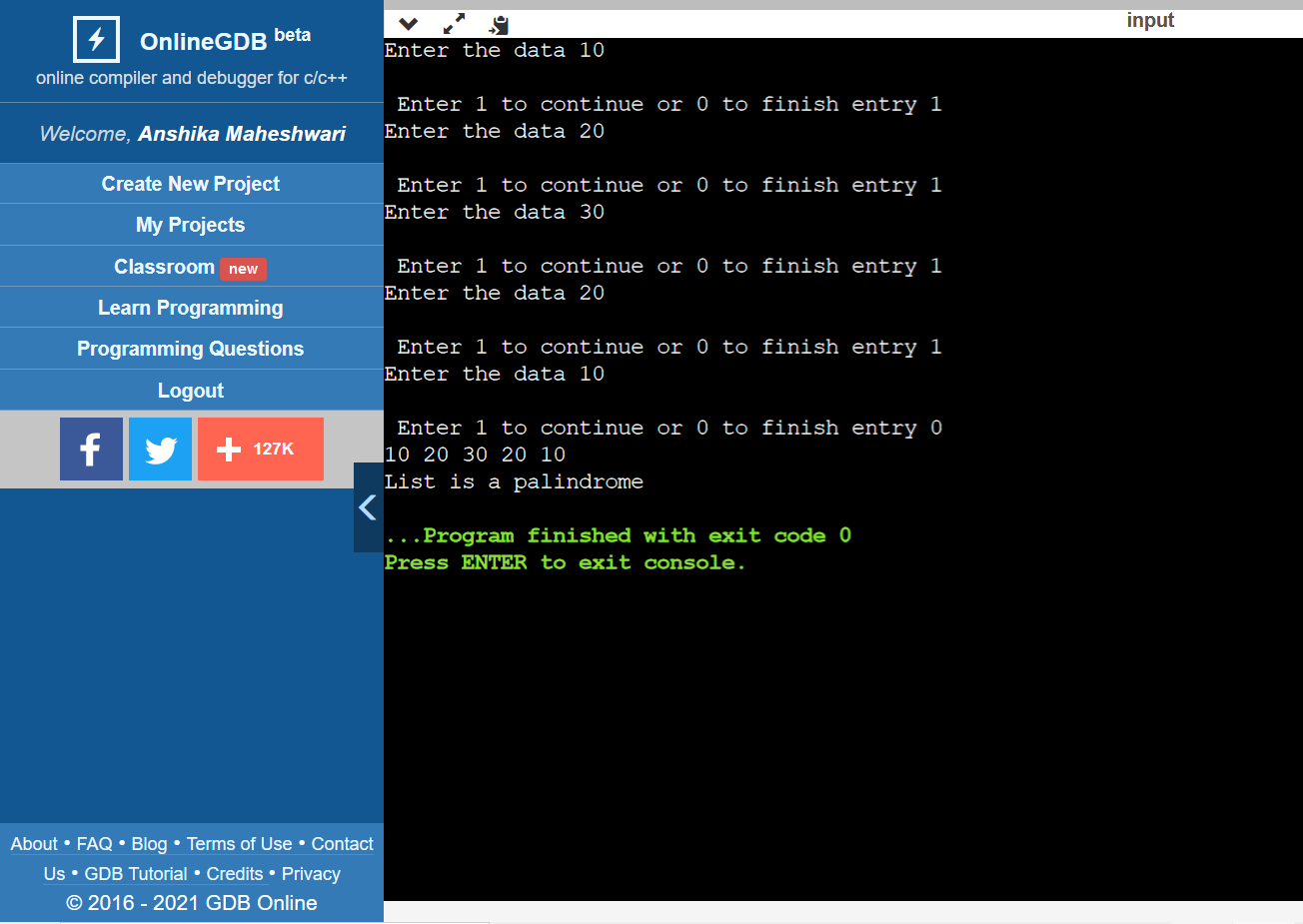
Display(head);

printf("\n");

check\_palindrome(&head);

}

**Output:**



14. Write a C program to create a circular linked list by adding the nodes in right hand side and then print the list.

**Source Code :**

#include <stdio.h>

#include <stdlib.h>

struct node

{

int info;

struct node \*next;

};

void push(struct node \*\*head, int val)

{

struct node \*newNode = malloc(sizeof(struct node));

newNode->info = val;

newNode->next = NULL;

if (\*head == NULL)

{

\*head = newNode;

newNode->next = \*head;

}

else

{

struct node \*lastNode = \*head;

while (lastNode->next != \*head)

{

lastNode = lastNode->next;

}

lastNode->next = newNode;

newNode->next = \*head;

}

}

void print(struct node \*ptr)

{

struct node \*temp = ptr;

do

{

printf("%d ", temp->info);

temp = temp->next;

} while (temp != ptr);

}

int main()

{

struct node \*start = NULL;

push(&start, 19);

push(&start, 18);

push(&start, 12);

push(&start, 11);

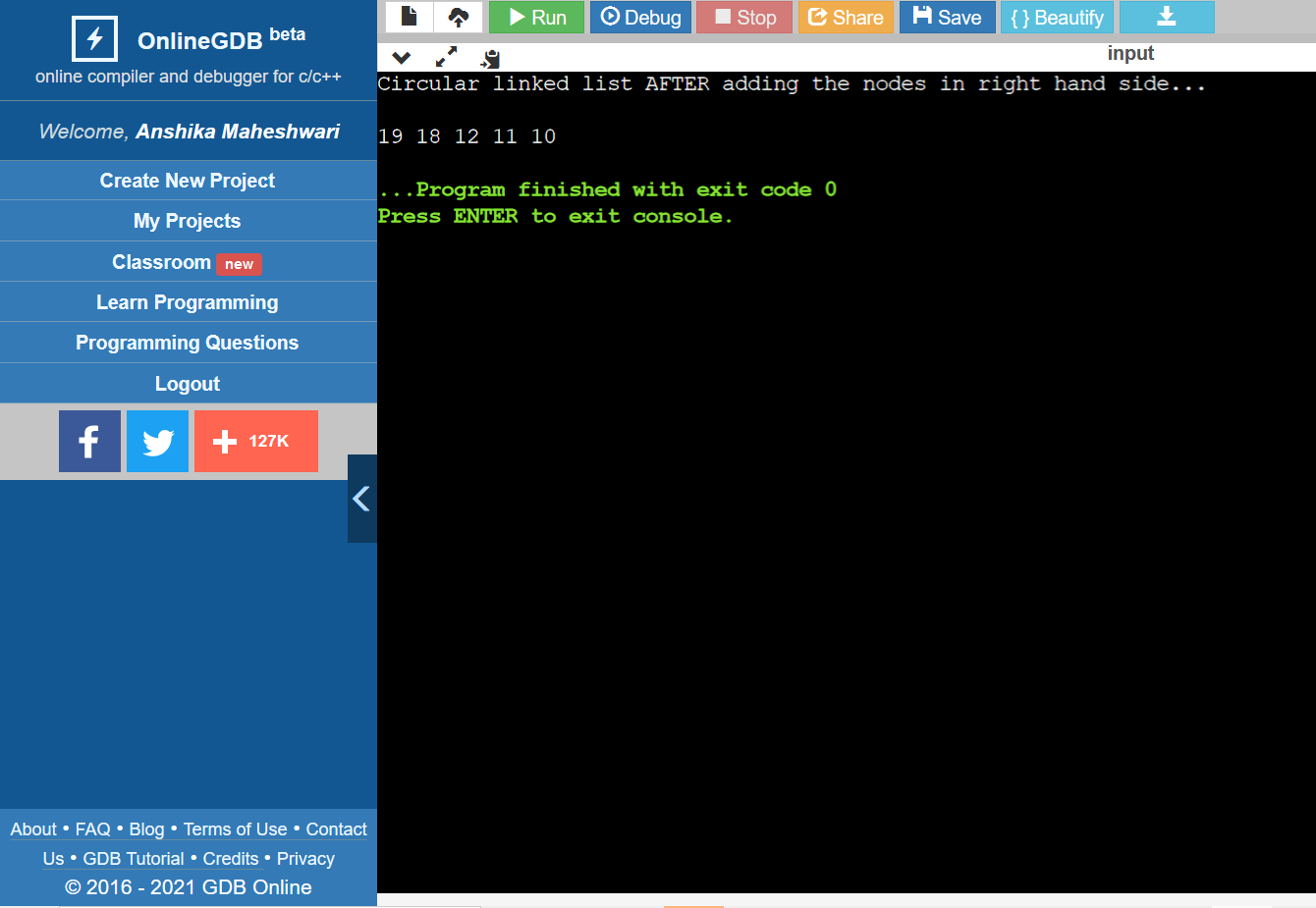
push(&start, 10);

printf("Circular linked list AFTER adding the nodes in right hand side...\n\n");

print(start);

**}**

**Output**

****