题目描述

给定一个正整数数组，检查数组中是否存在满足规则的数字组合

规则：**A = B + 2C**

输入描述

第一行输出数组的元素个数。

接下来一行输出所有数组元素，用空格隔开。

输出描述

如果存在满足要求的数，在同一行里依次输出规则里A/B/C的取值，用空格隔开。

如果不存在，输出0。

备注

1. 数组长度在3-100之间。
2. 数组成员为0-65535，数组成员可以重复，但每个成员只能在结果算式中使用一次。如：数组成员为[0, 0, 1, 5]，0出现2次是允许的，但结果0 = 0 + 2 * 0是不允许的，因为算式中使用了3个0。
3. 用例保证每组数字里最多只有一组符合要求的解。

用例

输入	4 2 7 3 0
输出	7 3 2
说明	7 = 3 + 2 * 2

输入	3 1 1 1
输出	0
说明	找不到满足条件的组合

题目解析

本题数组长度不大，只有3~100，因此三重for暴力破解，也不会超时。

我们可以先将数组降序，则第一层for选择的数必然是最大的，可以作为A，第二层和第三层是较小数，可以作为B和C，或者C和B。

备注中还说明了

> 用例保证每组数字里最多只有一组符合要求的解。

因此，我们一旦找到符合要求的解，就可以直接return

Java算法源码

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

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

    int n = Integer.parseInt(sc.nextLine());
    Integer[] arr =
        Arrays.stream(sc.nextLine().split(" ")).map(Integer::parseInt).toArray(Integer[]::new);

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

  public static String getResult(int n, Integer[] arr) {
    Arrays.sort(arr, (a, b) -> b - a);

    for (int i = 0; i < n; i++) {
      for (int j = i + 1; j < n; j++) {
        for (int k = j + 1; k < n; k++) {
          if (arr[i] == arr[j] + 2 * arr[k]) {
            return arr[i] + " " + arr[j] + " " + arr[k];
          }
          if (arr[i] == arr[k] + 2 * arr[j]) {
            return arr[i] + " " + arr[k] + " " + arr[k];
          }
        }
      }
    }

    return "0";
  }
}
```

JS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 === 2) {
    let n = parseInt(lines[0]);
    let arr = lines[1].split(" ").map(Number);

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

    lines.length = 0;
  }
});

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

  for (let i = 0; i < n; i++) {
    for (let j = i + 1; j < n; j++) {
      for (let k = j + 1; k < n; k++) {
        if (arr[i] == arr[j] + 2 * arr[k]) {
          return `${arr[i]} ${arr[j]} ${arr[k]}`;
        }

        if (arr[i] == arr[k] + 2 * arr[j]) {
          return `${arr[i]} ${arr[k]} ${arr[j]}`;
        }
      }
    }
  }

  return "0";
}
```

Python算法源码

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


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

    for i in range(n):
        for j in range(i + 1, n):
            for k in range(j + 1, n):
                if arr[i] == arr[j] + 2 * arr[k]:
                    return f"{arr[i]} {arr[j]} {arr[k]}"

                if arr[i] == arr[k] + 2 * arr[j]:
                    return f"{arr[i]} {arr[k]} {arr[j]}"

    return "0"


# 算法调用
```