题目描述

用一个数组A代表程序员的工作能力，公司想通过结对编程的方式提高员工的能力，假设结对后的能力为两个员工的能力之和，求一共有多少种结对方式使结对后能力为N。

输入描述

```
5
1 2 2 2 3
4
```

第一行为员工的总人数，取值范围[1,1000]

第二行为数组A的元素，每个元素的取值范围[1,1000]

第三行为N的值，取值范围[1,1000]

输出描述

```
4
```

满足结对后能力为N的结对方式总数。

用例

输入	5 1 2 2 2 3 4
输出	4
说明	满足要求的结对方式为：A[0]和A[4]，A[1]和A[2]，A[1]和A[3]，A[2]和A[3]。

题目解析

本题应该只能用暴力方法求解所有两两组队。

但是员工的总人数，取值范围[1,1000]，这意味着O(n^2)时间复杂度可能会超时，因此我们需要做一些优化。

我们可以先将数组A升序，然后外层循环 i 范围0 ~ A.len-1，内层循环 j 范围 A.len-1 ~ i+1，如果遇到A[i]+A[j] === N，则计数++，如果遇到A[i]+A[j] < N，则说明内层后续遍历出来的元素都无法满足要求，则内层循环终止，继续下一次外层。

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,
});

const lines = [];
rl.on("line", (line) => {
  lines.push(line);

  if (lines.length === 3) {
    const arr = lines[1].split(" ").map(Number);
    const n = parseInt(lines[2]);

    console.log(getResult(arr, n));

    lines.length = 0;
  }
});

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

  let ans = 0;
  for (let i = 0; i < arr.length; i++) {
    for (let j = arr.length - 1; j >= i + 1; j--) {
      let sum = arr[i] + arr[j];
      if (sum === n) ans++;
      if (sum < n) break;
    }
  }

  return ans;
}
```

Java算法源码

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

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

    int total = sc.nextInt();

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

    int n = sc.nextInt();

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

  // 算法入口
  public static int getResult(int[] arr, int n) {
    Arrays.sort(arr);

    int ans = 0;
    for (int i = 0; i < arr.length; i++) {
      for (int j = arr.length - 1; j >= i + 1; j--) {
        int sum = arr[i] + arr[j];
        if (sum == n) ans++;
        else if (sum < n) break;
      }
    }

    return ans;
  }
}
```

Python算法源码

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


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

    ans = 0
    for i in range(total):
        for j in range(total - 1, i, -1):
            sumV = arr[i] + arr[j]

            if sumV == n:
                ans += 1
            elif sumV < n:
                break

    return ans


# 算法调用
print(getResult())
```