题目描述

给定一个数组 arr 和整数 k，计算这个数组中所有长度为 k 的连续子数组的平均值。请输出一个新数组，其中第 i 个元素是 arr[i] 到 arr[i+k-1] 这 k 个数的平均值（保留两位小数）。

如果数组长度小于 k，则输出一个空数组。

例如，arr = [1, 3, 5, 7, 9], k = 3，则输出 [3.00, 5.00, 7.00]。

输入格式

一行输入包含一个以空格分隔的整数序列和一个整数 k。数组长度不超过 1000，k 为正整数。

输出格式

一行输出一个以空格分隔的浮点数序列，每个数保留两位小数。

示例

输入：

```
1 2 3 4 5 6 7 8 9 10
5
```

输出：

```
3.00 4.00 5.00 6.00 7.00 8.00
```

解释： 数组长度为 10，k 为 5，所以输出长度为 10 - 5 + 1 = 6。

Java 解法解析

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.List;

public class Main {
    // 主方法，程序的入口点
    public static void main(String[] args) {
        // 创建一个新的 Scanner 对象，用于从标准输入（控制台）读取数据
        Scanner scanner = new Scanner(System.in);

        // 读取一行输入，并将其按空格分割成字符串数组
        String[] parts = scanner.nextLine().split(" ");

        // 创建一个新的 ArrayList，用于存储从输入中解析出的整数
        ArrayList<Integer> numbers = new ArrayList<>();

        // 遍历刚才读取的字符串数组
        for (String part : parts) {
            // 检查当前部分不为空字符串
            if (!part.isEmpty()) {
                // 将当前部分字符串转换为整数，并添加到 numbers 列表中
                numbers.add(Integer.parseInt(part));
            }
        }

        // 读取一行输入，并将其解析为整数，作为窗口大小 k
        int k = scanner.nextInt();

        // 调用 calculate 方法来计算结果，并将结果存储在 result 列表中
        List<Double> result = calculate(numbers, k);

        // 遍历结果列表
        for (int i = 0; i < result.size(); i++) {
            // 打印当前结果，保留两位小数
            System.out.printf("%.2f", result.get(i));
            // 如果不是最后一个结果，则打印一个空格
            if (i < result.size() - 1) {
                System.out.print(" ");
            }
        }
        // 打印一个换行符
        System.out.println();
        // 关闭 scanner 对象，释放资源
        scanner.close();
    }

    // 计算滑动窗口平均值的方法
    public static List<Double> calculate(List<Integer> arr, int k) {
        // 创建一个新的 ArrayList，用于存储结果
        List<Double> result = new ArrayList<>();
        // 获取输入数组的长度
        int n = arr.size();
        // 检查窗口大小 k 是否大于数组长度，或者 k 是否小于等于 0
        if (k > n || k <= 0) {
            // 如果是，则返回空结果列表
            return result;
        }
        // 计算第一个窗口的总和
        double sum = 0;
        for (int i = 0; i < k; i++) {
            sum += arr.get(i);
        }
        // 将第一个窗口的平均值添加到结果列表中
        result.add(sum / k);
        // 开始滑动窗口，从第 k 个元素开始
        for (int i = k; i < n; i++) {
            // 减去窗口最左边的元素
            sum -= arr.get(i - k);
            // 加上当前元素
            sum += arr.get(i);
            // 将当前窗口的平均值添加到结果列表中
            result.add(sum / k);
        }
        // 返回结果列表
        return result;
    }
}
```

C++解法解析

```
#include <iostream>
#include <vector>
#include <sstream>
#include <string>
#include <algorithm>
#include <cstdlib>
#include <cstdio>

// 函数声明
std::vector<double> calculate(std::vector<int> arr, int k);

// 主函数
int main() {
    // 创建一个字符串流对象，用于从标准输入读取数据
    std::stringstream ss;
    // 将标准输入的内容读取到字符串流中
    ss << std::cin.rdbuf();
    // 从字符串流中获取一个字符串，用于存储输入的第一行
    std::string line;
    ss >> line;
    // 创建一个字符串流对象，用于解析输入的第一行
    std::stringstream lss(line);
    // 创建一个整数向量，用于存储从输入中解析出的数字
    std::vector<int> numbers;
    // 循环从字符串流中读取整数，直到失败
    int num;
    while (lss >> num) {
        // 将读取到的整数添加到 numbers 向量中
        numbers.push_back(num);
    }
    // 从字符串流中读取窗口大小 k
    int k;
    ss >> k;
    // 调用 calculate 函数来计算结果
    std::vector<double> result = calculate(numbers, k);
    // 遍历结果向量
    for (int i = 0; i < result.size(); i++) {
        // 打印当前结果，保留两位小数
        printf("%.2f", result[i]);
        // 如果不是最后一个结果，则打印一个空格
        if (i < result.size() - 1) {
            printf(" ");
        }
    }
    // 打印一个换行符
    printf("\n");
    // 返回 0 表示程序成功结束
    return 0;
}

// 计算滑动窗口平均值的函数
std::vector<double> calculate(std::vector<int> arr, int k) {
    // 创建一个双精度浮点数向量，用于存储结果
    std::vector<double> result;
    // 获取输入数组的长度
    int n = arr.size();
    // 检查窗口大小 k 是否大于数组长度，或者 k 是否小于等于 0
    if (k > n || k <= 0) {
        // 如果是，则返回空结果向量
        return result;
    }
    // 计算第一个窗口的总和
    double sum = 0;
    for (int i = 0; i < k; i++) {
        sum += arr[i];
    }
    // 将第一个窗口的平均值添加到结果向量中
    result.push_back(sum / k);
    // 开始滑动窗口，从第 k 个元素开始
    for (int i = k; i < n; i++) {
        // 减去窗口最左边的元素
        sum -= arr[i - k];
        // 加上当前元素
        sum += arr[i];
        // 将当前窗口的平均值添加到结果向量中
        result.push_back(sum / k);
    }
    // 返回结果向量
    return result;
}
```

C#解法解析

```
using System;
using System.Collections.Generic;
using System.Linq;

// 定义一个名为 Program 的类
class Program
{
    // 主方法，程序的入口点
    static void Main(string[] args)
    {
        // 创建一个新的 Scanner 对象，用于从标准输入（控制台）读取数据
        Scanner scanner = new Scanner();

        // 读取一行输入，并将其按空格分割成字符串数组
        string[] parts = scanner.NextLine().Split(' ');

        // 创建一个新的 List<int>，用于存储从输入中解析出的整数
        List<int> numbers = new List<int>();

        // 遍历刚才读取的字符串数组
        foreach (string part in parts)
        {
            // 检查当前部分不为空字符串
            if (!part.Equals(""))
            {
                // 将当前部分字符串转换为整数，并添加到 numbers 列表中
                numbers.Add(int.Parse(part));
            }
        }

        // 读取一行输入，并将其解析为整数，作为窗口大小 k
        int k = scanner.NextInt();

        // 调用 Calculate 方法来计算结果，并将结果存储在 result 列表中
        List<double> result = Calculate(numbers, k);

        // 遍历结果列表
        for (int i = 0; i < result.Count; i++)
        {
            // 打印当前结果，保留两位小数
            Console.Write($"{result[i]:0.00}");
            // 如果不是最后一个结果，则打印一个空格
            if (i < result.Count - 1)
            {
                Console.Write(" ");
            }
        }
        // 打印一个换行符
        Console.WriteLine();
        // 关闭 scanner 对象，释放资源
        scanner.Close();
    }

    // 计算滑动窗口平均值的方法
    static List<double> Calculate(List<int> arr, int k)
    {
        // 创建一个新的 List<double>，用于存储结果
        List<double> result = new List<double>();
        // 获取输入数组的长度
        int n = arr.Count;
        // 检查窗口大小 k 是否大于数组长度，或者 k 是否小于等于 0
        if (k > n || k <= 0)
        {
            // 如果是，则返回空结果列表
            return result;
        }
        // 计算第一个窗口的总和
        double sum = 0;
        for (int i = 0; i < k; i++)
        {
            sum += arr[i];
        }
        // 将第一个窗口的平均值添加到结果列表中
        result.Add(sum / k);
        // 开始滑动窗口，从第 k 个元素开始
        for (int i = k; i < n; i++)
        {
            // 减去窗口最左边的元素
            sum -= arr[i - k];
            // 加上当前元素
            sum += arr[i];
            // 将当前窗口的平均值添加到结果列表中
            result.Add(sum / k);
        }
        // 返回结果列表
        return result;
    }
}
```