**CSE2012**

**DAA LAB**

Name: Preyash

Registration Number: 20BPS1022

Ex 4: MAX SUB ARRAY CALCULATION

**BRUTE FORCE:**

**Code:**

#include <bits/stdc++.h>

#include<climits>

using namespace std;

int max\_subarraysum(int arr[],int n)

{

    int ans=INT\_MIN;

    for(int sas=1;sas<=n;++sas)

        {

        for(int si=0;si<n;++si)

        {

            if((si+sas)>n)

            {

                break;

            }

            int sum=0;

            for(int i=0;i<(si+sas);i++)

            sum+=arr[i];

            ans=max(ans,sum);

        }

    }

    return ans;

}

int main()

{

    int n;

    int arr1[10];

    cin>>n;

    for (int i=0;i<n;i++)

    {

        cin>>arr1[i];

    }

    int maxi;

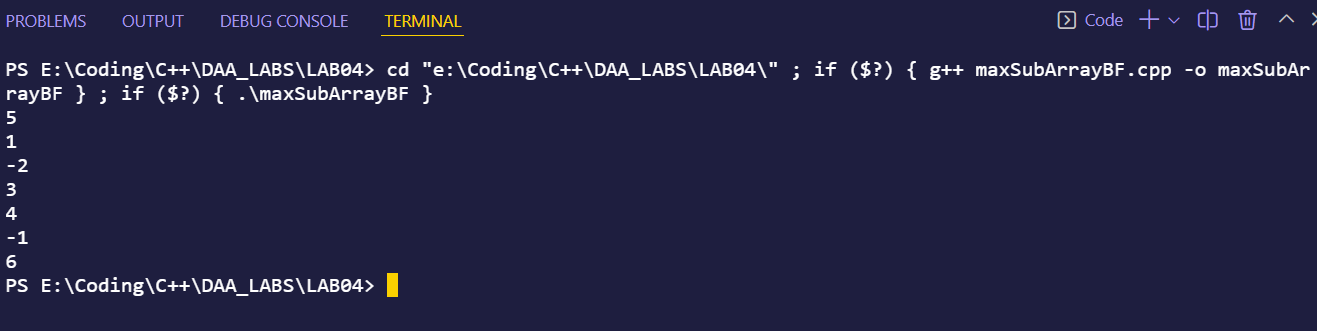
    maxi=max\_subarraysum(arr1,n);

    cout<<maxi;

    return 0;

}

**Output:**



**BRUTE FORCE ANOTHER APPROACH:**

**Code:**

#include <bits/stdc++.h>

#include<climits>

using namespace std;

int max\_subarraysum(int arr[],int n)

{

    int smax=INT\_MIN;

    for(int i=1;i<=n;i++)

    {

        int sum=0;

        for(int j=i;j<n;j++)

        {

            sum+=arr[j];

            smax=max(smax,sum);

        }

    }

    return smax;

}

int main()

{

    int n;

    int arr1[10];

    cin>>n;

    for (int i=0;i<n;i++)

    {

        cin>>arr1[i];

    }

    int maxi;

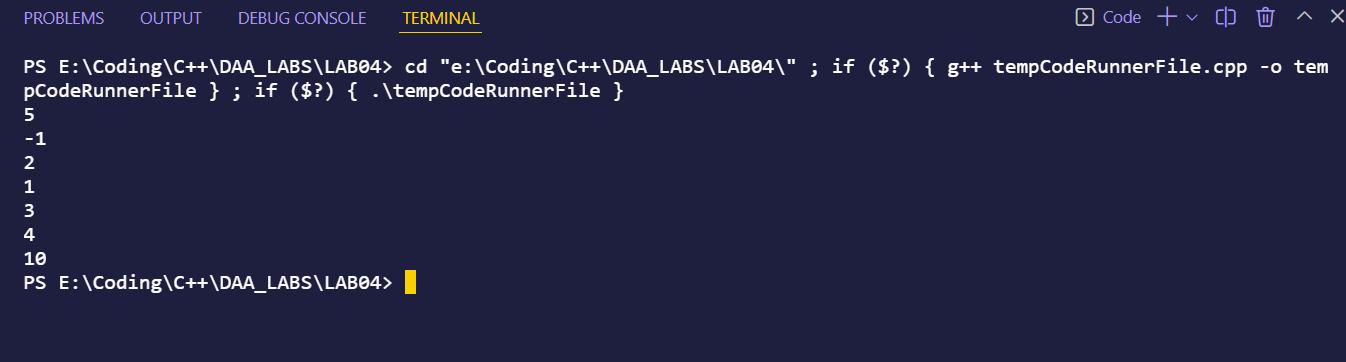
    maxi=max\_subarraysum(arr1,n);

    cout<<maxi;

    return 0;

}

**Output:**



**DIVIDE AND CONQUER:**

**Code:**

#include <bits/stdc++.h>

#include<climits>

using namespace std;

int max(int a, int b, int c)

 {

      return max(max(a, b), c);

}

int maxCrossingsum(int arr[],int l,int m,int h)

{

    int leftsum=INT\_MIN;

    int sum=0;

    for(int i=m;i>=l;i--)

    {

        sum+=arr[i];

        if(sum>leftsum)

        {

            leftsum=sum;

        }

    }

    int rightsum=INT\_MIN;

    sum=0;

    for(int i=m+1;i<=h;i++)

    {

        sum+=arr[i];

        if(sum>rightsum)

        {

            rightsum=sum;

        }

    }

    return max(leftsum+rightsum,leftsum,rightsum);

}

int max\_subarray(int arr[],int l,int h)

{

    int leftsum,rightsum,crosssum;

    if(h==l)

    {

        return arr[l];

    }

    int m=(l+h)/2;

    return max(max\_subarray(arr,l,m),maxCrossingsum(arr,l,m,h),max\_subarray(arr,m+1,h));

}

int main()

{

    int n;

    int arr[10];

    cin>>n;

    for(int i=0;i<n;i++)

    {

        cin>>arr[i];

    }

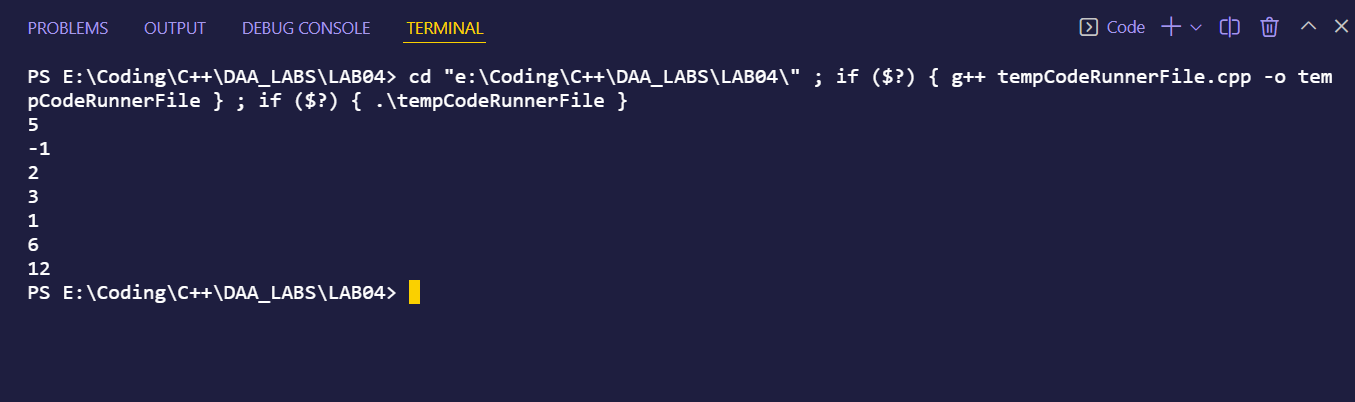
    int maximum=max\_subarray(arr,0,n-1);

    cout<<maximum;

    return 0;

}

**Output:**



**AUXILLIARY ARRAY:**

**Code:**

#include <bits/stdc++.h>

#include<climits>

using namespace std;

int max\_subarraysum(int arr[],int n)

{

    int meh[n];

    meh[0]=arr[0];

    for(int i=1;i<=n-1;i++)

    {

        if(meh[i-1]>0)

        {

            meh[i]=arr[i]+meh[i-1];

        }

        else

        meh[i]=arr[i];

    }

    int ans=INT\_MIN;

    for(int i=0;i<=n-1;i++)

    {

        ans=max(ans,meh[i]);

    }

    return ans;

}

int main()

{

    int n;

    int arr1[10];

    cin>>n;

    for (int i=0;i<n;i++)

    {

        cin>>arr1[i];

    }

    int maxi;

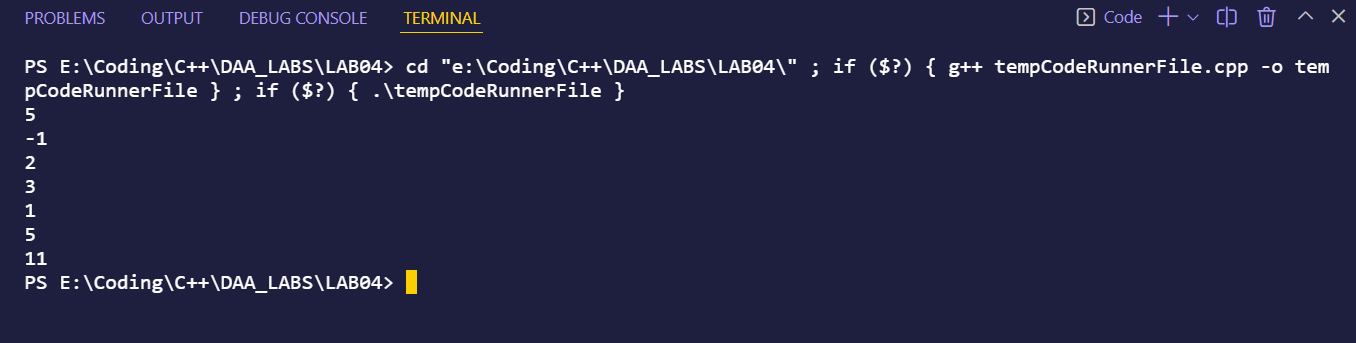
    maxi=max\_subarraysum(arr1,n);

    cout<<maxi;

    return 0;

}

**Output:**



**KADANES ALGORITHM:**

**Code:**

#include<bits/stdc++.h>

using namespace std;

int kadanes(int arr[],int n)

{

    int maxsofar=INT\_MIN;

    int maxending=0;

    for(int i=0;i<n;i++)

    {

        maxending+=arr[i];

        if(maxsofar<maxending)

        {

            maxsofar=maxending;

        }

        if(maxending<0)

        {

            maxending=0;

        }

    }

    return maxsofar;

}

int main()

{

 int n,arr[10];

 cin>>n;

 for(int i=0;i<n;i++)

 {

 cin>>arr[i];

 }

 int maximum=kadanes(arr,n);

 cout<<maximum;

 return 0;

}

**Output:**

