NAME – ANIKET SHARMA

REG. NO – RA1911003030436

BRANCH- CSE-I

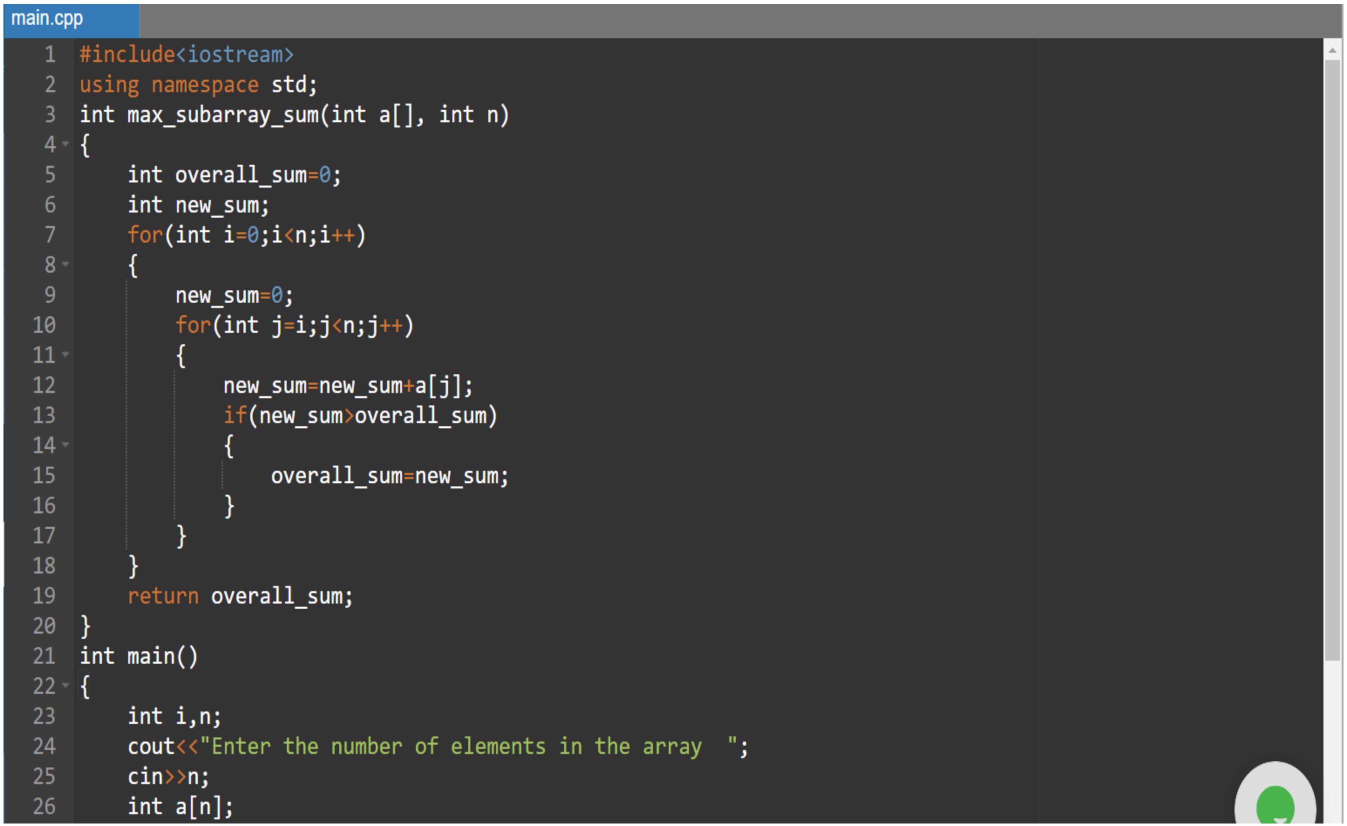
SUBJECT- Design and Analysis of Algorithms

SUBJECT CODE-18CSC204J

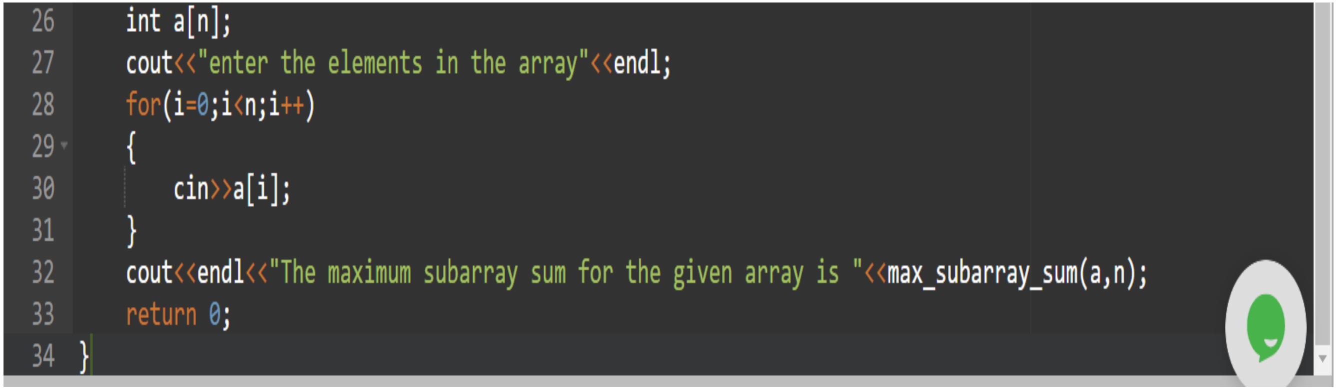
SUBMITTED TO- Mr. Himansu Sekhar Pattanayak

write c or c++ programs to solve maximum subarray problem using both brute force and divide & conquer approach.

Brute Force:-

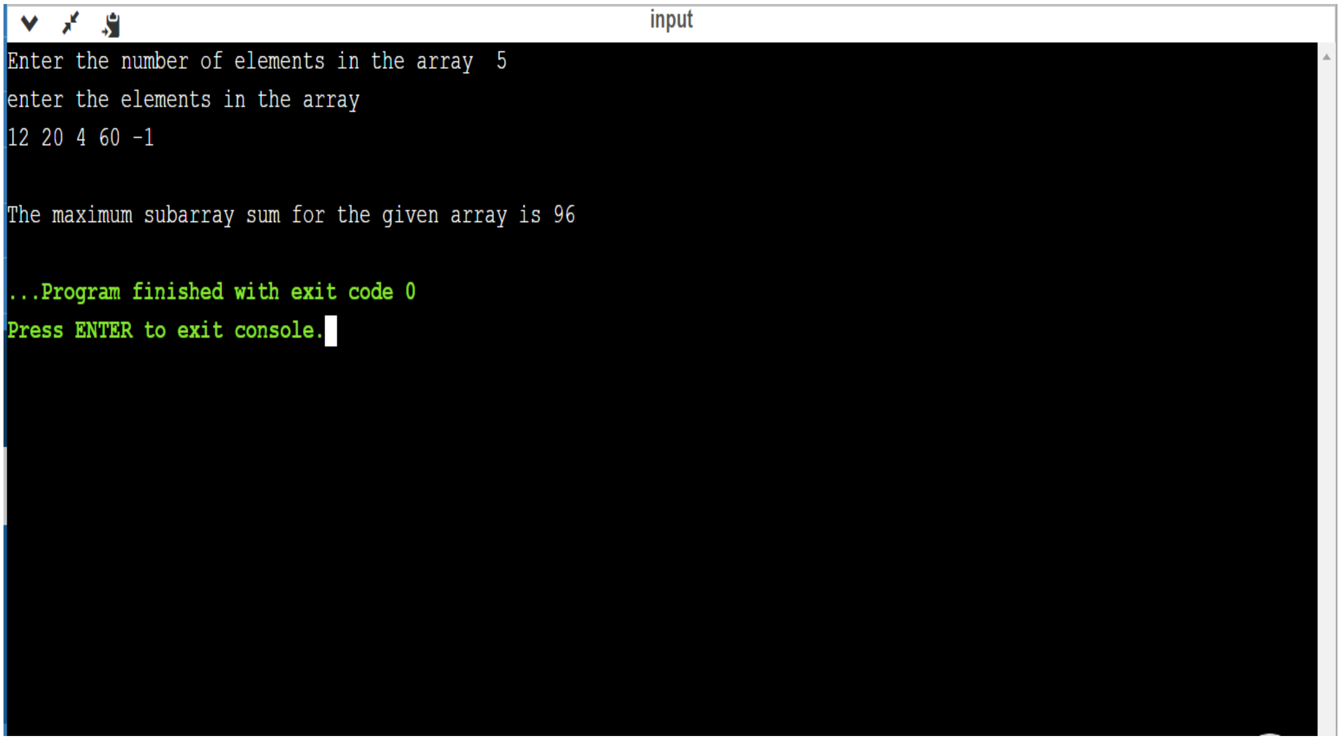


```cpp
#include<iostream>
using namespace std;
int max_subarray_sum(int a[], int n)
{
    int overall_sum=0;
    int new_sum;
    for(int i=0;i<n;i++)
    {
        new_sum=0;
        for(int j=i;j<n;j++)
        {
            new_sum=new_sum+a[j];
            if(new_sum>overall_sum)
            {
                overall_sum=new_sum;
            }
        }
    }
    return overall_sum;
}
int main()
{
    int i,n;
    cout<<"Enter the number of elements in the array  ";
    cin>>n;
    int a[n];
```



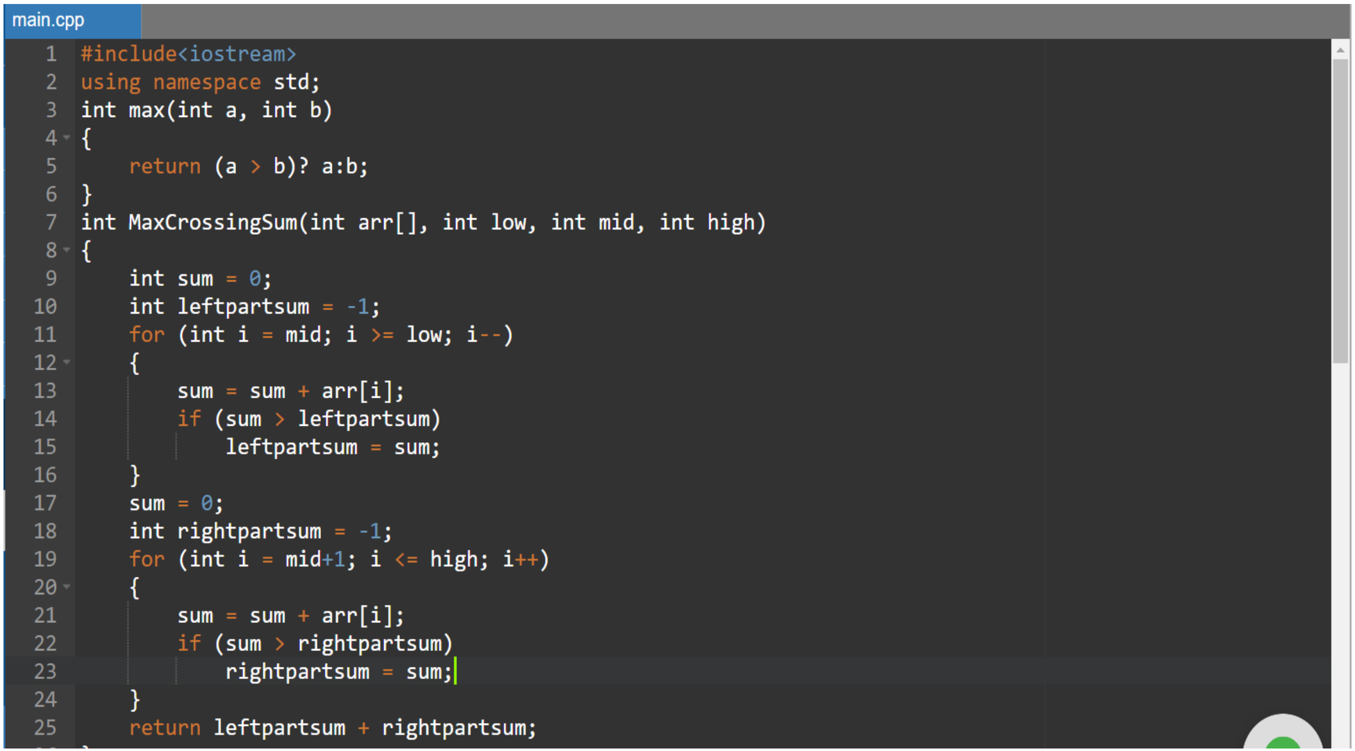
```cpp
    int a[n];
    cout<<"enter the elements in the array"<<endl;
    for(i=0;i<n;i++)
    {
        cin>>a[i];
    }
    cout<<endl<<"The maximum subarray sum for the given array is "<<max_subarray_sum(a,n);
    return 0;
}
```

OUTPUT:-

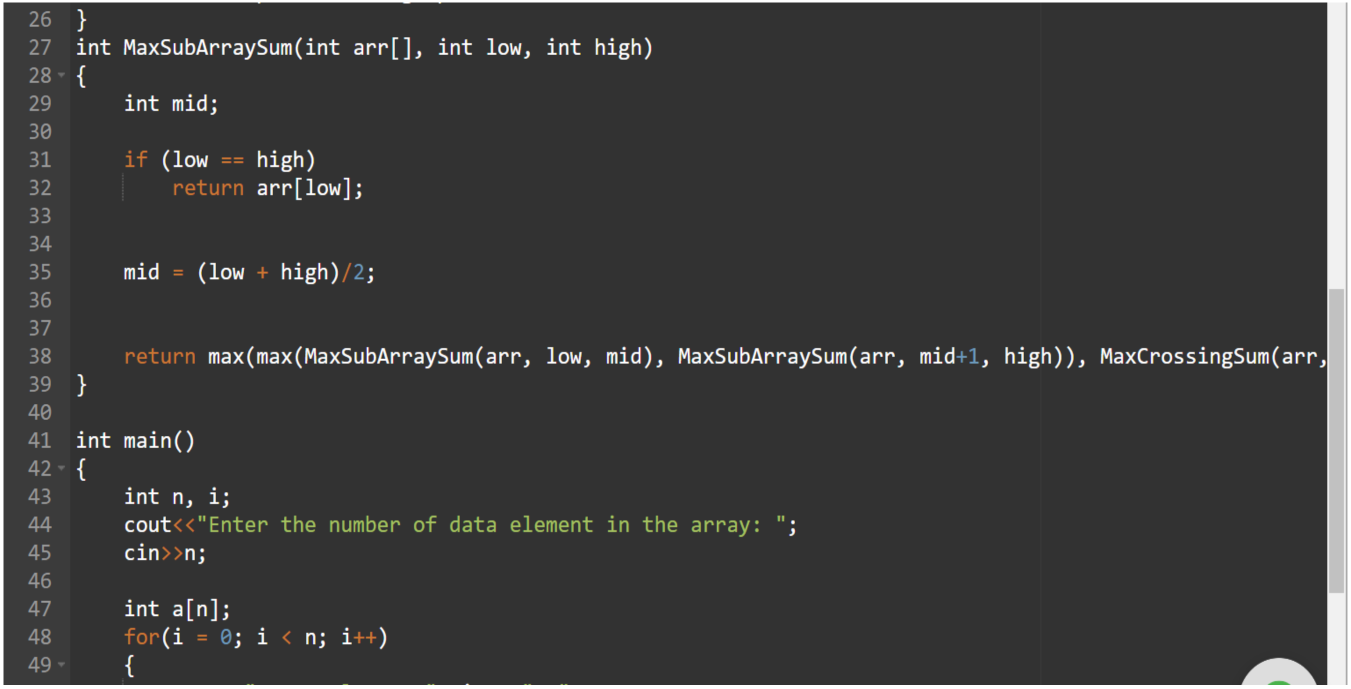


```
Enter the number of elements in the array  5
enter the elements in the array
12 20 4 60 -1

The maximum subarray sum for the given array is 96

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

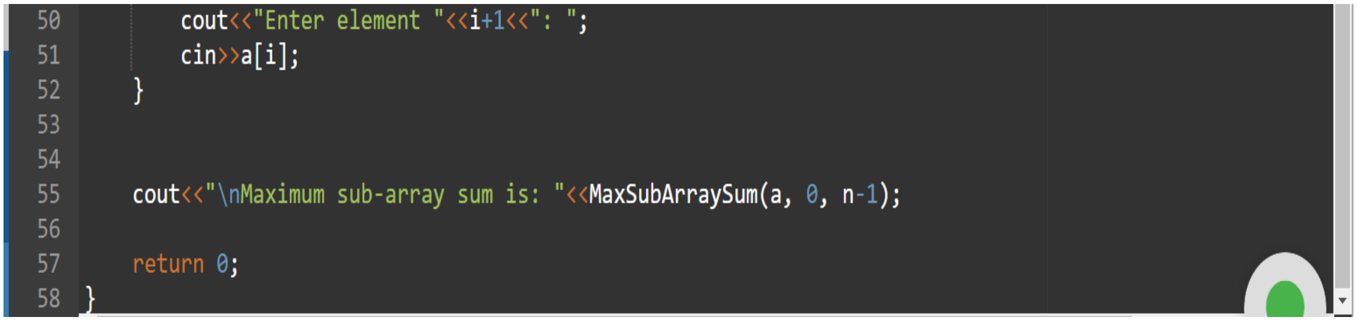
DIVIDE AND CONQUOR APPROACH:-



```cpp
#include<iostream>
using namespace std;
int max(int a, int b)
{
    return (a > b)? a:b;
}
int MaxCrossingSum(int arr[], int low, int mid, int high)
{
    int sum = 0;
    int leftpartsum = -1;
    for (int i = mid; i >= low; i--)
    {
        sum = sum + arr[i];
        if (sum > leftpartsum)
            leftpartsum = sum;
    }
    sum = 0;
    int rightpartsum = -1;
    for (int i = mid+1; i <= high; i++)
    {
        sum = sum + arr[i];
        if (sum > rightpartsum)
            rightpartsum = sum;
    }
    return leftpartsum + rightpartsum;
```

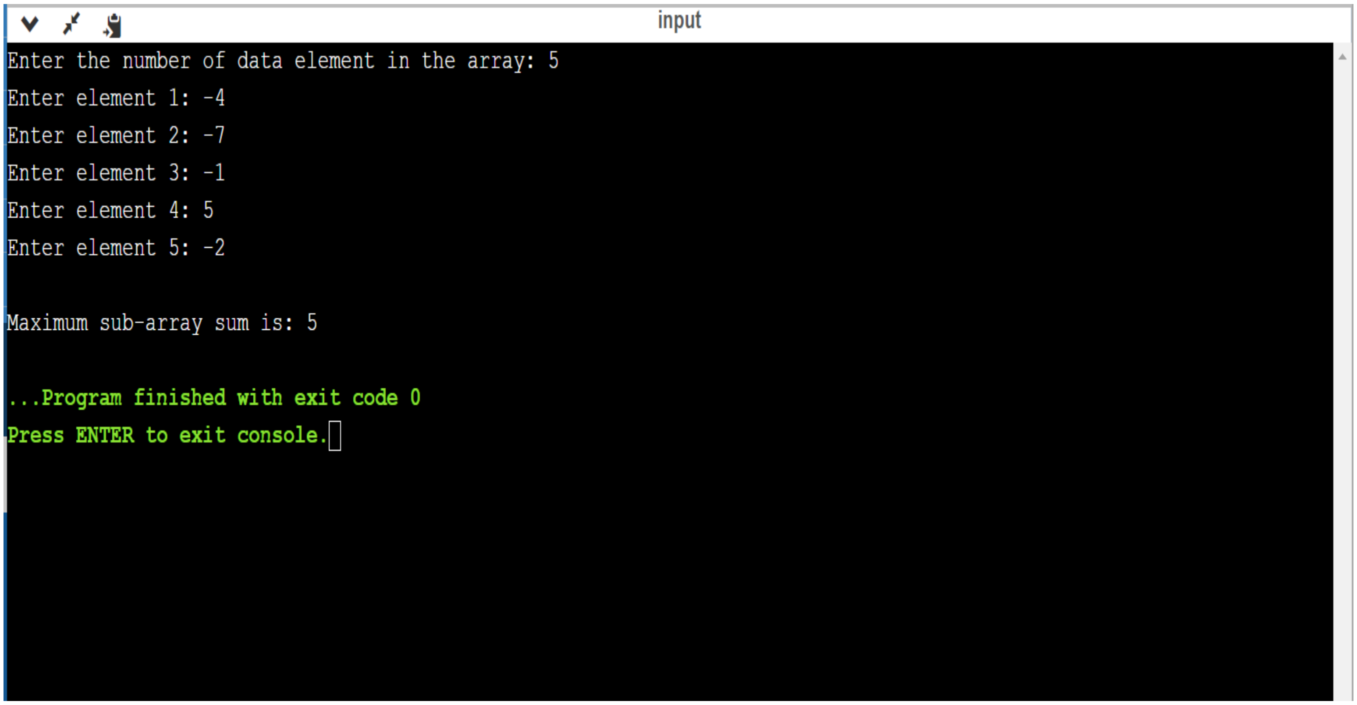


```cpp
}
int MaxSubArraySum(int arr[], int low, int high)
{
    int mid;

    if (low == high)
        return arr[low];


    mid = (low + high)/2;


    return max(max(MaxSubArraySum(arr, low, mid), MaxSubArraySum(arr, mid+1, high)), MaxCrossingSum(arr,
}

int main()
{
    int n, i;
    cout<<"Enter the number of data element in the array: ";
    cin>>n;

    int a[n];
    for(i = 0; i < n; i++)
    {
```



```
        cout<<"Enter element "<<i+1<<": ";
        cin>>a[i];
    }


    cout<<"\nMaximum sub-array sum is: "<<MaxSubArraySum(a, 0, n-1);

    return 0;
}
```

OUTPUT:-



```
Enter the number of data element in the array: 5
Enter element 1: -4
Enter element 2: -7
Enter element 3: -1
Enter element 4: 5
Enter element 5: -2

Maximum sub-array sum is: 5

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