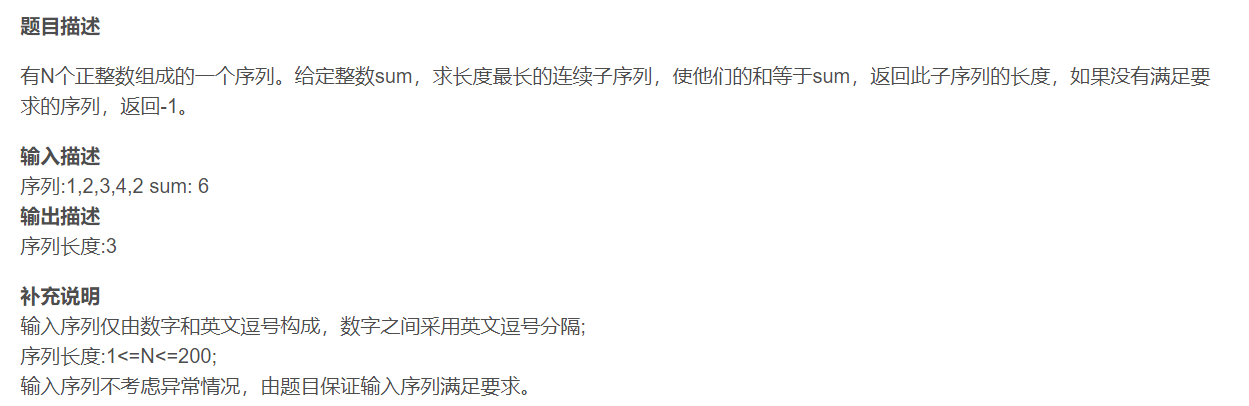
# **E卷-最长连续子序列[100分]（ Java | Python3 | C++ | C语言 | JsNode | Go）**







import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

String input = scanner.nextLine(); // 读取一行输入

String[] parts = input.split(","); // 以逗号分隔字符串

ArrayList<Integer> arr = new ArrayList<>();

// 将分割后的字符串转换为整数并存入ArrayList

for (String part : parts) {

arr.add(Integer.parseInt(part));

}

int sum = scanner.nextInt(); // 读取目标和

int res = -1; // 初始化结果为-1，表示尚未找到

// 遍历所有子序列

for (int i = 0; i < arr.size(); i++) {

int t = 0; // 子序列的和初始化为0

for (int j = i; j < arr.size(); j++) {

t += arr.get(j); // 累加当前子序列的元素值

if (t == sum) {

res = Math.max(res, j - i + 1); // 更新结果

}

}

}

System.out.println(res); // 输出结果

}

}



# 读取一行输入

input\_str = input()

# 以逗号分隔字符串并转换为整数列表

arr = list(map(int, input\_str.split(",")))

# 读取目标和

sum\_val = int(input())

res = -1 # 初始化结果为-1，表示尚未找到

# 遍历所有子序列

for i in range(len(arr)):

t = 0 # 子序列的和初始化为0

for j in range(i, len(arr)):

t += arr[j] # 累加当前子序列的元素值

if t == sum\_val:

res = max(res, j - i + 1) # 更新结果

print(res) # 输出结果



#include <iostream>

#include <vector>

#include <sstream>

using namespace std;

int main() {

string input;

// 读取一行输入，以获取逗号分隔的数字

getline(cin, input);

vector<int> arr;

stringstream ss(input); // 用字符串流处理输入的字符串

string temp;

// 以逗号为分隔符分割字符串并存入vector

while (getline(ss, temp, ',')) {

arr.push\_back(stoi(temp)); // 将字符串转换为整数后存入vector

}

int sum;

// 读取所需的目标和

cin >> sum;

int res = -1; // 初始化结果为-1，表示尚未找到最长连续子序列

// 遍历所有子序列

for (int i = 0; i < arr.size(); i++) {

int t = 0; // 子序列的和初始化为0

for (int j = i; j < arr.size(); j++) {

t += arr[j]; // 累加当前子序列的元素值

// 如果累加和等于目标和

if (t == sum) {

// 更新结果，取当前长度和先前最大长度的较大值

res = max(res, j - i + 1);

}

}

}

// 输出结果

cout << res;

return 0;

}



#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main() {

char input[1000];

fgets(input, 1000, stdin); // 读取一行输入

input[strcspn(input, "\n")] = 0; // 去除换行符

int arr[1000], arr\_size = 0;

char \*token = strtok(input, ",");

// 以逗号分隔字符串并转换为整数存入数组

while (token != NULL) {

arr[arr\_size++] = atoi(token);

token = strtok(NULL, ",");

}

int sum;

scanf("%d", &sum); // 读取目标和

int res = -1; // 初始化结果为-1，表示尚未找到

// 遍历所有子序列

for (int i = 0; i < arr\_size; i++) {

int t = 0; // 子序列的和初始化为0

for (int j = i; j < arr\_size; j++) {

t += arr[j]; // 累加当前子序列的元素值

if (t == sum) {

res = (res > (j - i + 1)) ? res : (j - i + 1); // 更新结果

}

}

}

printf("%d\n", res); // 输出结果

return 0;

}



const readline = require("readline");

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

rl.question("", (input) => {

const arr = input.split(",").map(Number); // 以逗号分隔字符串并转换为整数数组

rl.question("", (sum) => {

sum = parseInt(sum);

let res = -1; // 初始化结果为-1，表示尚未找到

// 遍历所有子序列

for (let i = 0; i < arr.length; i++) {

let t = 0; // 子序列的和初始化为0

for (let j = i; j < arr.length; j++) {

t += arr[j]; // 累加当前子序列的元素值

if (t === sum) {

res = Math.max(res, j - i + 1); // 更新结果

}

}

}

console.log(res); // 输出结果

rl.close();

});

});



package main

import (

"fmt"

"strconv"

"strings"

)

func main() {

var input string

fmt.Scanln(&input)

parts := strings.Split(input, ",") // 以逗号分隔字符串

var arr []int

for \_, part := range parts {

num, \_ := strconv.Atoi(part)

arr = append(arr, num) // 将分割后的字符串转换为整数并存入切片

}

var sum int

fmt.Scan(&sum) // 读取目标和

res := -1 // 初始化结果为-1，表示尚未找到

// 遍历所有子序列

for i := 0; i < len(arr); i++ {

t := 0 // 子序列的和初始化为0

for j := i; j < len(arr); j++ {

t += arr[j] // 累加当前子序列的元素值

if t == sum {

if res <= (j - i + 1) {

res = j - i + 1 // 更新结果

}

}

}

}

fmt.Println(res) // 输出结果

}