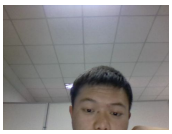
每日一题Java方向day13_12月7日-任栋-测评结果

考生信息



任栋

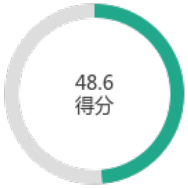
投递编号：2 | 学校：西安理工大学 | 邮箱：1104580363@qq.com | 职位：2020大四春招冲刺班 |

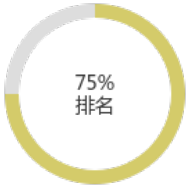
参考区域: 陕西省西安市（221.11.20.102） |

做题用时：02:11:59(2020-12-06 23:23:48开始答题，2020-12-07 22:58:00交卷) | 作答设备： PC |

已同意诚信声明和隐私协议

考生成绩





题型	得分	正确题数	排名	用时	是否阅卷
单选	40.0	8	22	00:13:43	--
编程	8.6	0	94	01:34:24	--

知识点技能图谱

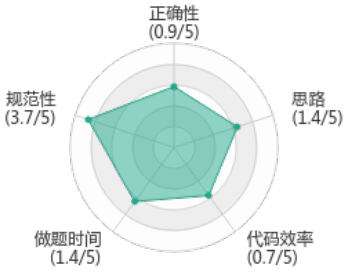
知识点	得分	正确题数
其他知识点	3.6	0
Java	40.0	8
编程基础	5.0	1
字符串	5.0	0

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	大四春招冲刺班JavaSE考试	31.0%	26.0/60	单选:26.0分	否	2020-10-28 17:33:16	2020-10-29 10:30:13
2	大四春招冲刺班数据结构考试	77.0%	22.0/60	单选:22.0分 编程:0.0分	否	2020-11-23 11:55:15	2020-11-24 10:40:01
3	Java方向每日一题day02_11月24日	70.0%	55.0/100	单选:30.0分 编程:25.0分	否	2020-11-23 12:10:19	2020-11-25 15:36:05
4	Java方向每日一题day03_11月25日	79.0%	55.0/100	单选:30.0分 编程:25.0分	否	2020-11-24 15:19:25	2020-11-24 22:21:11
5	Java方向每日一题day04_11月26日	61.0%	85.0/100	单选:35.0分 编程:50.0分	否	2020-11-25 14:58:20	2020-11-25 23:17:45

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
6	Java方向每日一题day05_11月27日	16.0%	85.0/100	单选:35.0分 编程:50.0分	否	2020-11-25 15:54:38	2020-11-26 23:32:02
7	Java方向每日一题day06_11月28日	43.0%	77.5/100	单选:30.0分 编程:47.5分	否	2020-11-27 14:19:26	2020-11-27 22:53:24
8	Java方向每日一题day07_11月30日	1.0%	100.0/100	单选:50.0分 编程:50.0分	否	2020-11-29 13:58:20	2020-11-30 22:17:40
9	Java方向每日一题day08_12月1日	28.0%	90.0/100	单选:40.0分 编程:50.0分	是，代码抄袭	2020-11-30 10:48:03	2020-12-01 22:29:01
10	Java方向每日一题day09_12月2日	86.0%	50.0/100	单选:35.0分 编程:15.0分	否	2020-12-01 10:43:40	2020-12-02 21:12:07
11	Java方向每日一题day10_12月3日	44.0%	75.0/100	单选:25.0分 编程:50.0分	否	2020-12-02 12:27:01	2020-12-04 09:14:04
12	Java方向每日一题day11_12月4日	72.0%	57.14/100	单选:25.0分 编程:32.14分	否	2020-12-03 10:46:54	2020-12-04 11:05:57
13	Java方向每日一题day12_12月5日	66.0%	60.0/100	单选:35.0分 编程:25.0分	否	2020-12-04 10:43:45	2020-12-05 21:50:32

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	14%	15	00:40:10	Java	26ms	10708K			70%
编程题2	20%	9	00:54:14	Java	32ms	10676K			53%

1 [平均分2.6分 | 60人正确/115人做题 | 用时：<1分 得分：0.0 / 5.0

下面有关JVM内存，说法错误的是？

A 程序计数器是一个比较小的内存区域，用于指示当前线程所执行的字节码执行到了第几行，是线程隔离的

B Java方法执行内存模型，用于存储局部变量，操作数栈，动态链接，方法出口等信息，是线程隔离的

C 方法区用于存储JVM加载的类信息、常量、静态变量、即时编译器编译后的代码等数据，是线程隔离的

D 原则上讲，所有的对象都在堆区上分配内存，是线程之间共享的

他的回答： D (错误)

正确答案： C

2 [平均分4.7分 | 108人正确/114人做题 | 用时：2分 得分：5.0 / 5.0

下列程序段的输出结果是：（ ）

```
public void complicatedexpression_r(){
    int x=20, y=30;
    boolean b;
    b = x > 50 && y > 60 || x > 50 && y < -60 || x < -50 && y > 60 || x < -50 && y < -60;
    System.out.println(b);
}
```

A true

B false
C 1
D 0

他的回答： B (正确)

正确答案： B

3 [平均分4.7分 | 107人正确/114人做题 | 用时：<1分 得分：5.0 / 5.0

输入流将数据从文件，标准输入或其他外部输入设备中加载道内存，在 java 中其对应于抽象类（ ）及其子类。

A java.io.InputStream
B java.io.OutputStream
C java.os.InputStream
D java.os.OutputStream

他的回答： A (正确)

正确答案： A

4 [平均分3.5分 | 80人正确/113人做题 | 用时：2分 得分：5.0 / 5.0

以下程序的输出结果是

```
public class Print{
 static boolean out(char c){
  System.out.println(c);
  return true;
 }
 public static void main(String[] argv){
  int i = 0;
  for(out('A');out('B') && (i<2);out('C')){
   i++;
   out('D');
  }
 }
}
```

A ABDCBDCB
B BCDABCD
C 编译错误
D 运行错误

他的回答： A (正确)

正确答案： A

5 [平均分1.7分 | 39人正确/113人做题 | 用时：<1分 得分：0.0 / 5.0

下面关于程序编译说法正确的是（ ）

A java语言是编译型语言，会把java程序编译成二进制机器指令直接运行
B java编译出来的目标文件与具体操作系统有关
C java在运行时才进行翻译指令
D java编译出来的目标文件，可以运行在任意jvm上

他的回答： D (错误)

正确答案： C

6 [平均分4.4分 | 101人正确/115人做题 | 用时：<1分 得分：5.0 / 5.0

（ ）仅包含方法定义和常量值。

A 接口
B 变量
C 单元
D 成员

他的回答： A (正确)

正确答案： A

---

7 [平均分3.1分 | 70人正确/114人做题 | 用时：<1分 得分：5.0 / 5.0

下面程序的运行结果：()

```
public static void main(String args[]) {
  Thread t=new Thread(){
  public void  run(){
   dianping();

  }
 };
 t.run();
 System.out.print("dazhong");
 }
 static void dianping(){
  System.out.print("dianping");
}
```

A dazhongdianping
B dianpingdazhong
C a和b都有可能
D dianping循环输出，dazhong夹杂在中间

他的回答： B (正确)

正确答案： B

---

8 [平均分3.4分 | 77人正确/114人做题 | 用时：<1分 得分：5.0 / 5.0

```
public interface IService {String NAME="default";}
```

默认类型等价表示是哪一项:

A public String NAME="default";
B public static String NAME="default";
C public static final String NAME="default";
D private String NAME="default";

他的回答： C (正确)

正确答案： C

---

9 [平均分3.1分 | 72人正确/115人做题 | 用时：<1分 得分：5.0 / 5.0

有以下类定义：

```
abstract class Animal{
 abstract void say();
}
public class Cat extends Animal{
 public Cat(){
  System.out.printf("I am a cat");
```

```
}
public static void main(String[] args) {
 Cat cat=new Cat();
}
}
```

运行后：

A I am a cat
B Animal能编译，Cat不能编译
C Animal不能编译，Cat能编译
D 编译能通过，但是没有输出结果

他的回答： B (正确)

正确答案： B

---

10 [平均分4.0分 | 91人正确/114人做题 | 用时：2分 得分：5.0 / 5.0

类Test1定义如下:

```
public class Test1{//1
    public float aMethod(float a,float b){}//2
    //3
}//4
```

将以下哪种方法插入行3是不合法的。

A public int aMethod(int a,int b){}
B private float aMethod(int a,int b,int c){}
C public float aMethod(float a,float b){}
D public float aMethod(float a,float b,float c){}

他的回答： C (正确)

正确答案： C

---

11 ACM编程题 语言限制 [平均分19.2分 | 80人正确/108人做题 | 提交: 15 次 得分：3.6 / 25.0

标题：跟奥巴马一起编程(15) | 时间限制：1秒 | 内存限制：32768K | 语言限制： 不限

【跟奥巴马一起编程(15)】美国总统奥巴马不仅呼吁所有人都学习编程，甚至以身作则编写代码，成为美国历史上首位编写计算机代码的总统。2014年底，为庆祝“计算

机科学教育周”正式启动，奥巴马编写了很简单的计算机代码：在屏幕上画一个正方形。现在你也跟他一起画吧！

输入描述：

输入在一行中给出正方形边长N（3<=N<=20）和组成正方形边的某种字符C，间隔一个空格。

输出描述：

输出由给定字符C画出的正方形。但是注意到行间距比列间距大，所以为了让结果看上去更像正方形，我们输出的行数实际上是列数的50%（四舍五入取整）。

示例1：

输入

```
10 a
```

输出

```
aaaaaaaaaa
a a
a a
a a
```

aaaaaaaaaa

代码片段

功能实现	TA的	平均
总通过率	14%	76%
基本测试用例通过率	1/4 (25%)	77%
边缘测试用例通过率	0/3 (0%)	75%

代码提交统计	TA的	平均
使用语言	Java	
做题用时	00:40:10	00:33:46
提交次数	15	6

代码执行统计

格式错误 ：10
答案错误 ：2
返回非零 ：2
编译错误 ：1

代码效率	TA的	参考
运行时间	26ms	1s
占用内存	10708K	32768K

代码规范及可读性

代码规范得分 3.2

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

Line 8:17: Local variable name 'a' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [LocalVariableName]

Line 9:20: Local variable name 'b' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [LocalVariableName]

Line 16: 'METHOD_DEF' should be separated from previous statement. [EmptyLineSeparator]

Line 16:35: Parameter name 'a' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

Line 16:45: Parameter name 'b' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

Line 22: 'METHOD_DEF' should be separated from previous statement. [EmptyLineSeparator]

Line 22:35: Parameter name 'a' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

Line 22:45: Parameter name 'b' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [ParameterName]

他的代码：

做题用时: 40 分钟　　语言：Java　　运行时间：26ms　　占用内存：10708K　　程序状态：格式错误

```
import java.util.*;
public class Main{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (scanner.hasNextLine()){
            String st = scanner.nextLine();
            String[] str = st.split(" ");
            int a = Integer.parseInt(str[0]);
            String b = str[1];
            func1(a, b);
            System.out.println();
            func2(a, b);
            func1(a, b);
        }
    }
    private static void func2(int a, String b) {
        for (int i = 0; i < a/2-2; i++) {
            System.out.println(b + ' ' + b);
            System.out.println();
        }
    }
    private static void func1(int a, String b) {
        for (int i = 0; i < a; i++) {
```

```
            System.out.print(b);
        }
    }
}
```

点此或手机扫描二维码查看代码编写过程

12 ACM编程题 语言限制 [平均分16.5分 | 62人正确/94人做题 | 提交: 9 次 得分：5.0 / 25.0

标题：超长正整数相加 | 时间限制：1秒 | 内存限制：32768K | 语言限制： 不限

【超长正整数相加】
请设计一个算法完成两个超长正整数的加法。

接口说明

```
/*
请设计一个算法完成两个超长正整数的加法。
输入参数：
String addend：加数
String augend：被加数
返回值：加法结果
*/
public String AddLongInteger(String addend, String augend)
{
    /*在这里实现功能*/

 return null;
}
```

输入描述：

> 输入两个字符串数字

输出描述：

> 输出相加后的结果，string型

示例1：

输入

> 99999999999999999999999999999999999999999999999999
> 1

输出

> 100000000000000000000000000000000000000000000000000

代码片段

功能实现	代码提交统计	代码执行统计

	TA的	平均
总通过率	20%	66%
基本测试用例通过率	1/6 (17%)	66%
边缘测试用例通过率	1/4 (25%)	66%

	TA的	平均
使用语言	Java	
做题用时	00:54:14	00:32:35
提交次数	9	4

答案错误 ：3
返回非零 ：2
编译错误 ：4

代码效率

	TA的	参考
运行时间	32ms	1s
占用内存	10676K	32768K

代码规范及可读性

代码规范得分 4.2

Line 3: 'CLASS_DEF' should be separated from previous statement. [EmptyLineSeparator]

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

Line 12:26: Method name 'AddLongInteger' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9_]*$'. [MethodName]

Line 24:21: Local variable name 'a' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [LocalVariableName]

他的代码：

做题用时: 54 分钟　语言：Java　运行时间：32ms　占用内存：10676K　程序状态：答案错误

```
import java.util.Scanner;
import java.util.Stack;
public class Main{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (scanner.hasNextLine()){
            String addend = scanner.nextLine();
            String augend = scanner.nextLine();
            AddLongInteger(addend,augend);
        }
    }
    public static String AddLongInteger(String addend ,String augend) {
        Stack<Integer> stack1 = new Stack<>();
        Stack<Integer> stack2 = new Stack<>();
        StringBuffer sum = new StringBuffer();
        for (int i = 0; i <addend.length() ; i++) {
            stack1.add(Integer.parseInt(String.valueOf(addend.charAt(i))));
        }
        for (int i = 0; i <augend.length() ; i++) {
            stack2.add(Integer.parseInt(String.valueOf(augend.charAt(i))));
        }
        while (!stack1.isEmpty()){
            if (!stack2.isEmpty()){
                int a = stack1.pop()+stack2.pop();
                if (a < 10){
                    sum.append(a);
                }else {
                    sum.append(a % 10);
                    stack1.add(stack1.pop()+1);
                }
            }else {
                if (stack1.peek()<10){
                    while (!stack1.isEmpty()){
                        sum.append(stack1.pop());
                    }
                }else {
                    if (stack1.size() != 1){
```

```
                    sum.append(stack1.pop()%10);
                    stack1.add(stack1.pop()+1);
                }else {
                    sum.append(stack1.peek()%10);
                    sum.append(stack1.pop()/10);
                }
            }
        }
    }
    System.out.println(sum.reverse().toString());
    return sum.reverse().toString();
  }
}
```

点此或手机扫描二维码查看代码编写过程