题目描述

篮球(5V5)比赛中，每个球员拥有一个战斗力，每个队伍的所有球员战斗力之和为该队伍的总体战斗力。

现有10个球员准备分为两队进行训练赛，教练希望2个队伍的战斗力差值能够尽可能的小，以达到最佳训练效果。

给出10个球员的战斗力，如果你是教练，你该如何分队，才能达到最佳训练效果?请说出该分队方案下的**最小战斗力差值。**

输入描述

10个篮球队员的战斗力(整数，范围[1,10000]),战斗力之间用空格分隔，如:10987654321

不需要考虑异常输入的场景。

输出描述

最小的战斗力差值，如:1

用例

输入	10 9 8 7 6 5 4 3 2 1
输出	1
说明	1 2 5 9 10分为一队，3 4 6 7 8分为一队，两队战斗力之差最小，输出差值1。备注：球员分队方案不唯一，但最小战斗力差值固定是1

题目解析

本题由于只有十个数，并且平均分为两队，即每队五个数，因此只需要在10个数中选5个数组合即可模拟一队能力值。

有了一队的能力值之和A，我们用十个数的总和减去A，即得到了另一队的能力值之和B，最后求解Math.abs(A-B)，即为两队能力值之差。

因此，有多少个10选5组合，就有多少组能力值之差，我们取其中最小的即为题解。

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,
});

rl.on("line", (line) => {
  const arr = line.split(" ").map(Number);
  console.log(getResult(arr));
});

function getResult(arr) {
  arr.sort((a, b) => a - b);

  const res = [];
  // dfs求10选5的去重组合，并将组合之和记录进res中，则res中记录的是所有可能性的5人小队实力值之和
  dfs(arr, 0, 0, 0, res);

  const sum = arr.reduce((p, c) => p + c);

  // 某队实力为subSum，则另一队实力为sum - subSum，则两队实力差为 abs((sum - subSum) - subSum)，允求最小实力差
  return res
    .map((subSum) => Math.abs(sum - 2 * subSum))
    .sort((a, b) => a - b)[0];
}

// 求解去重组合
function dfs(arr, index, level, sum, res) {
  if (level === 5) {
    return res.push(sum);
  }

  for (let i = index; i < 10; i++) {
    if (i > 0 && arr[i] == arr[i - 1]) continue; // arr已经升序，这里进行树层去重
    dfs(arr, i + 1, level + 1, sum + arr[i], res);
  }
}
```

Java算法源码

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

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

    int[] arr = new int[10];
    for (int i = 0; i < 10; i++) {
      arr[i] = sc.nextInt();
    }

    System.out.println(getResult(arr));
  }

  public static int getResult(int[] arr) {
    Arrays.sort(arr);

    ArrayList<Integer> res = new ArrayList<>();
    // dfs求10选5的去重组合，共将组合之和记录进res中，即res中记录的是所有可能性的5人小队实力值之和
    dfs(arr, 0, 0, 0, res);

    int sum = Arrays.stream(arr).reduce(Integer::sum).orElse(0);
    // 其队实力方subSum，则另一队实力方sum - subSum，则两队实力差为 abs((sum - subSum) - subSum)，求求最小实力差
    return res.stream().map(subSum -> Math.abs(sum - 2 * subSum)).min((a, b) -> a - b).orElse(0);
  }

  // 求解去重组合
  public static void dfs(int[] arr, int index, int level, int sum, ArrayList<Integer> res) {
    if (level == 5) {
      res.add(sum);
      return;
    }

    for (int i = index; i < 10; i++) {
      if (i > 0 && arr[i] == arr[i - 1]) continue; // arr已经升序，这里进行树层去重
      dfs(arr, i + 1, level + 1, sum + arr[i], res);
    }
  }
}
```

Python算法源码

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


# 求解组合
def dfs(arr, index, level, sumV, res):
    if level == 5:
        res.append(sumV)
        return

    for i in range(index, 10):
        if i > 0 and arr[i] == arr[i - 1]: # arr已经升序，这里进行树层去重
            continue
        dfs(arr, i + 1, level + 1, sumV + arr[i], res)


# 算法入口
def getResult(arr):
    arr.sort()

    res = []
    # dfs求10选5的组合，并将组合之和记录进res中，即res中记录的是所有可能性的5人小队实力值之和
    dfs(arr, 0, 0, 0, res)

    sumV = sum(arr)
    #  求队实力为subSum，则另一队实力为sum - subSum，则两队实力差为 abs((sum - subSum) - subSum)，先求最小实力差
    return min(map(lambda subSum: abs(sumV - 2 * subSum), res))


# 算法调用
print(getResult(arr))
```