Practical-3

AIM:Implement Sorting Algorithm(s).

(a) Bubble Sort

(b) Selection Sort

(c) Insertion Sort

Program:

(a)Bubble sort

#include<iostream>

using namespace std;

int sorting(int arr[],int n)

{

int cnt=0;

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

{

for(int j=0;j<n-1;j++)

{

if(arr[j]>arr[j+1])

{

int temp;

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

cnt++;

}

}

}

cout<<"sorted elements are "<<endl;

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

{

cout<<"Elements ["<<i+1<<"] :"<<arr[i]<<endl;

}

if(cnt==0)

{

cout<<"Number of times Elements exchanged is 0:";

}

else

{

cout<<"Number of times Elements exchanged is :"<<cnt;

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n\n";

int arr[20];

int size;

cout<<"Enter the number of elements :";

cin>>size;

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

{

cout<<"Elements ["<<j+1<<"] :";

cin>>arr[j];

}

sorting(arr,size);

}

(b)Merge sort

#include<iostream>

using namespace std;

int sorting(int arr[],int n)

{

int min\_index;

for(int pass=0;pass<n-1;pass++)

{

min\_index=pass;

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

{

if(arr[i]<arr[min\_index])

{

min\_index=i;

}

}

if(min\_index!=pass)

{

int temp;

temp=arr[pass];

arr[pass]=arr[min\_index];

arr[min\_index]=temp;

}

}

cout<<"sorted elements are "<<endl;

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

{

cout<<"Elements ["<<i+1<<"] :"<<arr[i]<<endl;

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n\n";

int arr[20];

int size;

cout<<"Enter the number of elements :";

cin>>size;

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

{

cout<<"Elements ["<<j+1<<"] :";

cin>>arr[j];

}

sorting(arr,size);

}

(c)insertion sort

#include<iostream>

using namespace std;

int isorting(int arr[],int n)

{

int key,j;

for(int i=2;i<=n;i++)

{

key=arr[i];

j=i-1;

while(j>0 && arr[j]>key)

{

arr[j+1]=arr[j];

j=j-1;

}

arr[j+1]=key;

}

cout<<"sorted elements are "<<endl;

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

{

cout<<"Elements ["<<i<<"] :"<<arr[i]<<endl;

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n\n";

int arr[20];

int size;

cout<<"Enter the number of elements :";

cin>>size;

for(int j=1;j<=size;j++)

{

cout<<"Elements ["<<j<<"] :";

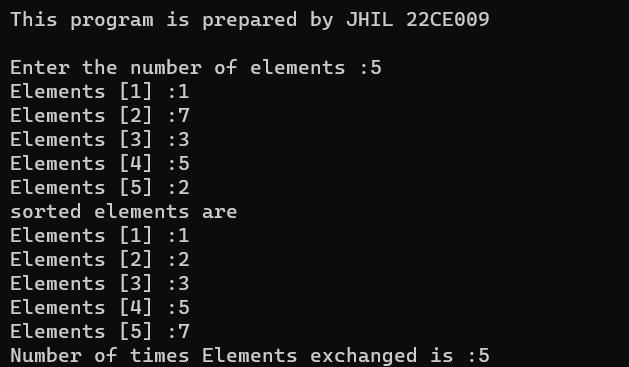
cin>>arr[j];

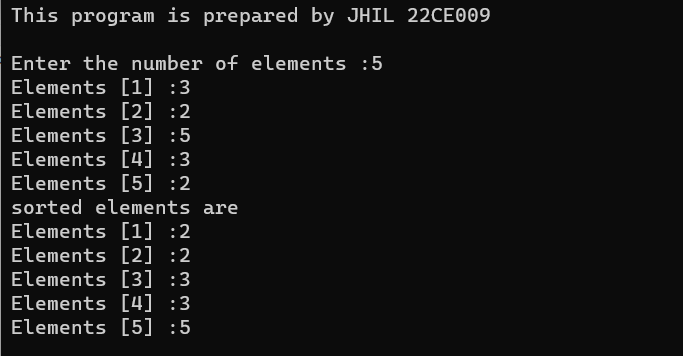
}

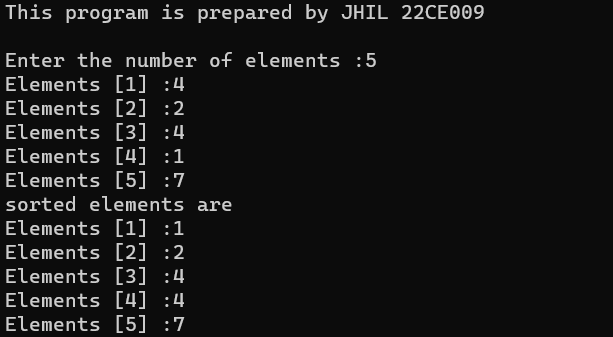
isorting(arr,size);

}

Output:-







CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-4

AIM: Sort Colors: Given an array nums with n objects colored red, white, or blue, sort them in-place so that objects of the same color are adjacent, with the colors in the order red, white, and blue. We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively. Note: You must solve this problem without using the library's sort function.

Program:

#include<iostream>

using namespace std;

int main()

{

Cout<<”This program is prepared by JHIL 22CE009”;

int a[50];

int cnt1=0,cnt2=0,cnt3=0;

int size;

cout<<"Enter the size of colours in an array : ";

cin>>size;

cout<<"Enter 0 for red colour"<<endl;

cout<<"Enter 1 for white colour"<<endl;

cout<<"Enter 2 for blue colour"<<endl;

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

{

cout<<"colour["<<i<<"]:";

cin>>a[i];

}

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

{

if(a[i]==0)

{

cnt1++;

}

if(a[i]==1)

{

cnt2++;

}

if(a[i]==2)

{

cnt3++;

}

}

for(int i=0;i<cnt1;i++)

{

a[i]=0;

}

for(int i=cnt1;i<cnt2+cnt1;i++)

{

a[i]=1;

}

for(int i=cnt1+cnt2;i<size;i++)

{

a[i]=2;

}

cout<<"Sorted colours "<<endl;

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

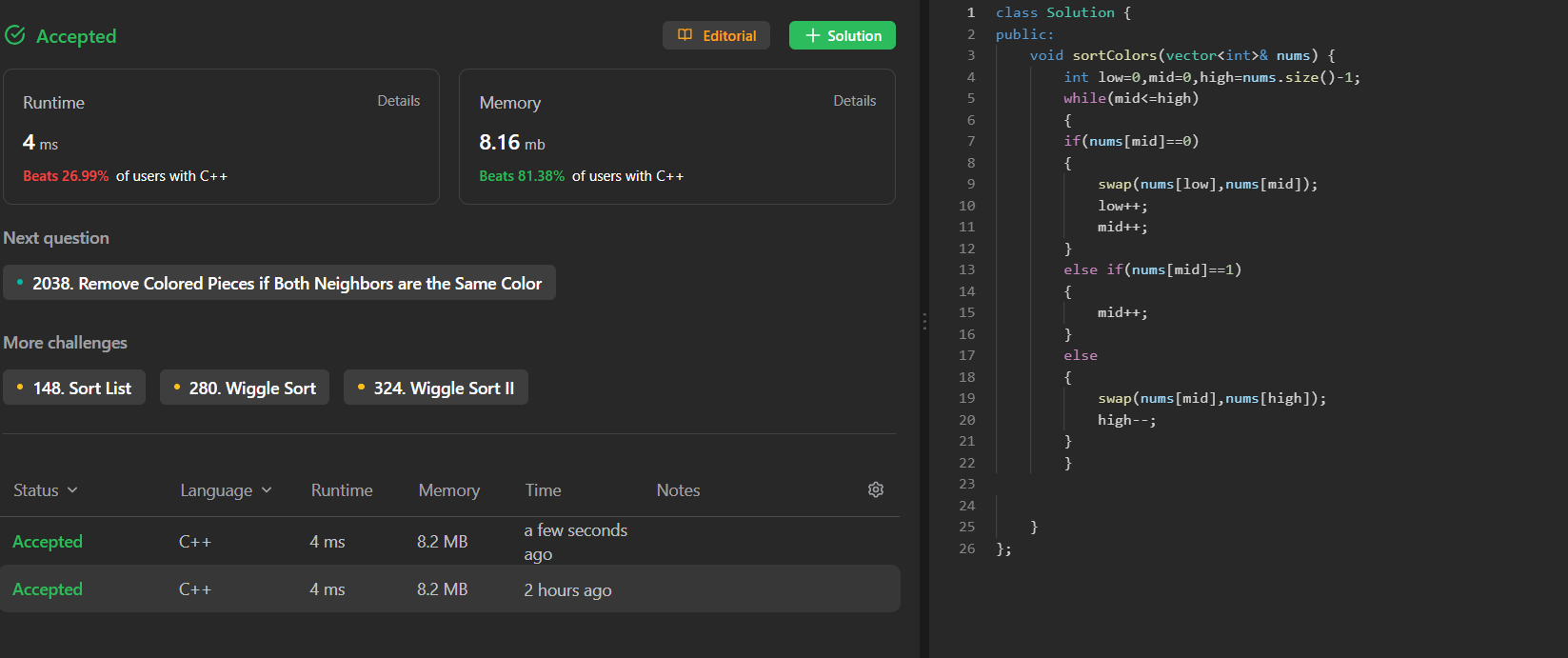
{

cout<<"colour["<<i<<"]:"<<a[i]<<endl;

}

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-5

AIM: Chef and his little brother are playing with sticks. They have total N sticks. Length of i-th stick is Ai. Chef asks his brother to choose any four sticks and to make a rectangle with those sticks its sides. Chef warns his brother to not to break any of the sticks, he has to use sticks as a whole. Also, he wants that the rectangle formed should have the maximum possible area among all the rectangles that Chef's brother can make.

Chef's little brother takes this challenge up and overcomes it. Can you also do so? That is, you have to tell whether it is even possible to create a rectangle? If yes, then you have to tell the maximum possible area of rectangle.

Input

* The first line contains a single integer T denoting the number of test- cases. T test cases follow.
* The first line of each test case contains a single integer N denoting the number of sticks.
* The second line of each test case contains N space-separated integers A1, A2, ..., AN denoting the lengths of sticks.

Output

* For each test case, output a single line containing an integer representing the maximum possible area for rectangle or -1 if it's impossible to form any rectangle using the available sticks.

Input 2

5

1 2 3 1 2

4

1 2 2 3

Program:

#include<iostream>

using namespace std;

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n"<<endl;

int arr[50];

int size,cnt=0;

int length,breadth;

cout<<"Enter the size of an array :";

cin>>size;

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

{

cout<<"arr["<<i<<"]:";

cin>>arr[i];

}

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

{

for(int j=0;j<size-1;j++)

{

if(arr[j]<arr[j+1])

{

int temp;

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

//for(int i=0;i<size;i++)

//{

//cout<<"arr["<<i<<"]:"<<arr[i]<<endl;

//}

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

{

if(arr[j]==arr[j+1])

{

if(cnt==1)

{

breadth=arr[j];

break;

}

cnt++;

length=arr[j];

j++;

}

}

if(cnt==0)

{

cout<<"Can't determine maximum area of rectangle";

}

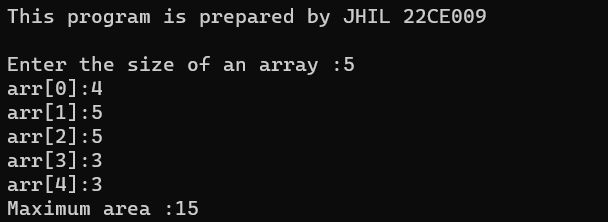
else{

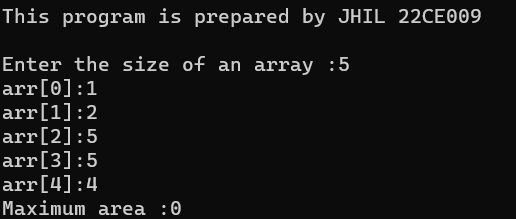
cout<<"Maximum area :"<<length\*breadth;

}

}

Output:-





CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-10

AIM: **Implement stack using array**

Implement a program to implement a Stack using Array. Your task is to use the class as shown in the comments in the code editor and complete the functions push () and pop () to implement a stack.

**Example 1**: Input: push(2)

push(3)

pop()

push(4)

pop()

Output: 3, 4

Program:

#include<iostream>

int top=-1;

int arr[100];

using namespace std;

int push(int x,int n)

{

if(top==(n-1))

{

cout<<"overflow"<<endl;

}

else

{

top++;

arr[top]=x;

}

}

int pop()

{

if(top==-1)

{

cout<<"underflow"<<endl;

}

else

{

int x=arr[top];

top--;

return x;

}

}

void display()

{

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

{

cout<<arr[i];

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n"<<endl;

int n,x,j;

cout<<"Enter the size of an array :";

cin>>n;

do

{

cout<<"Enter 1 for push"<<endl;

cout<<"Enter 2 for pop"<<endl;

cout<<"Enter 3 for display"<<endl;

cout<<"Enter 4 for exit"<<endl;

cin>>j;

switch(j)

{

case 1:

cout<<"stack :";

cin>>x;

push(x,n);

break;

case 2:

x=pop();

cout<<x;

break;

case 3:

display();

break;

case 4:

return 0;

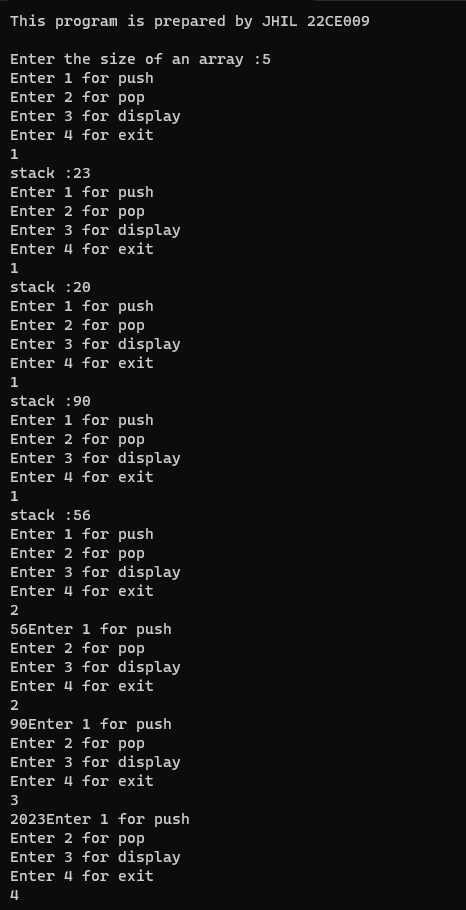
}

}

while(j>=1 && j<=4);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-12

AIM: **Implement Stack using Linked List**

You have a linked list and you have to implement the functionalities push and pop of stack using this given linked list. Your task is to use the class as shown in the comments in the code editor and complete the functions push () and pop () to implement a stack.

**Example 1**: **Input:** push(2)

push(3)

pop()

push(4)

pop()

**Output: 3 4**

Program:

#include<iostream>

using namespace std;

struct node

{

int data;

struct node \*link;

};

struct node \*head=NULL;

void push(int item)

{

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

if(p==NULL)

{

cout<<"Memory error";

}

if (head==NULL)

{

p->data=item;

p->link=NULL;

head=p;

}

else

{

p->data=item;

p->link=head;

head=p;

}

}

int pop()

{

int item;

struct node \*p;

if(head==NULL)

{

cout<<("underflow");

return -1;

}

item=head->data;

p=head;

head=head->link;

free(p);

return item;

}

void display()

{

struct node \*ptr;

ptr=head;

if(ptr==NULL)

{

cout<<"Stack i empty\n";

}

else

{

cout<<"printing stack elements";

while(ptr!=NULL)

{

cout<<"\n"<<ptr->data;

ptr=ptr->link;

}

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n"<<endl;

int n,x,j;

cout<<"Enter the size of an link-list :";

cin>>n;

do

{

cout<<"Enter 1 for push"<<endl;

cout<<"Enter 2 for pop"<<endl;

cout<<"Enter 3 for display"<<endl;

cout<<"Enter 4 for exit"<<endl;

cin>>j;

switch(j)

{

case 1:

cin>>x;

push(x);

break;

case 2:

x=pop();

cout<<x;

break;

case 3:

display();

break;

case 4:

return 0;

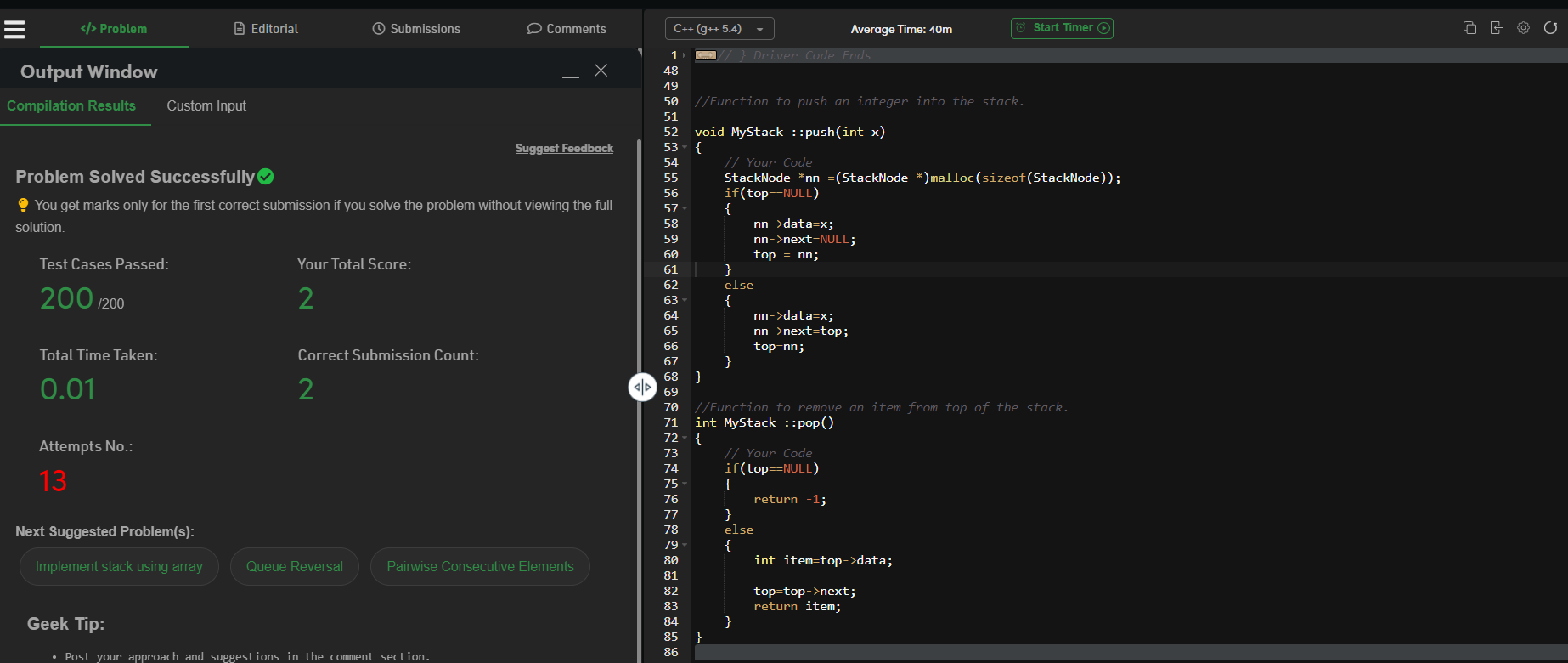
}

}

while(j>=1 && j<=4);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-14

AIM: [**Valid Parentheses**](https://leetcode.com/problems/valid-parentheses/)

Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if

the input string is valid.

An input string is valid if:

Open brackets must be closed by the same type of brackets. Open brackets must be closed in the correct order.

Every close bracket has a corresponding open bracket of the same type.

**Example 1:**

**Input: s = "()" Output: true**

Program:

#include<iostream>

int top=-1;

char arr[100];

using namespace std;

int push(int x)

{

top++;

arr[top]=x;

}

int pop()

{

int x=arr[top];

top--;

return x;

}

void display()

{

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

{

cout<<arr[i];

}

}

void balanced\_paren(int n)

{

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

{

cout<<"arr["<<i<<"]:";

cin>>arr[i];

}

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

{

if(arr[j]=='[' || arr[j]=='{' || arr[j]=='(')

{

push(arr[j]);

}

else if((arr[j]==']'&&arr[top]=='[') || (arr[j]=='}'&&arr[top]=='{') || (arr[j]==')'&&arr[top]=='('))

{

pop();

}

}

if(top==-1)

{

cout<<"BALANCED PARENTHESIS"<<endl;

}

else

{

cout<<"NOT A BALANCED PARENTHESIS"<<endl;

}

}

int main()

{

cout<<"This program is prepared by JHIL 22CE009\n"<<endl;

int n;

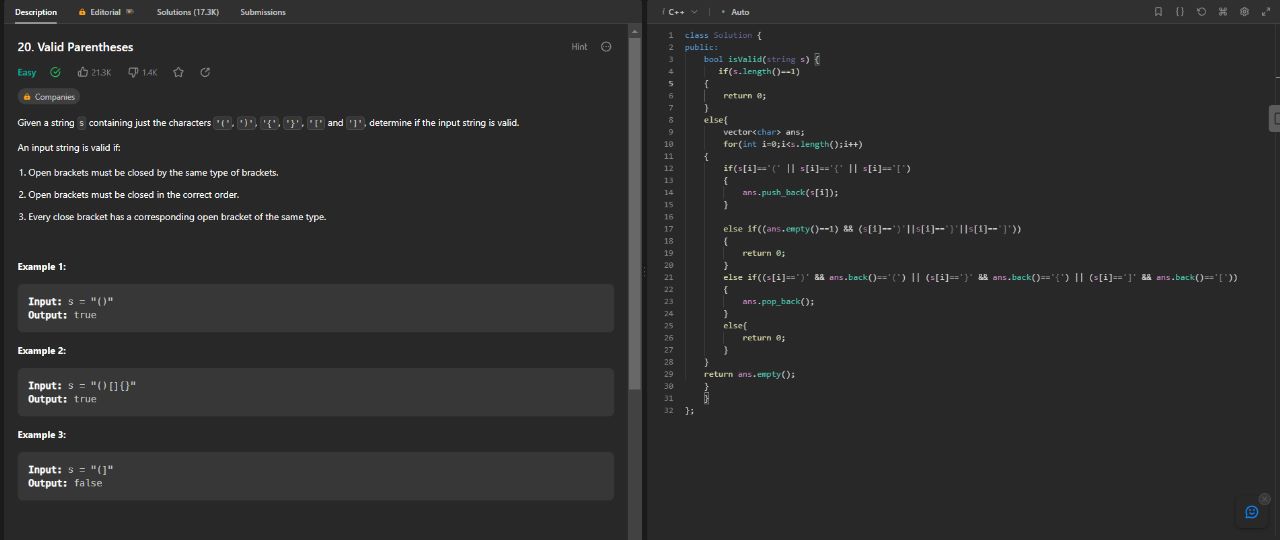
cout<<"Enter the size of an brackets :";

cin>>n;

balanced\_paren(n);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-13

AIM: [**Implement Queue using Linked List**](https://practice.geeksforgeeks.org/problems/implement-queue-using-linked-list/1)

A Query Q is of 2 Types

1. 1 x (a query of this type means pushing 'x' into the queue)
2. 2 (a query of this type means to pop an element from the queue and print the popped element) **Example 1:** Input:

Q = 5

Queries = 1 2 1 3 2 1 4 2

Output: 2 3

Program:

#include<iostream>

using namespace std;

struct node

{

int data;

struct node \*next;

};

struct node \*front=NULL,\*rear=NULL;

void enqueue(int val)

{

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

newnode->data=val;

newnode->next=NULL;

if(front==NULL && rear==NULL)

{

front=rear=newnode;

}

else

{

rear->next=newnode;

rear=newnode;

}

}

void dequeue()

{

struct node \*temp;

if(front==NULL)

{

cout<<"Queue is empty";

}

else

{

cout<<front->data<<endl;

temp=front;

front=front->next;

if(front==NULL)

{

rear=NULL;

}

free(temp);

}

}

void display()

{

while(front!=NULL)

{

cout<<"Display elements";

cout<<front->data<<endl;

front=front->next;

}

}

int main()

{

int j,x;

do

{

cout<<"Enter 1 for Enqueue"<<endl;

cout<<"Enter 2 for Dequeue"<<endl;

cout<<"Enter 3 for Display"<<endl;

cout<<"Enter 4 for Exit"<<endl;

cin>>j;

switch(j)

{

case 1:

cout<<"Queue :";

cin>>x;

enqueue(x);

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

return 0;

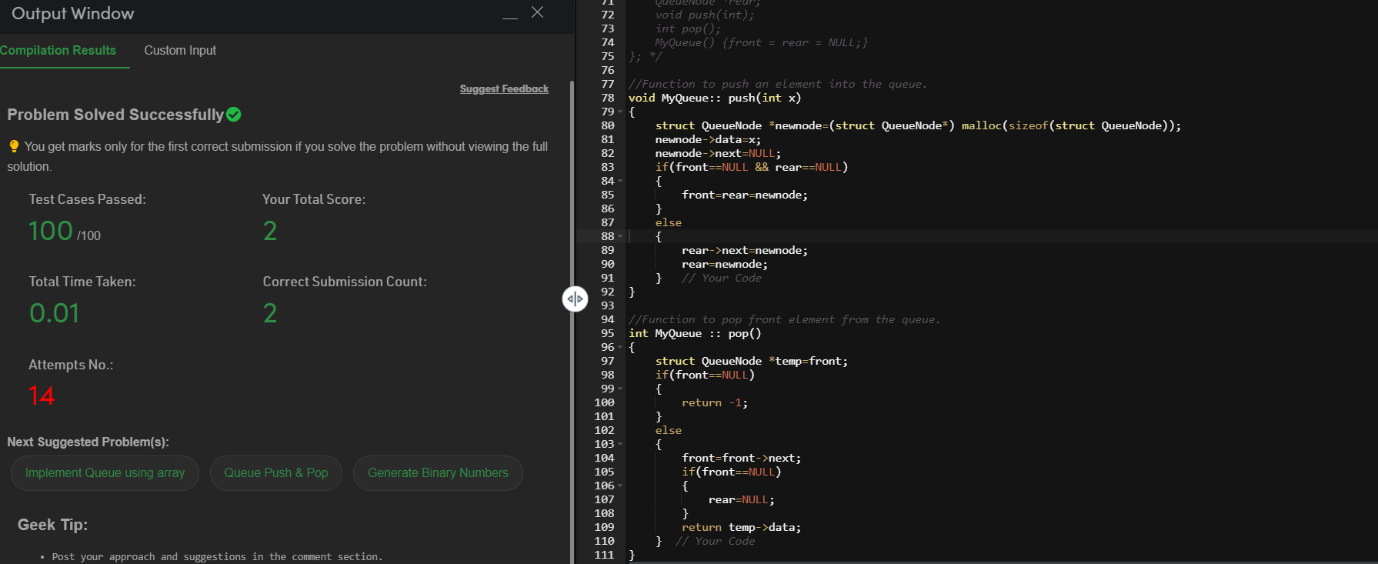
}

}

while(j>=1 && j<=3);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-6

AIM: Implement below operations of singly linked list.

1. Insert a node at front
2. Delete a node at last
3. Delete all nodes of linked list

**Note: Display content of linked list after each operation.**

Program:

#include<iostream>

using namespace std;

struct node

{

int info;

struct node\* link;

};

struct node\* first=NULL;

struct node\* create(int a)

{

struct node\* temp1;

temp1=(struct node\*)malloc(sizeof(struct node));

temp1->info=a;

temp1->link=NULL;

return temp1;

}

struct node\* inserttAtFirst(int x)

{

struct node\* temp=create(x);

if(first==NULL)

{

first=temp;

}

else

{

temp->link=first;

first=temp;

}

return temp;

}

int deleteeAtLast()

{

if(first->link==NULL)

{

free(first);

}

else

{

struct node\* pre;

struct node\* trav;

trav=first;

while(trav->link != NULL)

{

pre=trav;

trav=trav->link;

}

pre->link=NULL;

int x = trav->info;

free(trav);

return x;

}

}

void deleteeall()

{

struct node\* trav;

while(first !=NULL)

{

trav=first;

first=first->link;

free(trav);

}

cout<<"All the nodes are deleted"<<endl;

}

void display()

{

struct node\* t=first;

while(t != NULL)

{

cout<<t->info<<endl;

t=t->link;

}

}

main()

{

cout<<"This program is prepared by 22CE009 JHIL"<<endl;

int ch,x,n,i;

cout<<"1. INSERT AT FIRST"<<endl<<"2. DELETE AT LAST"<<endl<<"3. DISPLAY"<<endl<<"4. DELETE ALL"<<endl;

do{

cout<<"Enter choice: ";

cin>>ch;

switch(ch)

{

case 1:

cout<<"Enter the element you have to insert: ";

cin>>x;

first=inserttAtFirst(x);

break;

case 2:

n=deleteeAtLast();

cout<<"The deleted info of the node: "<<n<<endl;

break;

case 3:

display();

break;

case 4:

deleteeall();

break;

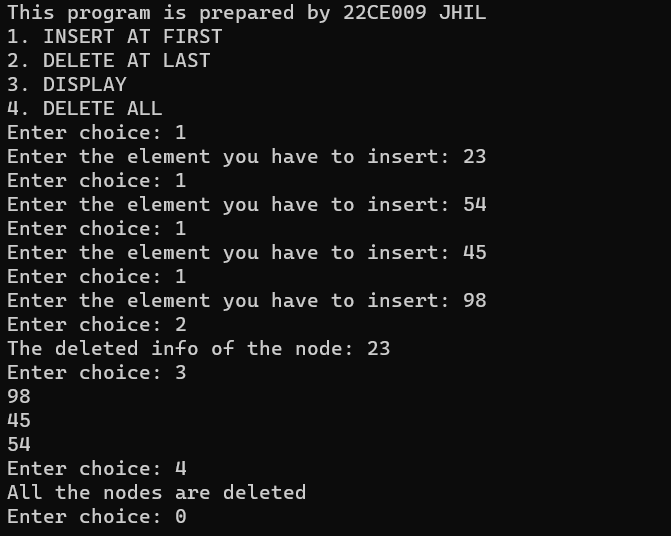
}

}

while(ch!=0);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

Practical-11

AIM: **Implement Queue using array**

Implement a Queue using an Array. Queries in the Queue are of the following type:

1. 1 x (a query of this type means pushing 'x' into the queue)

2 (a query of this type means to pop element from queue and print the poped element)

Program:

#include<iostream>

using namespace std;

int f=-1;

int rear=-1;

int q[5];

int l=5;

void enque(int n)

{

if(rear==l-1)

{

cout<<"Queue overflow";

}

else

{

rear++;

q[rear]=n;

// cout<<q[rear]<<" ";

}

if(f==-1)

{

f=f+1;

}

}

void dequee()

{

int y;

if(f== -1)

{

cout<<"Queue underflow";

}

else if(f==rear)

{

y=q[f];

cout<<"Deleted element: "<< y<<" "<<endl;

f=-1;

rear=-1;

}

else

{

y=q[f];

cout<<"Deleted element: "<< y<<" "<<endl;

f=f+1;

}

}

void display()

{

for(int i=f;i<=rear;i++)

{

cout<<q[i]<<" ";

}

cout<<endl;

}

main()

{

cout<<"This program is prepared by 22CE009 JHIL"<<endl;

int choice,n;

do

{

cout<<"Enter choice: "<<endl<<"1. FOR INSERT"<<endl<<"2. FOR DELETE"<<endl<<"3. FOR DISPLAY"<<endl;

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter element you want to insert: ";

cin>>n;

enque(n);

break;

case 2:

dequee();

//cout<<"The deleted element is: "<<d<<" ";

break;

case 3:

display();

break;

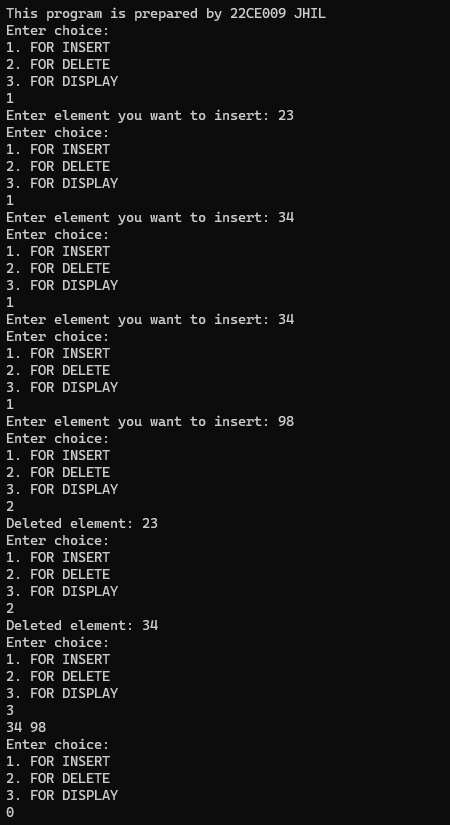
}

}

while(choice!=0);

}

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks

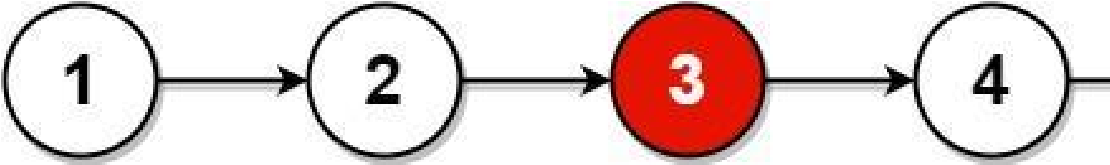
Practical-8

AIM: **Middle of the Linked List**

Given the head of a singly linked list, return the middle node of the linked list.

If there are two middle nodes, return the second middle node.

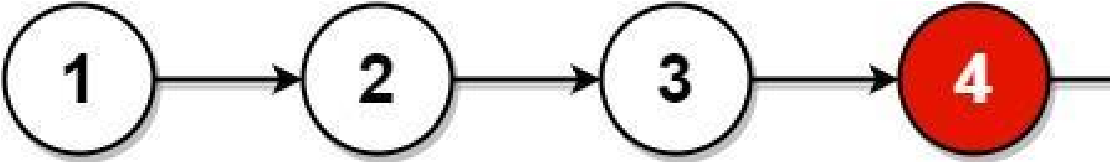
**Example 1:**



Input: head = [1,2,3,4,5]

Output: [3,4,5]

Explanation: The middle node of the list is node 3.

**Example 2:** 

Input: head = [1,2,3,4,5,6]

Output: [4,5,6]

Explanation: Since the list has two middle nodes with values 3 and 4, we return the second one.

Program:

/\*\*

 \* Definition for singly-linked list.

 \* struct ListNode {

 \*     int val;

 \*     ListNode \*next;

 \*     ListNode() : val(0), next(nullptr) {}

 \*     ListNode(int x) : val(x), next(nullptr) {}

 \*     ListNode(int x, ListNode \*next) : val(x), next(next) {}

 \* };

 \*/

class Solution {

public:

    ListNode\* middleNode(ListNode\* head) {

        ListNode\* fast = head;

        ListNode\* slow = head;

        while(fast!=NULL && fast->next!=NULL)

        {

            slow=slow->next;

            fast=fast->next->next;

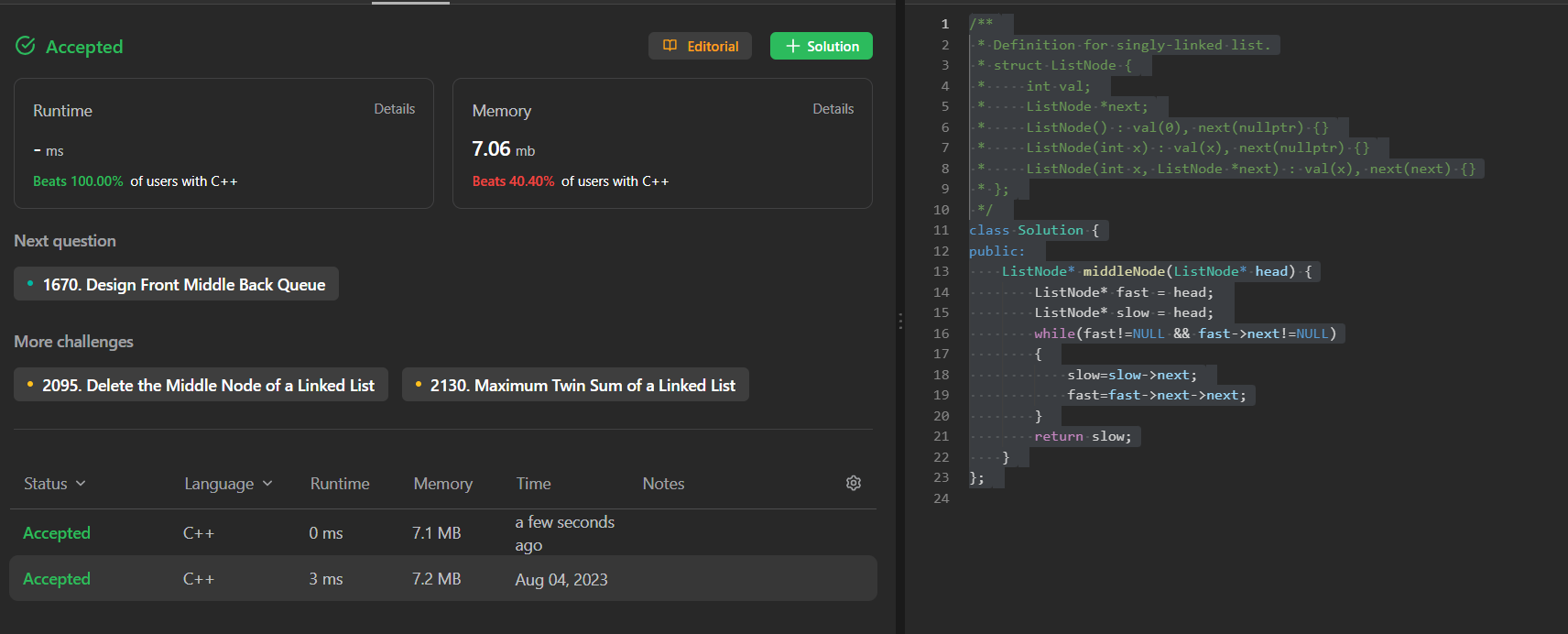
        }

        return slow;

    }

};

Output:-



CONCLUSION:-

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Student Signature Faculty Signature Marks