**华东交通大学****软件+轨道信号阶段作业一**

**班级:软件工程（轨道交通信号）2班**

**姓名： 张凌宇**

**成绩：**

**一、问答题（每小题10分，共100分）**

1. System.out.println()和System.out.print()有什么区别？

换行: System.out.println() 在输出后会自动换行，而 System.out.print() 则不会。

参数要求: System.out.println() 可以不写参数，而 System.out.print() 的参数不能为空，必须有。

输出效果: System.out.println() 输出后光标会移动到下一行的开始位置，而 System.out.print() 则将光标定位在所显示的最后一个字符之后。

1. Java的注释方式有哪几种，格式为何？

Java 语言提供三种添加注释的方法，分别为单行注释、多行注释和文档注释。 “//”为单行注释标记。 从符号“//”开始直到换行为止的所有内容均作为注释而被编译器忽略。 “/\* \*/”为多行注释标记，符号“/\*”与“\*/”之间的所有内容均为注释内容。 注释中的内容可以换行。 \*… “/\*\* \*/”为文档注释标记。 符号“/\*\*”与“\*/”之间的内容均为文档注释内容。

1. Java的基本数据类型有哪些？String是基本数据类型吗？

tring 是 Java 中的一种类，不是基本数据类型，而是引用数据类型。Java 中的基本数据类型有八种，分别是 boolean、char、byte、short、int、long、float 和 double。 基本数据类型拥有直接的值，而非对象的引用。 而 String 类是基于字符序列的类，可以包含任何字符。 在使用时，需要使用 new 操作创建 String 类对象，将其赋值给变量。

1. 判断如下程序的运行结果

*//如下代码的计算结果是：*

int i = 1;

i \*= 0.2;

i++;

System.out.println("i=" + i);

i=1

1. 说明break和continue使用上的相同点和不同点

**功能：break 用于完全终止循环，而 continue 用于跳过当前迭代的剩余部分**。 影响范围：break 会终止最近的循环，而 continue 仅影响当前迭代。 用途：break 通常用于在满足某个条件时提前结束循环，而 continue 用于忽略某些不需要处理的迭代。 控制流：当 break 被执行时，控制流会立即跳出循环；而 continue 会跳过当前迭代的剩余部分，并开始下一次迭代。

1. switch后面使用的表达式可以是哪些数据类型？

byte、short、char、int，包装类如Byte、Short、Character、Integer，枚举类型Enum，以及从JDK1.7开始支持的字符串类型String

1. 不同类型的一维数组元素的默认初始化值各是多少？

1. 基本数据类型数组

整数类型（byte、short、int、long）：默认值为 0（对应类型的零值，如 long 为0L）

浮点类型（float、double）：默认值为 0.0（float 为0.0f，double 为0.0d）

字符类型（char）：默认值为 '\u0000'（空字符，即 ASCII 码为 0 的字符）

布尔类型（boolean）：默认值为 false

2. 引用数据类型数组

包括对象数组（如String[]、Integer[]）、自定义类数组等，默认值均为 null（表示未引用任何对象）

1. 数组中常见的异常有哪些？请举例说明

数组索引越界异常

如：int[] arr = new int[3]; // 数组长度为3，有效索引为0、1、2

System.out.println(arr[3]); // 索引3超出范围，抛出此异常

空指针异常

如：String[] strArr = null; // 数组未初始化，指向null

System.out.println(strArr.length); // 访问null的属性，抛出此异常

类型转换异常

如：Object[] objArr = new String[2]; // 声明为Object数组，实际存储String

objArr[0] = 100; // 存入Integer类型，与String不兼容，抛出此异常

负数数组长度异常

如：int[] arr = new int[-5]; // 长度为负数，初始化时抛出此异常

1. 异常(Exception)分为哪两大类？

受检查异常与非受检查异常

10、throw与throws的区别？

1.作用不同

throw：用于主动抛出一个具体的异常对象，直接在代码中触发异常。

throws：用于声明方法可能抛出的异常类型，告知调用者该方法有潜在异常需要处理。

2. 使用位置不同

throw：出现在方法体内部，可以在任何执行语句的地方使用（通常配合条件判断）。

throws：出现在方法声明的末尾，紧跟在方法参数列表之后。

3. 语法格式不同

throw：后面跟单个异常对象（必须是Throwable类或其子类的实例）。throws：后面跟一个或多个异常类型（用逗号分隔），表示方法可能抛出这些类型的异常。

4. 数量不同

throw：一次只能抛出一个异常对象。

throws：可以声明多个异常类型，用逗号分隔。

5. 目的不同

throw：主动触发异常，中断当前代码流程，将异常传递给上层调用者处理。

throws：声明异常，告知方法的调用者 “此方法可能会抛出这些异常，请做好处理准备”（调用者需用try-catch捕获或继续用throws声明）。

.throw 是执行动作，抛出具体异常；

throws 是声明说明，告知可能抛出的异常类型。