

# 【华为OD机考 统一考试机试C卷】寻找连续区间/数组连续和（C++ Java JavaScript Python C语言）

## 华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷

目前在考C卷，经过两个月的收集整理，**C卷真题已基本整理完毕**

抽到原题的概率为2/3到3/3，**也就是最少抽到两道原题。请注意：大家刷完C卷真题，最好要把B卷的真题刷一下，因为C卷的部分真题来自B卷。**

另外订阅专栏还可以联系笔者开通在线 **OJ** 进行刷题，提高刷题效率。

**真题目录：**华为OD机考机试 真题目录（C卷 + D卷 + B卷 + A卷） + 考点说明

**专栏：**2023华为OD机试( B卷+C卷+D卷) (C++JavaJSPy)

**华为OD面试真题精选：**华为OD面试真题精选

**在线OJ：**点击立即刷题，模拟真实机考环境

## 题目描述

给定一个含有N个正整数的数组, 求出有多少个连续区间（包括单个正整数），它们的和大于等于x。

## 输入描述

第一行两个整数N x ( $0 < N \leq 100000$ ,  $0 \leq x \leq 10000000$ )

第二行有N个正整数（每个正整数小于等于100）。

## 输出描述

输出一个整数，表示所求的个数。

## 用例1

输入

```
1 | 3 7
2 |
```

3 4 7

输出

1 | 4

第一行的3表示第二行数组输入3个数，第一行的7是比较数，用于判断连续数组是否大于该数；组合为 3 + 4; 3 + 4 + 7; 4 + 7; 7; 都大于等于指定的7；所以共四组。

## 用例2

输入

```
1 | 10 10000
2 | 1 2 3 4 5 6 7 8 9 10
```

输出

1 | 0

所有元素的和小于10000，所以返回0。

## C++

```
1 | #include <iostream>
2 | using namespace std;
3 |
4 | int main() {
5 |     int n, x;
6 |     cin >> n >> x;
7 |
8 |     int nums[n];
9 |     for (int i = 0; i < n; i++) cin >> nums[i];
10 |
11 |     int left = 0; // 滑动窗口的左端点
12 |
```

```

--
13     int right = 0; // 滑动窗口的右端点
14     int count = 0; // 记录连续区间个数
15     int sum = 0; // 记录当前区间的和
16
17     while (right < n) {
18         sum += nums[right];
19
20         while (sum >= x) {
21             // 如果当前区间和大于等于x, 那么以left为起点的所有连续区间都符合要求
22             count += n - right;
23             sum -= nums[left];
24             left++;
25         }
26
27         right++;
28     }
29
30     cout << count << endl;
31
32     return 0;
33 }

```

## JavaScript

```

1  const readline = require('readline');
2
3  const rl = readline.createInterface({
4      input: process.stdin,
5      output: process.stdout
6  });
7
8  let n = 0;
9  let x = 0;
10 let nums = [];
11
12 rl.on('line', (line) => {
13     if (!n) {
14         [n, x] = line.trim().split(' ').map(Number);
15     } else {
16

```

```

16     nums = line.trim().split(' ').map(Number);
17
18     let left = 0; // 滑动窗口的左端点
19     let right = 0; // 滑动窗口的右端点
20     let count = 0; // 记录连续区间个数
21     let sum = 0; // 记录当前区间的和
22
23     while (right < n) {
24         sum += nums[right];
25
26         while (sum >= x) {
27             // 如果当前区间和大于等于x, 那么以left为起点的所有连续区间都符合要求
28             count += n - right;
29             sum -= nums[left];
30             left++;
31         }
32
33         right++;
34     }
35
36     console.log(count);
37 }
38 });

```

## python

```

1 import sys
2
3 n, x = map(int, sys.stdin.readline().split())
4
5 nums = list(map(int, sys.stdin.readline().split()))
6
7 left = 0
8 right = 0
9 count = 0
10 sum = 0
11
12 while right < n:
13     sum += nums[right]
14
15

```

```

15     while sum >= x:
16         count += n - right
17         sum -= nums[left]
18         left += 1
19
20     right += 1
21
22 print(count)

```

## Java

```

1  import java.util.Scanner;
2
3  public class Main {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6          int n = scanner.nextInt();
7          int x = scanner.nextInt();
8
9          int[] nums = new int[n];
10         for (int i = 0; i < n; i++) nums[i] = scanner.nextInt();
11
12         int left = 0; // 滑动窗口的左端点
13         int right = 0; // 滑动窗口的右端点
14         int count = 0; // 记录连续区间个数
15         int sum = 0; // 记录当前区间的和
16
17         while (right < n) {
18             sum += nums[right];
19
20             while (sum >= x) {
21                 // 如果当前区间和大于等于x，那么以left为起点的所有连续区间都符合要求
22                 count += n - right;
23                 sum -= nums[left];
24                 left++;
25             }
26
27             right++;
28         }
29
30

```

```
30     System.out.println(count);
31 }
32 }
```

## C语言

```
1  #include <stdio.h>
2
3  int main() {
4      int n, x;
5      scanf("%d %d", &n, &x); // 读取N和x
6
7      int nums[n];
8      for (int i = 0; i < n; i++) {
9          scanf("%d", &nums[i]); // 读取数组中的正整数
10     }
11
12     int left = 0; // 滑动窗口的左端点
13     int right = 0; // 滑动窗口的右端点
14     int count = 0; // 记录连续区间个数
15     int sum = 0; // 记录当前区间的和
16
17     // 遍历数组, 使用滑动窗口计算连续区间的和
18     while (right < n) {
19         sum += nums[right]; // 将右端点的值加到区间和中
20
21         // 当区间和大于等于x时, 移动左端点
22         while (sum >= x) {
23             count += n - right; // 以left为起点的所有连续区间都符合要求
24             sum -= nums[left]; // 移动左端点, 区间和减去左端点的值
25             left++;
26         }
27
28         right++; // 移动右端点
29     }
30
31     printf("%d\n", count); // 输出符合条件的连续区间个数
32
33     return 0;
34 }
```

完整用例

用例1

1	3 7
2	3 4 7

用例2

1	5 10
2	1 2 3 4 5

用例3

1	4 5
2	1 2 3 4

用例4

1	6 15
2	1 2 3 4 5 6

用例5

1	8 20
2	2 4 6 8 10 12 14 16

用例6

1	10 100
2	10 20 30 40 50 60 70 80 90 100

用例7

1	7 50
2	10 20 30 40 50 60 70

用例8

1	15 80
2	5 10 15 20 25 30 35 40 45 50 55 60 65 70 75

用例9

1	12 60
2	12 24 36 48 60 72 84 96 108 120 132 144

用例10

1	9 70
2	7 14 21 28 35 42 49 56 63

文章目录

华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷	
题目描述	
输入描述	
输出描述	
用例1	
用例2	
C++	
javaScript	
python	
Java	
C语言	
完整用例	
用例1	
用例2	
用例3	
用例4	
用例5	
用例6	



用例7  
用例8  
用例9  
用例10

# 机考真题 华为OD



CSDN @算法大师