

【华为OD机考 统一考试机试C卷】 游戏分组/王者荣耀 (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

2020年题：

英雄联盟是一款十分火热的对战类游戏。每一场对战有10位玩家参与，分为两组，每组5人。每位玩家都有一个战斗力，代表着这位玩家的厉害程度。为了对战尽可能精彩，我们需要把玩家们分为实力尽量相等的两组。一组的实力可以表示为这一组5位玩家的战斗力和。现在，给你10位玩家的战斗力，请你把他们分为实力尽量相等的两组。请你输出这两组的实力差。

2023年题：

部门准备举办一场王者荣耀表演赛，有10名游戏爱好者参与，分5为两队，每队5人。每位参与者都有一个评分，代表着他的游戏水平。为了表演赛尽可能精彩，我们需要把10名参赛者分为实力尽量相近的两队。一队的实力可以表示为这一队5名队员的评分总和。

现在给你10名参与者的游戏水平评分，请你根据上述要求分队最后输出这两组的实力差绝对值。

例: 10名参赛者的评分分别为5 1 8 3 4 6 7 10 9 2，分组为 (135 8 10) (24 679)，两组实力差最小，差值为1。有多种分法，但实力差的绝对值最小为1。

输入描述

10个整数，表示10名参与者的游戏水平评分。范围在[1,10000]之间

输出描述

1个整数，表示分组后两组实力差绝对值的**最小值**。

用例1

输入：

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

输出：

```
1 | 1
```

说明：

10名队员分成两组，两组实力差绝对值最小为1。

解题思路

C++

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <cmath>
5  #include <climits>
6
7  using namespace std;
8
9  int res = INT_MAX;
10 int totalSum = 0;
11 int targetSum = 0;
12
13 // 深度优先搜索函数
14 void dfs(vector<int>& nums, int idx, int count, int currentSum) {
15     // 剪枝条件：如果当前总和超过目标，则停止，考友反馈，去掉可得100%
16 }
```

```

17 // if (currentSum > targetSum) return;
18
19 // 当我们为一个队伍选择了5名玩家时
20 if (count == 5) {
21     // 计算另一个队伍的总和
22     int otherTeamSum = totalSum - currentSum;
23     // 用较小的差值更新结果
24     res = min(res, abs(currentSum - otherTeamSum));
25     return;
26 }
27
28 // 如果我们已经考虑了所有玩家, 停止递归
29 if (idx == 10) return;
30
31 // 为第一个队伍选择当前玩家
32 dfs(nums, idx + 1, count + 1, currentSum + nums[idx]);
33
34 // 不为第一个队伍选择当前玩家
35 dfs(nums, idx + 1, count, currentSum);
36 }
37
38 int main() {
39     vector<int> nums(10);
40     for (int i = 0; i < 10; ++i) {
41         cin >> nums[i];
42         totalSum += nums[i];
43     }
44     targetSum = totalSum / 2;
45     dfs(nums, 0, 0, 0);
46     cout << res << endl;
47     return 0;
48 }

```

Java

```

1 import java.util.*;
2
3 public class Main {
4     static int res = Integer.MAX_VALUE;
5     static int totalSum = 0;
6
7     // 递归函数
8     // 参数: nums, 当前索引, 当前队伍的玩家数量, 当前队伍的总和
9     // 返回值: 当前队伍和另一个队伍之间的最小差值
10    public void dfs(int[] nums, int idx, int count, int currentSum) {
11        // 如果当前队伍的总和大于目标总和, 返回
12        if (currentSum > targetSum) return;
13
14        // 如果已经考虑了所有玩家, 停止递归
15        if (idx == nums.length) return;
16
17        // 为当前队伍选择当前玩家
18        dfs(nums, idx + 1, count + 1, currentSum + nums[idx]);
19
20        // 不为当前队伍选择当前玩家
21        dfs(nums, idx + 1, count, currentSum);
22    }
23
24    public static void main(String[] args) {
25        Scanner sc = new Scanner(System.in);
26        int n = sc.nextInt();
27        int[] nums = new int[n];
28        for (int i = 0; i < n; ++i) {
29            nums[i] = sc.nextInt();
30        }
31        totalSum = 0;
32        for (int num : nums) {
33            totalSum += num;
34        }
35        targetSum = totalSum / 2;
36        Main main = new Main();
37        main.dfs(nums, 0, 0, 0);
38        System.out.println(res);
39    }
40 }

```

```

6   static int targetSum = 0;
7
8   public static void main(String[] args) {
9       Scanner cin = new Scanner(System.in);
10      int[] nums = Arrays.stream(cin.nextLine().split(" "))
11          .mapToInt(Integer::parseInt).toArray();
12      for (int num : nums) {
13          totalSum += num;
14      }
15      targetSum = totalSum / 2;
16      dfs(nums, 0, 0, 0);
17      System.out.println(res);
18      cin.close();
19  }
20
21  static void dfs(int[] nums, int idx, int count, int currentSum) {
22      // 剪枝条件: 如果当前总和超过目标, 则停止. 考友反馈, 去掉可得100%
23      // if (currentSum > targetSum) return;
24
25      // 当我们为一个队伍选择了5名玩家时
26      if (count == 5) {
27          // 计算另一个队伍的总和
28          int otherTeamSum = totalSum - currentSum;
29          // 用较小的差值更新结果
30          res = Math.min(res, Math.abs(currentSum - otherTeamSum));
31          return;
32      }
33
34      // 如果我们已经考虑了所有玩家, 停止递归
35      if (idx == 10) return;
36
37      // 为第一个队伍选择当前玩家
38      dfs(nums, idx + 1, count + 1, currentSum + nums[idx]);
39
40      // 不为第一个队伍选择当前玩家
41      dfs(nums, idx + 1, count, currentSum);
42  }
43 }

```

```

1  const readline = require('readline');
2
3  const rl = readline.createInterface({
4    input: process.stdin,
5    output: process.stdout
6  });
7
8  let res = Number.MAX_SAFE_INTEGER;
9  let totalSum = 0;
10 let targetSum = 0;
11
12 // 深度优先搜索函数
13 function dfs(nums, idx, count, currentSum) {
14   // 剪枝条件: 如果当前总和超过目标, 则停止 考友反馈, 去掉可得100%
15   // if (currentSum > targetSum) return;
16
17   // 当我们为一个队伍选择了5名玩家时
18   if (count === 5) {
19     // 计算另一个队伍的总和
20     let otherTeamSum = totalSum - currentSum;
21     // 用较小的差值更新结果
22     res = Math.min(res, Math.abs(currentSum - otherTeamSum));
23     return;
24   }
25
26   // 如果我们已经考虑了所有玩家, 停止递归
27   if (idx === 10) return;
28
29   // 为第一个队伍选择当前玩家
30   dfs(nums, idx + 1, count + 1, currentSum + nums[idx]);
31
32   // 不为第一个队伍选择当前玩家
33   dfs(nums, idx + 1, count, currentSum);
34 }
35
36 rl.on('line', (input) => {
37   let nums = input.split(' ').map(Number);
38   for (let num of nums) {
39     totalSum += num;
40   }
41

```

```

41     targetSum = totalSum / 2;
42     dfs(nums, 0, 0, 0);
43     console.log(res);
44     rl.close();
45 });

```

Python

```

1  import sys
2
3  res = sys.maxsize
4  totalSum = 0
5  targetSum = 0
6
7  # 深度优先搜索函数
8  def dfs(nums, idx, count, currentSum):
9      global res, totalSum, targetSum
10     # 剪枝条件: 如果当前总和超过目标, 则停止. 考友反馈, 去掉可得100%
11     # if currentSum > targetSum:
12     #     return
13
14     # 当我们为一个队伍选择了5名玩家时
15     if count == 5:
16         # 计算另一个队伍的总和
17         otherTeamSum = totalSum - currentSum
18         # 用较小的差值更新结果
19         res = min(res, abs(currentSum - otherTeamSum))
20         return
21
22     # 如果我们已经考虑了所有玩家, 停止递归
23     if idx == 10:
24         return
25
26     # 为第一个队伍选择当前玩家
27     dfs(nums, idx + 1, count + 1, currentSum + nums[idx])
28
29     # 不为第一个队伍选择当前玩家
30     dfs(nums, idx + 1, count, currentSum)
31
32 nums = list(map(int, input().split()))
33

```

```

33 for num in nums:
34     totalSum += num
35 targetSum = totalSum // 2
36 dfs(nums, 0, 0, 0)
37 print(res)

```

C语言

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <limits.h>
4
5  int res = INT_MAX;
6  int totalSum = 0;
7  int targetSum = 0;
8
9  // 深度优先搜索函数
10 void dfs(int nums[10], int idx, int count, int currentSum) {
11     // 剪枝条件: 如果当前总和超过目标, 则停止. 考友反馈, 去掉可得100%
12     // if (currentSum > targetSum) return;
13
14     // 当我们为一个队伍选择了5名玩家时
15     if (count == 5) {
16         // 计算另一个队伍的总和
17         int otherTeamSum = totalSum - currentSum;
18         // 用较小的差值更新结果
19         res = abs(currentSum - otherTeamSum) < res ? abs(currentSum - otherTeamSum) : res;
20         return;
21     }
22
23     // 如果我们已经考虑了所有玩家, 停止递归
24     if (idx == 10) return;
25
26     // 为第一个队伍选择当前玩家
27     dfs(nums, idx + 1, count + 1, currentSum + nums[idx]);
28
29     // 不为第一个队伍选择当前玩家
30     dfs(nums, idx + 1, count, currentSum);
31 }
32
33

```

```
33 | int main() {
34 |     int nums[10];
35 |     for (int i = 0; i < 10; ++i) {
36 |         scanf("%d", &nums[i]);
37 |         totalSum += nums[i];
38 |     }
39 |     targetSum = totalSum / 2;
40 |     dfs(nums, 0, 0, 0);
41 |     printf("%d\n", res);
42 |     return 0;
43 | }
```

完整用例

用例1

```
1 | 1 1 1 1 1 1 1 1 1 1
```

用例2

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

用例3

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

用例4

```
1 | 1 1 1 1 1 1 10000 10000 10000 10000 10000
```

用例5

```
1 | 4500 4600 4700 4800 4900 5100 5200 5300 5400 5500
```

用例6

```
1 | 1000 1000 1000 1000 1000 9000 9000 9000 9000 9000
```


用例7

1 | 1234 5678 910 1112 1314 1516 789 2345 6789 34

用例8

1 | 1000 900 800 700 600 500 400 300 200 100

用例9

1 | 1000 3000 5000 7000 9000 2000 4000 6000 8000 10000

用例10

1 | 5000 5000 4000 4000 3000 3000 2000 2000 1000 1000

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
 - 解题思路
 - C++
 - Java
 - javaScript
 - Python
 - 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