_036)_Max_Sum_of_K_Consecutive_Element.cpp

```cpp
#include<iostream>
using namespace std;

int maxSumOf_K_Element(int arr[], int k, int n)
{
    int max_sum=0;
    for(int i=0; i+k-1<n; i++)
    {
        int sum = 0;
        for(int j=0; j<k; j++)
            sum += arr[i+j];
        max_sum = max(sum, max_sum);
    }
    return max_sum;
}

int main()
{
    int arr[] = {1, 8, 30, -5, 20, 7}, k=3, n=6;
    cout<<"Max Sum of 3 Consecutive Element = "<<maxSumOf_K_Element(arr, k, n)<<endl;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36-...
Max Sum of 3 Consecutive Element = 45

Process returned 0 (0x0)   execution time : 0.297 s
Press any key to continue.
```

_037)_Max_Sum_of_K_Consecutive_Element.cpp

```cpp
#include<iostream>
using namespace std;

int maxSumOf_K_Element(int arr[], int k, int n)
{
    int curr_sum = 0;
    for(int i=0; i<k; i++)
        curr_sum += arr[i];
    int max_sum = curr_sum;
    for(int i=k; i<n; i++)
    {
        curr_sum += (arr[i] - arr[i-k]);
        max_sum = max(max_sum, curr_sum);
    }
    return max_sum;
}

int main()
{
    int arr[] = {1, 8, 30, -5, 20, 7}, k=3, n=6;
    cout<<"Max Sum of 3 Consecutive Element = "<<maxSumOf_K_Element(arr, k, n)<<endl;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Arr...
Max Sum of 3 Consecutive Element = 45

Process returned 0 (0x0)   execution time : 0.483 s
Press any key to continue.
```

_038)_Find_SubArray_of_Given_Sum.cpp

```cpp
#include<iostream>
using namespace std;

int subArraySum(int arr[], int n, int given_sum)
{
    for(int i=0; i<n; i++)
    {
        int sum = 0;
        for(int j=i; j<n; j++)
        {
            sum += arr[j];
            if(sum == given_sum)
            {
                cout<<"Sum found between indexes "<< i <<" and "<< j <<endl;
                return true;
            }
        }
    }
    cout<<"No subArray found"<<endl;
}

int main()
{
    int arr[] = {1, 4, 20, 3, 10, 5}, n=6, given_sum=23;
    subArraySum(arr, n, given_sum);
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36...
Sum found between indexes 2 and 3

Process returned 0 (0x0)   execution time : 0.212 s
Press any key to continue.
```

_039)_Find_SubArray_of_Given_Sum.cpp

```cpp
#include<iostream>
using namespace std;

int subArraySum(int arr[], int n, int given_sum)
{
    int curr_sum = arr[0], start=0;
    for(int end=1; end<n; end++)
    {
        while(curr_sum > given_sum && start < end-1)
        {
            curr_sum -= arr[start];
            start++;
        }
        if(curr_sum == given_sum)
        {
            cout<<"Sum found between indexes "<< start <<" and "<< end-1 <<endl;
            return true;
        }
        if(end<n)
            curr_sum += arr[end];
    }
    cout<<"No subArray found"<<endl;
}

int main()
{
    int arr[] = {1, 4, 20, 3, 10, 5}, n=6, given_sum=23;
    subArraySum(arr, n, given_sum);
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36-44]...
Sum found between indexes 2 and 3

Process returned 0 (0x0)   execution time : 0.218 s
Press any key to continue.
```

_040)_Print_N_bonacci_Numbers.cpp

```cpp
#include<iostream>
using namespace std;

void bonacciseries(long n, int m)
{
    cout<< "N-bonacci Numbers = " ;
    int a[m] = { 0 };
    a[n - 1] = 1;
    a[n] = 1;
    for (int i = n + 1; i < m; i++)
        a[i] = 2 * a[i - 1] - a[i - n - 1];
    for (int i = 0; i < m; i++)
        cout<< a[i] << " ";
    cout<<endl;
}

int main()
{
    int N = 5, M = 15;
    bonacciseries(N, M);
    return 0;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36-...
N-bonacci Numbers = 0 0 0 0 1 1 2 4 8 16 31 61 120 236 464

Process returned 0 (0x0)   execution time : 0.194 s
Press any key to continue.
```

_041)_GetSum_on_Given_Queries.cpp

```cpp
#include<iostream>
using namespace std;

int prefix_sum[10000];
void preSum(int arr[], int n)
{
    prefix_sum[0] = arr[0];

    for(int i = 1; i < n; i++)
    {
        prefix_sum[i] = prefix_sum[i - 1] + arr[i];
    }
}

int getSum(int prefix_sum[], int l, int r)
{
    if(l != 0)
        return prefix_sum[r] - prefix_sum[l - 1];
    else
        return prefix_sum[r];
}



int main()
{
    int arr[] = {2, 8, 3, 9, 6, 5, 4}, n = 7;
    preSum(arr, n);
    cout<<"Sum of subarray index from 1 to 3 = "<<getSum(prefix_sum, 1, 3)<<endl;
    cout<<"Sum of subarray index from 0 to 2 = "<<getSum(prefix_sum, 0, 2)<<endl;
}
```

```
"C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36-44])\_041)_Get...
Sum of subarray index from 1 to 3 = 20
Sum of subarray index from 0 to 2 = 13

Process returned 0 (0x0)   execution time : 0.156 s
Press any key to continue.
```

_042)_Find_Equilibrium_Point_in_an_Array.cpp

```cpp
#include<iostream>
using namespace std;

bool checkEquilibrium(int arr[], int n)
{
    for(int i  = 0; i < n; i++)
    {
        int l_sum = 0, r_sum = 0;
        for(int j = 0; j < i; j++)
            l_sum += arr[j];
        for(int j = i + 1; j < n; j++)
            r_sum += arr[j];
        if(l_sum == r_sum)
            return true;
    }
    return false;
}



int main()
{
    int arr[] = {3, 4, 8, -9, 20, 6}, n = 6;
    cout<<checkEquilibrium(arr, n)? "true" : "false";
    cout<<endl;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[3...
1

Process returned 0 (0x0)   execution time : 0.849 s
Press any key to continue.
```

_043)_Find_Equilibrium_Point_in_an_Array.cpp

```cpp
#include<iostream>
using namespace std;

bool checkEquilibrium(int arr[], int n)
{
    int sum = 0;
    for(int i = 0; i < n; i++)
        sum += arr[i];
    int l_sum = 0;
    for(int i = 0; i < n; i++)
    {
        if(l_sum == sum - arr[i])
            return true;
        l_sum += arr[i];
        sum -= arr[i];
    }
    return false;
}

int main()
{
    int arr[] = {4, 2, 2}, n = 3;
    cout<<checkEquilibrium(arr, n)? "true" : "false";
    cout<<endl;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Arra...
0

Process returned 0 (0x0)   execution time : 0.456 s
Press any key to continue.
```

_044)_In_Two_Array_Find_Max_Occuring_Element.cpp

```cpp
#include<iostream>
#include<bits/stdc++.h>
using namespace std;

int maxOcc(int L[], int R[], int n)
{
        int arr[1000];
        memset(arr, 0, sizeof(arr));
        for(int i = 0; i < n; i++)
        {
            arr[L[i]]++;
            arr[R[i] + 1]--;
        }
        int maxm = arr[0], res = 0;
        for(int i = 1; i < 1000; i++)
        {
            arr[i] += arr[i - 1];
            if(maxm < arr[i])
            {
                maxm = arr[i];
                res = i;
            }
        }
        return res;
}

int main()
{
    int L[] = {1, 2, 3}, R[] = {3, 5, 7}, n = 3;
    cout<<"Maximum Occuring Element in both Array = "<<maxOcc(L, R, n)<<endl;
}
```

```
Select "C:\Users\Akash Singh\Documents\Coding\' CHALLENGE '\365-Days-of-Code\175). Complete DSA (Array[36-44])\_044)_In_T...
Maximum Occuring Element in both Array = 3

Process returned 0 (0x0)   execution time : 0.347 s
Press any key to continue.
```