题目描述

有一个N个整数的数组，和一个长度为M的窗口，窗口从数组内的第一个数开始滑动直到窗口不能滑动为止，

每次窗口滑动产生一个窗口和（窗口内所有数的和），求窗口滑动产生的所有窗口和的最大值。

输入描述

- 第一行输入一个正整数N，表示整数个数。（0<N<100000）
- 第二行输入N个整数，整数的取值范围为[-100,100]。
- 第三行输入一个正整数M，M代表窗口的大小，M<=100000，且M<=N。

输出描述

- 窗口滑动产生所有窗口和的最大值。

用例

输入	6 10 20 30 15 23 12 3
输出	68
说明	窗口长度为3，窗口滑动产生的窗口和分别为 10+20+30=60， 20+30+15=65， 30+15+23=68， 15+23+12=50， 所以窗口滑动产生的所有窗口和的最大值为68。

题目解析

此题应该是考察我们如何计算新的滑动窗口的值，一般有两种方式：

1、将滑动窗口中的元素值相加求和

2、基于前一个滑动窗口的和sum，计算新的滑动窗口的和，原理如下

如上图所示，第二个滑动窗口，相当于第一个滑动窗口减去10，加上15。

公式如下：i为第二个滑动窗口的起点索引，sum为第一个滑动窗口的和

sum = sum - arr[i - 1] + arr[i + m - 1];

JavaScript算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

const lines = [];
rl.on("line", (line) => {
  lines.push(line);

  if (lines.length === 3) {
    const n = lines[0] - 0;
    const arr = lines[1].split(" ").map(Number);
    const m = lines[2] - 0;

    console.log(getMaxWindowSum(arr, n, m));

    lines.length = 0;
  }
});

function getMaxWindowSum(arr, n, m) {
  let sum = arr.slice(0, m).reduce((p, c) => p + c);
  let ans = sum;

  for (let i = 1; i <= n - m; i++) {
    sum += arr[i + m - 1] - arr[i - 1];
    ans = Math.max(ans, sum);
  }

  return ans;
}
```

Java算法源码

```java
import java.util.Scanner;

public class Main {
  // 输入获取
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt();

    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
      arr[i] = sc.nextInt();
    }

    int m = sc.nextInt();

    System.out.println(getResult(n, arr, m));
  }

  // 算法入口
  public static int getResult(int n, int[] arr, int m) {
    int sum = 0;
    for (int i = 0; i < m; i++) { // 初始滑窗内和
      sum += arr[i];
    }

    int ans = sum;

    for (int i = 1; i <= n - m; i++) {
      sum += arr[i + m - 1] - arr[i - 1]; // 基于初始滑窗进行差异求和，避免O(n)求和
      ans = Math.max(ans, sum);
    }

    return ans;
  }
}
```

Python算法源码

```python
# 输入获取
n = int(input())
arr = list(map(int, input().split()))
m = int(input())


# 算法入口
def getResult():
    sumV = sum(arr[:m])
    ans = sumV

    for i in range(1, n-m+1):
        sumV += arr[i+m-1] - arr[i-1]
        ans = max(ans, sumV)

    return ans


# 算法调用
print(getResult())
```