题目描述

众数是指一组数据中出现次数量多的那个数，众数可以是多个。

中位数是指把一组数据从小到大排列，最中间的那个数，如果这组数据的个数是奇数，那最中间那个就是中位数，如果这组数据的个数为偶数，那就把中间的两个数之和除以2，所得的结果就是中位数。

查找整型数组中元素的众数并组成一个新的数组，求新数组的中位数。

输入描述

输入一个一维整型数组，数组大小取值范围 0<N<1000，数组中每个元素取值范围 0<E<1000

输出描述

输出众数组成的新数组的中位数

用例

输入	10 11 21 19 21 17 21 16 21 18 15
输出	21

输入	2 1 5 4 3 3 9 2 7 4 6 2 15 4 2 4
输出	3

输入	5 1 5 3 5 2 5 5 7 6 7 3 7 11 7 55 7 9 98 9 17 9 15 9 9 1 39
输出	7

题目解析

逻辑题。具体逻辑请看代码注释。

Java算法源码

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

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

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

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

  public static int getResult(Integer[] nums) {
    HashMap<Integer, Integer> count = new HashMap<>();

    // 统计各数字出现次数
    for (int num : nums) {
      count.put(num, count.getOrDefault(num, 0) + 1);
    }

    // 获取最大出现次数
    int max = count.values().stream().max((a, b) -> a - b).orElse(0);

    // 将众数挑选出来
    ArrayList<Integer> ans = new ArrayList<>();
    for (Integer k : count.keySet()) {
      if (count.get(k) == max) ans.add(k);
    }

    // 众数升序
    ans.sort((a, b) -> a - b);

    // 中位数取值
    int mid = ans.size() / 2;
    if (ans.size() % 2 == 0) {
      // 偶数个数时，取中间两个位置的平均值
      return (ans.get(mid) + ans.get(mid - 1)) / 2;
    } else {
      // 奇数个数时，取中间位置的值
      return ans.get(mid);
    }
  }
}
```

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,
});

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

function getResult(nums) {
  const count = {};

  // 统计各数字出现次数
  for (let num of nums) {
    count[num] = (count[num] ?? 0) + 1;
  }

  // 获取最大出现次数
  const max = Math.max(...Object.values(count));

  // 将众数挑选出来
  const mode = [];
  for (let k in count) {
    if (count[k] == max) mode.push(parseInt(k));
  }

  // 众数升序
  mode.sort((a, b) => a - b);

  // 中位数取值
  const mid = Math.floor(mode.length / 2);
  if (mode.length % 2 == 0) {
    // 偶数个数时，取中间两个位置的平均值
    return Math.floor((mode[mid] + mode[mid - 1]) / 2);
  } else {
    // 奇数个数时，取中间位置的值
    return mode[mid];
  }
}
```

Python算法源码

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


# 算法入口
def getResult():
    count = {}

    # 统计各数字出现次数
    for num in nums:
        count[num] = count.get(num, 0) + 1

    # 获取最大出现次数
    maxCount = max(count.values())

    # 将众数挑选出来
    mode = []
    for k in count:
        if count[k] == maxCount:
            mode.append(int(k))

    # 众数升序
    mode.sort()

    # 中位数取值
    mid = len(mode) // 2
    if len(mode) % 2 == 0:
        # 偶数个数时，取中间两个位置的平均值
        return (mode[mid] + mode[mid - 1]) // 2
    else:
        # 奇数个数时，取中间位置的值
        return mode[mid]


# 算法调用
```