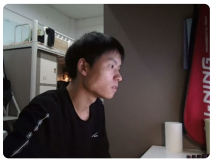
比特就业课Java方向笔试强训48天day22_11月10日（105Java）-王世国-测评结果

考生信息 存在作弊行为



王世国

投递编号：91 | 学校：武汉轻工大学 | 邮箱：1477649017@qq.com | 职位：比特就业课105期Java2班 |

参考区域: 湖北武汉（113.57.53.192） |

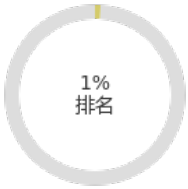
做题用时：02:20:57(2022-11-09 21:58:00开始答题，2022-11-10 13:08:48交卷) | 作答设备： PC |

已同意诚信声明和隐私协议

100.0 分 / 100分

在本次考试中，考生总成绩为100.0分/100分，评级为A（排名前1%），编程能力优秀（2题完全通过，分数排名前1%），编程思路完全正确，编程规范性高。该考生在本次考试中存在作弊行为，和较早通过的代码相似度极高（95.7%），无视频监控截图，未开启摄像头。

考生成绩



题型	得分	正确题数	排名	用时	是否阅卷
单选	50.0	10	1	00:19:01	已阅
编程	50.0	2	1	01:26:52	已阅

作弊风险

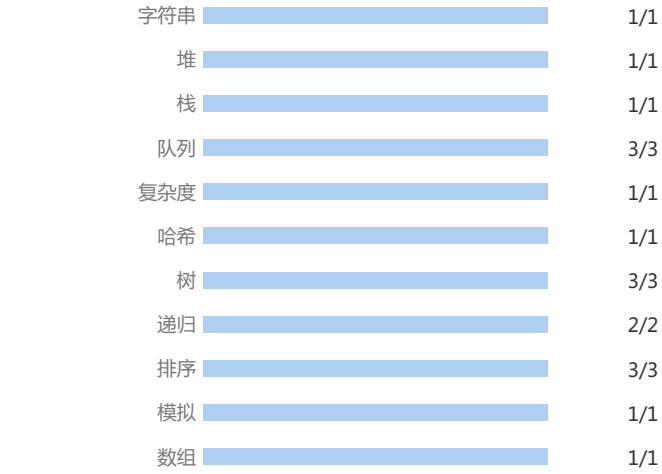
高风险 未开启摄像头

考生考试过程中未开启摄像头

高风险 代码相似度

编程题1 95.65%

知识点技能图谱



知识点	得分	正确题数
字符串	25.0	1
堆	5.0	1
栈	5.0	1
队列	15.0	3
复杂度	5.0	1
哈希	5.0	1

知识点	得分	正确题数
树	15.0	3
递归	30.0	2
排序	15.0	3
模拟	25.0	1
数组	5.0	1
链表	15.0	3
操作系统	5.0	1
数学	25.0	1

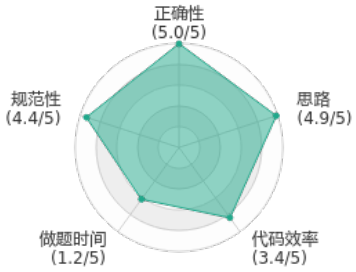
历史笔试记录

序号	试卷名称	排名	总得分	得分详情	作弊嫌疑	安排笔试时间	交卷时间
1	比特就业课105期+2022寒假班C1考试	20.0%	48.8/60	单选:30.0分 编程:18.75分	否	2022-03-29 11:16:18	2022-03-31 18:51:27
2	比特就业课105期+2022寒假班C2考试	66.0%	24.0/60	单选:24.0分 编程:0.0分	否	2022-04-11 14:12:23	2022-04-11 20:12:16
3	比特就业课 105期JavaSE考试	11.0%	56.0/60	单选:26.0分 编程:30.0分	否	2022-07-12 16:00:16	2022-07-13 15:48:42
4	比特就业课 105期java方向 数据结构考试	47.0%	50.0/60	单选:20.0分 编程:30.0分	否	2022-07-23 12:49:22	2022-07-25 09:56:28
5	比特就业课105期Java方向笔试强训48天day01_10月10日	36.0%	80.0/100	单选:40.0分 编程:40.0分	是，摄像头监控异常	2022-10-09 17:29:16	2022-10-09 21:57:25
6	比特就业课105期Java方向笔试强训48天day02_10月11日	13.0%	90.0/100	单选:40.0分 编程:50.0分	是，代码抄袭	2022-10-10 10:43:48	2022-10-10 21:13:15
7	比特就业课105期Java方向笔试强训48天day03_10月12日	1.0%	95.0/100	单选:45.0分 编程:50.0分	是，摄像头监控异常	2022-10-11 10:40:53	2022-10-12 10:03:09
8	比特就业课105期Java方向笔试强训48天day04_10月13日	2.0%	95.0/100	单选:30.0分 不定项选择:15.0分 编程:50.0分	是，摄像头监控异常	2022-10-12 10:31:10	2022-10-12 21:14:05
9	比特就业课105期Java方向笔试强训48天day05_10月14日	12.0%	85.0/100	单选:35.0分 不定项选择:0.0分 编程:50.0分	是，摄像头监控异常|代码抄袭	2022-10-13 11:41:43	2022-10-14 11:33:24

序号	试卷名称	排名	总得分	得分详情	作弊嫌疑	安排笔试时间	交卷时间
10	比特就业课105期Java方向笔试强训48天day06_10月15日	13.0%	86.7/100	单选:25.0分 不定项选择:11.67分 编程:50.0分	是，摄像头监控异常	2022-10-14 10:59:38	2022-10-14 21:18:11
11	比特就业课105期Java方向笔试强训48天day07_10月17日	1.0%	100.0/100	单选:50.0分 编程:50.0分	是，摄像头监控异常	2022-10-16 16:46:53	2022-10-16 20:38:39
12	比特就业课105期Java方向笔试强训48天day08_10月18日	33.0%	87.5/100	单选:40.0分 编程:47.5分	是，摄像头监控异常	2022-10-17 16:18:42	2022-10-17 20:16:45
13	比特就业课105期Java方向笔试强训48天day09_10月19日	3.0%	95.0/100	单选:45.0分 编程:50.0分	是，摄像头监控异常	2022-10-18 17:07:17	2022-10-19 11:02:55
14	比特就业课105期Java方向笔试强训48天day10_10月20日	58.0%	65.0/100	单选:40.0分 编程:25.0分	是，摄像头监控异常	2022-10-19 15:29:54	2022-10-19 21:44:34
15	比特就业课105期Java方向笔试强训48天day11_10月21日	2.0%	95.0/100	单选:45.0分 编程:50.0分	是，摄像头监控异常	2022-10-20 16:46:09	2022-10-20 23:36:15
16	比特就业课105期Java方向笔试强训48天day12_10月22日	15.000001%	90.0/100	单选:40.0分 编程:50.0分	是，摄像头监控异常	2022-10-21 16:44:18	2022-10-21 22:49:24
17	比特就业课105期Java方向笔试强训48天day13_10月24日	31.0%	65.0/100	单选:40.0分 编程:25.0分	是，摄像头监控异常|代码抄袭	2022-10-22 16:07:25	2022-10-23 23:15:22
18	比特就业课105期Java方向笔试强训48天day14_10月25日	58.999996%	55.0/100	单选:30.0分 编程:25.0分	是，摄像头监控异常	2022-10-24 14:08:32	2022-10-24 23:42:01
19	比特就业课105期Java方向笔试强训48天day15_10月26日	42.0%	55.0/100	单选:30.0分 编程:25.0分	是，摄像头监控异常	2022-10-24 14:11:25	2022-10-25 22:58:05
20	比特就业课105期Java方向笔试强训48天day16_10月27日	13.0%	89.4/100	单选:45.0分 编程:44.44分	是，摄像头监控异常	2022-10-24 14:25:44	2022-10-26 21:33:12
21	比特就业课105期Java方向笔试强训48天day17_10月28日	24.0%	83.3/100	单选:40.0分 编程:43.33分	是，摄像头监控异常	2022-10-24 14:29:13	2022-10-28 22:18:13
22	比特就业课105期Java方向笔试强训48天day18_10月29日	12.0%	84.1/100	单选:40.0分 编程:44.12分	是，摄像头监控异常	2022-10-24 14:31:43	2022-10-29 15:39:41
23	比特就业课Java方向Java班笔试强训48天day19_11月7日（105Java班）	47.0%	72.5/95	单选:40.0分 编程:32.5分	是，摄像头监控异常	2022-11-06 14:19:57	2022-11-07 11:29:43
24	比特就业课Java方向笔试强训48天day20_11月8日（105Java班）	1.0%	100.0/100	单选:50.0分 编程:50.0分	是，摄像头监控异常|代码抄袭	2022-11-07 09:57:17	2022-11-07 20:19:33

序号	试卷名称	排名	总得分	得分详情	作弊嫌疑	安排笔试时间	交卷时间
25	比特就业课Java方向笔试强训48天day21_11月9日（105Java班）	24.0%	83.8/100	单选:40.0分 编程:43.75分	是，摄像头监控异常	2022-11-07 10:00:37	2022-11-09 11:17:55

编码能力



题号	正确性	提交次数	做题用时	使用语言	运行时间	占用内存	编程思路	代码规范	成绩排名
编程题1	100%	3	01:05:49	Java	272ms	21436K	良	良	1%
编程题2	100%	6	00:21:03	Java	41ms	10848K	良	良	1%

1 [单选题 | 平均分2.69分 | 43人正确/80人做题 | 用时：<1分 得分： 5.0 / 5.0

若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算，则利用（ ）存储方式最节省时间。

A 顺序表
B 双链表
C 带头结点的双循环链表
D 单循环链表

他的回答： A (正确)

正确答案： A

2 [单选题 | 平均分2.05分 | 32人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

下列数据结构具有记忆功能的是？

A 队列
B 循环队列
C 栈 栈，比如函数之间的调用关系是保存在栈上的
D 顺序表

他的回答： C (正确)

正确答案： C

3 [单选题 | 平均分2.44分 | 38人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

循环队列放在一维数组A[0...M-1]中，end1指向队头元素，end2指向队尾元素的后一个位置。假设队列两端均可进行入队和出队操作，队列中最多能容纳M-1个元素。初始时为空，下列判断队空和队满的条件中，正确的是（ ）

A 队空：end1==end2；队满：end1==(end2+1) mod M
B 队空：end1==end2；队满：end2==(end1+1) mod (M-1)
C 队空：end2==(end1+1) mod M；队满：end1==(end2+1) mod M
D 队空：end1==(end2+1) mod M；队满：end2==(end1+1) mod (M-1)

他的回答： A (正确)

正确答案： A

参考答案：

end1指向队头元素，可知出队操作是先从A[end1]读数，然后end1再加1。

end2指向队尾元素的后一个位置，可知入队操作是先存数到A[end2]，然后end2再加1。

若用A[0]存储第一个元素，队列初始时，入队操作是先把数据放到A[0]中，然后end2自增，即可知end2初值为0；

而end1指向的是队头元素，队头元素在数组A中的下标为0，所以得知end1的初值也为0，可知队空条件为end1==end2；

然后考虑队列满时，因为队列最多能容纳M-1个元素，假设队列存储在下标为0到M-2的M-1个区域，队头为A[0]，队尾为A[M-2]，此时队列满，考虑在这种情况下end1和end2的状态，end1指向队头元素，可知end1=0，end2指向队尾元素的后一个位置，可知end2=M-2+1=M-1，所以可知队满的条件为end1==（end2+1）mod M，选A。

4 [单选题 | 平均分2.12分 | 33人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

对递归程序的优化的一般的手段为（ ）

A 尾递归优化

B 循环优化

C 堆栈优化

D 停止值优化

比如快速排序递归到最后趋于有序时候会直接选择使用插入排序

他的回答： A (正确)

正确答案： A

5 [单选题 | 平均分4.49分 | 70人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

将一颗有 100 个结点的完全二叉树从根这一层开始，每一层从左到右依次对结点进行编号，根节点编号为 1 ，则编号为 98 的节点的父节点编号为（ ）

A 47

B 48

C 49

D 50

他的回答： C (正确)

正确答案： C

6 [单选题 | 平均分3.78分 | 59人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

将一棵二叉树的根结点放入队列，然后递归的执行如下操作，将出队结点所有子结点加入队。以上操作可以实现哪种遍历（ ）

A 前序遍历

B 中序遍历

C 后序遍历

D 层序遍历

他的回答： D (正确)

正确答案： D

7 [单选题 | 平均分3.65分 | 57人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

有 1000 个无序的整数，希望使用最快的方式找出前 50 个最大的，最佳的选择是（ ）

A 冒泡排序

B 基数排序

C 堆排序

D 快速排序

他的回答： C (正确)

正确答案： C

8 [单选题 | 平均分2.95分 | 46人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

下列说法中错误的是（ ）

A 红黑树插入操作的平均时间复杂度为O(logn)，最坏时间复杂度为O(logn)

B B+树插入操作的平均时间复杂度为O(logn)，最坏时间复杂度为O(logn)

C Hash表插入操作的平均时间复杂度为O(logn)，最坏时间复杂度为O(n)

D 排序链表插入操作的平均时间复杂度为O(n)，最坏时间复杂度为O(n)

他的回答： C (正确)

正确答案： C

参考答案：
各种数据结构的search、insert和delete操作在平均情况下的时间复杂度比较

数据结构	search	insert	delete
数组	O(n)，有序数组折半查找是O(lgn)	O(n)	O(n)
双向链表	O(n)	O(1)	O(1)
排序二叉树	O(lgn)	O(lgn)	O(lgn)
哈希表（n与槽数m成正比）	O(1)	O(1)	O(1)

9 [单选题 | 平均分3.46分 | 54人正确/78人做题 | 用时：<1分 得分： 5.0 / 5.0

将两个各有n个元素的有序表归并成一个有序表,最少的比较次数是（ ）

A 2n
B 2n-1
C n-1
D n

他的回答： D (正确)

正确答案： D

10 [单选题 | 平均分2.69分 | 42人正确/78人做题 | 用时：10分 得分： 5.0 / 5.0

下列排序法中，每经过一次元素的交换会产生新的逆序的是（ ）

A 快速排序
B 冒泡排序
C 简单插入排序
D 简单选择排序

对于快速排序而言，例如Hoare版，以第一个元素为基准，后面选取到大于基准值的往前放，前面选取到大于基准值的往后放，但是相对于这个基准值而言，在这一轮比较没有比较完之前，没交换一次，对于基准值来说就会有一个逆序，因为后面持续存在比它小的元素

他的回答： A (正确)

正确答案： A

参考答案：
在数据元素的序列中，对于某个元素，如果其后存在一个元素小于它，则称之为存在一个逆序。冒泡排序只交换相邻元素，但不是每次移动都产生新的逆序。简单插入排序每一次比较后最多移掉一个逆序。快速排序每一次交换移动都会产生新的逆序，因为当不会有新的逆序产生时，本轮比较结束。简单选择排序的基本思想是先从所有 n 个待排序的数据元素中选择最小的元素，将该元素与第一个元素交换，再从剩下的 n-1 个元素中选出最小的元素与第 2 个元素交换，这样做不会产生逆序。故本题答案为 A 选项。

11 ACM编程题 语言限制 [编程题 | 平均分21.92分 | 64人正确/73人做题 | 提交: 2 次 得分： 25.0 / 25.0

标题：小易的升级之路 | 时间限制：1秒 | 内存限制：32768K | 语言限制： 不限

【小易的升级之路】小易经常沉迷于网络游戏.有一次,他在玩一个打怪升级的游戏,他的角色的初始能力值为 a.在接下来的一段时间内,他将会依次遇见n个怪物,每个怪物的防御力为b1,b2,b3...bn. 如果遇到的怪物防御力bi小于等于小易的当前能力值c,那么他就能轻松打败怪物,并 且使得自己的能力值增加bi;如果bi大于c,那他也能打败怪物,但他的能力值只能增加bi 与c的最大公约数.那么问题来了,在一系列的锻炼后,小易的最终能力值为多少?

输入描述：

> 对于每组数据,第一行是两个整数n(1≤n<100000)表示怪物的数量和a表示小易的初始能力值.
> 然后输入n行，每行整数,b1,b2...bn(1≤bi≤n)表示每个怪物的防御力

输出描述：

对于每组数据,输出一行.每行仅包含一个整数,表示小易的最终能力值

示例1：

输入

```
3 50
50 105 200
5 20
30 20 15 40 100
```

输出

```
110
205
```

代码片段

功能实现	TA的	平均
总通过率	100%	87%
基本测试用例通过率	6/6 (100%)	87%
边缘测试用例通过率	4/4 (100%)	87%

代码提交统计	TA的	平均
使用语言	Java	
做题用时	01:05:49	00:20:20
提交次数	3	1

代码执行统计

答案错误 ：2

答案正确 ：1

代码效率	TA的	参考
运行时间	272ms	1s
占用内存	21436K	32768K

代码规范及可读性

代码规范得分 4.0

Line 2: 'CLASS_DEF' should be separated from previous statement. [EmptyLineSeparator]

Line 3:32: Parameter name 'a' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

Line 3:38: Parameter name 'b' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

Line 12: 'METHOD_DEF' should be separated from previous statement. [EmptyLineSeparator]

Line 15:17: Local variable name 'n' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [LocalVariableName]

他的代码：

做题用时: 65 分钟　　语言：Java　　运行时间：272ms　　占用内存：21436K　　程序状态：答案正确

```
import java.util.*;
public class Main {
    public static int func(int a,int b){
        //求最大公约数
        while(b != 0){
            int mod = a%b;
            a = b;
            b = mod;
        }
        return a;
    }
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        while(scan.hasNext()){
            int n = scan.nextInt();
```

```
            int iniVal = scan.nextInt();//初始能力值
            int[] arr = new int[n];
            for(int i = 0;i < n;i++){
                arr[i] = scan.nextInt();
            }
            for(int i = 0;i < n;i++){
                if(arr[i] <= iniVal){
                    iniVal += arr[i];
                }else{
                    iniVal += func(arr[i],iniVal);
                }
            }
            System.out.println(iniVal);
        }
    }
}
```

点此或手机扫描二维码查看代码编写过程

参考答案：

```
#include <cstdio>
#include <cstring>
#include <cstdlib>
#include <algorithm>
using namespace std;
int gcd(int a,int b){
    int tmp;
    while(b){
        tmp = b; b = a % b ; a = tmp;
    }
    return a;
}
int main(){
    int n,a;
    while(scanf("%d%d",&n,&a) != EOF){
        for(int i = 0,x;i < n;++ i){ scanf("%d",&x); if(x <= a) a += x; else a += gcd(x,a); } printf("%d\n",a); } return 0; }
```

12 ACM编程题 语言限制 [编程题 | 平均分22.8分 | 62人正确/74人做题 | 提交: 6 次 得分： 25.0 / 25.0

标题：找出字符串中第一个只出现一次的字符 | 时间限制：1秒 | 内存限制：32768K | 语言限制： 不限

【找出字符串中第一个只出现一次的字符】

找出字符串中第一个只出现一次的字符

数据范围：输入的字符串长度满足 $1 \leq n \leq 1000$

输入描述：

> 输入一个非空字符串

输出描述：

> 输出第一个只出现一次的字符，如果不存在输出-1

示例1：

输入

> asdfasdfo

输出

> o

代码片段

功能实现		
	TA的	平均
总通过率	100%	91%
基本测试用例通过率	12/12 (100%)	91%
边缘测试用例通过率	8/8 (100%)	90%

代码提交统计		
	TA的	平均
使用语言	Java	
做题用时	00:21:03	00:19:45
提交次数	6	3

代码执行统计
答案错误 ：3
答案正确 ：3

代码效率		
	TA的	参考
运行时间	41ms	1s
占用内存	10848K	32768K

代码规范及可读性	
代码规范得分	4.8
Line 2: 'CLASS_DEF' should be separated from previous statement. [EmptyLineSeparator]	

他的代码：

做题用时: 21 分钟　　语言：Java　　运行时间：41ms　　占用内存：10848K　　程序状态：答案正确

```
import java.util.*;
public class Main {
    public static void main(String[] args){
        HashMap<Character,Integer> map = new HashMap<>();
        Scanner scan = new Scanner(System.in);
        String str = scan.next();
        for(int i = 0;i < str.length();i++){
            char ch = str.charAt(i);
            if(map.get(ch) == null){
                map.put(ch,1);
            }else{
                int val = map.get(ch);
                map.put(ch,val + 1);
            }
        }

        for(int i = 0;i < str.length();i++){
            if(map.get(str.charAt(i)) == 1){
                System.out.println(str.charAt(i));
                return ;
            }
        }
        System.out.println(-1);
    }
}
```

点此或手机扫描二维码查看代码编写过程

监控截图

代码相似度

编程题1相似代码

相似代码来源考生：黄超　　试卷：比特就业课Java方向笔试强训48天day22_11月10日（105Java）　　相似度：95.65%　　通过时间提前：17:19:21

本代码

```
import java.util.*;
public class Main {
    public static int func(int a,int b){
        //求最大公约数
        while(b != 0){
            int mod = a%b;
            a = b;
            b = mod;
        }
        return a;
    }
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        while(scan.hasNext()){
            int n = scan.nextInt();
            int iniVal = scan.nextInt();//初始能力值
            int[] arr = new int[n];
            for(int i = 0;i < n;i++){
                arr[i] = scan.nextInt();
            }
            for(int i = 0;i < n;i++){
                if(arr[i] <= iniVal){
                    iniVal += arr[i];
                }else{
                    iniVal += func(arr[i],iniVal);
                }
            }
            System.out.println(iniVal);
        }
    }
}
```

相似代码

```
import java.util.Scanner;

// 注意类名必须为 Main, 不要有任何 package xxx 信息
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while(sc.hasNext()){
            int m = sc.nextInt();
            int n = sc.nextInt();
            int[] arr = new int[m];
            for(int i = 0;i<m;i++){
                arr[i] = sc.nextInt();
            }
            for(int i = 0;i<m;i++){
                if(n >= arr[i]){
                    n += arr[i];
                }else{
                    n += solve(arr[i] , n);
                }
            }
            System.out.println(n);
        }
    }
    public static int solve(int m , int n){
        while(n != 0){
            int c = m % n;
            m = n;
            n = c;
        }
        return m;
    }
}
```