链滴

2023 PDD 笔试

作者：Hildaquan

原文链接：https://ld246.com/article/1678626775975

来源网站：链滴

NO.1

A 了

```
/**
 * Description:
 * date: 2023/03/12 下午 7:14
 *
 * @author Quan
 */

import java.util.*;
import java.io.InputStreamReader;
import java.io.BufferedReader;
import java.io.IOException;

public class Main {
    static BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));

    public static void main(String[] args) throws IOException {
        String s = bf.readLine();
        char[] str = s.toCharArray();
        int n = str.length;
        StringBuilder sb = new StringBuilder();

        int numEnd = 0;
        for (int i = 0, j = 0; i <= j && j < n; ) {
            if (isNum(str[j])) {
                j++;
            } else {
                numEnd = j;
                while (j < n && !isNum(str[j])) j++;

                int num = Integer.parseInt(s.substring(i, numEnd));
                String c = s.substring(numEnd, j);
                System.out.println(num + " " + c);

                while (num-- > 0) sb.append(c);

                // 更新下表
                i = j;
            }
        }

        System.out.println(sb);
    }

    private static boolean isNum(char c) {
        return (c >= '0' && c <= '9');
    }
}
```

原文链接：2023 PDD 笔试

No.2

A 了

```
/**
 * Description:
 * date: 2023/03/12 下午 7:20
 *
 * @author Quan
 */
import java.util.*;
import java.io.InputStreamReader;
import java.io.BufferedReader;
import java.io.IOException;

// 读取单个数字：Scanner
// 读取一串字符：BufferedReader
public class Main {
    static BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
    static Scanner in = new Scanner(System.in);

    public static void main(String[] args) throws IOException {
        int T = in.nextInt(), N = in.nextInt();
        int[][] enermy = new int[T][N];
        for (int i = 0; i < T; i++) {
            for (int j = 0; j < N; j++) {
                enermy[i][j] = in.nextInt();
            }
        }


        // 贪心算法：让所有小于1的先归零，最后统计全部剩余的元素
        int[] ans = new int[T];
        int i = 0;
        for (int[] e : enermy) {
            // 1. 先升序排序
            Arrays.sort(e);

            // 2. 计算
            ans[i++] = attack(e);
        }

        for (int a : ans) System.out.println(a);
    }

    private static int attack(int[] e) {
        int a = 0;
        for (int i = 0; i < e.length; i++) {
```

```
            if (e[i] == 1) {
                a++;
                e[i] = 0;
            }
        }

        if (a % 2 != 0) {
            e[0] = 1;
            a--;
        }
        a /= 2;
        int b = 0;
        for (int n : e) {
            if (n != 0) b++;
        }

        return a + b;
    }
}
```

No.3

只过了 35%。。。。

原本思路打算用贪心算法做的。。。

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.*;

/**
 * Description:
 * date: 2023/03/12 下午 7:46
 *
 * @author Quan
 */
public class Main {
    static BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
    static Scanner in = new Scanner(System.in);

    public static void main(String[] args) throws IOException {
        int N = in.nextInt();
        String[] joins = new String[N];
        for (int i = 0; i < N; i++) {
            joins[i] = in.next();
        }

        Map<Character, int[]> map = new HashMap<>();
```

```
        for (int i = 0; i < 3; i++) {
            int[] nums = new int[2];
            nums[0] = in.nextInt();
            nums[1] = in.nextInt();
            map.put((char)('A' + i), nums);
        }
        // 最小花费在先
        List<Map.Entry<Character, int[]>> list = new ArrayList<>(map.entrySet());
        list.sort((a, b) -> {
            int[] na = a.getValue();
            int[] nb = b.getValue();
            return na[1] - nb[1];
        });


        // 保证先消费最省钱的，再保证消费只有一个选择的
        int minSum = 0;
        for (int k = 0; k < list.size(); k++) {
            Map.Entry<Character, int[]> l = list.get(k);

            Character activity = l.getKey();
            int nums = l.getValue()[0];
            int ticket = l.getValue()[1];
            //System.out.println(activity + " " + nums + " " + ticket);
            if (nums == 0) continue;

            for (int i = 0; i < joins.length; i++) {
                String s = joins[i];
                if (s.equals("-")) continue;

                for (int j = 0; j < s.length(); j++) {
                    char c = s.charAt(j);
                    if (c == activity && nums > 0) {
                        minSum += ticket;
                        // 标记已完成
                        joins[i] = "-";
                        // 更新票数
                        l.setValue(new int[]{nums--, ticket});
                        break;
                    }
                }
            }
        }

//        for (Map.Entry<Character, int[]> l : list) {
//            System.out.println(l.getKey() + " " + l.getValue()[0] + " " + l.getValue()[1]);
//        }

        boolean flag = false;
        int count = 0;
        for (String s : joins) {
            if (!s.equals("-")) {
                flag = true;
                break;
```

```
            }
            count++;
        }

        if (!flag) {
            System.out.println("YES");
            System.out.println(minSum);
        } else {
            System.out.println("NO");
            System.out.println(count);
        }
    }
}
```

No.4

前缀和求平均数

双堆求数据流中位数

● LeetCode 原题：295.数据流的中位数

```
package PDD.n04;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.PriorityQueue;
import java.util.Scanner;

/**
 * Description:
 * date: 2023/03/12 下午 8:33
 *
 * @author Quan
 */
public class Main {
    static BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
    static Scanner in = new Scanner(System.in);

    public static void main(String[] args) throws IOException {
        int N = in.nextInt();
        int[] cus = new int[N];
        for (int i = 0; i < N; i++) cus[i] = in.nextInt();

        // 前缀和
        int[] prefix = new int[N];
        prefix[0] = cus[0];
        for (int i = 1; i < N; i++) {
            prefix[i] = prefix[i - 1] + cus[i];
        }
```

```
        // 1. 求平均数
        for (int i = 0; i < N; i++) {
            if (i == 0) {
                System.out.print(prefix[i] + " ");
                continue;
            }
            double ans = Math.ceil(prefix[i] / ((i + 1) * 1.0));
            System.out.print((int)ans + " ");
        }
        System.out.println();

        // 2. 求中位数
        MyQue que = new MyQue();
        for (int n : cus) {
            que.addNum(n);
            int ans = que.getMid();
            System.out.print(ans + " ");
        }
    }
}

class MyQue {
    PriorityQueue<Integer> left = new PriorityQueue<>((a, b) -> b - a);     // 大根堆
    PriorityQueue<Integer> right = new PriorityQueue<>((a, b) -> a - b);    // 小根堆

    // 确保left - right == 1
    public void addNum(int num) {
        int leftLen = left.size(), rightLen = right.size();
        if (leftLen == rightLen) {
            // 当right为空，或者num比右边最小值小 ---> 添加到左边
            if (right.isEmpty() || num < right.peek()) {
                left.add(num);
            } else {
                // 当num大于等于右边最小值 ---> 先将右边的最小值移到左边，再将num加到右边
                left.add(right.poll());
                right.add(num);
            }
        }
        // 此处保证是 leftLen > rigthLen
        else {
            if (num >= left.peek()) {
                right.add(num);
            } else {
                // 此处rigthLen有可能为0，所以不需要判断!
                right.add(left.poll());
                left.add(num);
            }
        }
    }

    int getMid() {
        if ((left.size() + right.size()) % 2 == 0) {
            double ans = Math.ceil((left.peek() + right.peek()) / 2.0);
```

```
            //System.out.println("1:" + left.peek() + " " + right.peek());
            return (int)ans;
        }
        return left.peek();
    }
}
```