题型：

选择	20-30
填空	10-20
简答	20
阅读	20
综合（设计）	20-30

知识点参考：

1. 知识点（选择填空，部分阅读和设计）
   Java 类继承的特点
   Java 基础数据类型
   Java 标识符构成
   常用运算符使用
   接口和抽象类的使用
   常用的修饰符及其区别，比如 public、protected、final 等
   内部类的定义与使用
   线程的基本概念及线程的使用
   常用的布局管理器
   字符串与基本类型的转换
   Java 语言面向对象的特点
   GUI 编程常用包
   this 与 super 的使用
   构造方法
   类继承及接口实现方法
   Java 字符集
   常用事件接口比如 ActionListener，实现方法及获取事件源方法
   简单选择结构及 Switch 结构的使用
   四-六章课后大题
2. 以下知识点（主要是简答）

- 编程语言的几个发展阶段:

面向机器语言
面向过程语言
面向对象语言
面向对象编程主要体现下列三个特性：封装性 ；继承；多态
类是 Java 语言中最重要的“数据类型”，类声明的变量被称作对象（见后面的 4.3 节），即类是用来创建对象的模板。
类的实现包括两部分：类声明和类体。
类体的内容由两部分构：一部分是变量的声明，用来刻画属性；另一部分是方法的定义，用

来刻画行为功能。

- 成员变量和局部变量

类体中变量定义部分所定义的变量被称为类的成员变量。
成员变量在整个类内都有效，其有效性与它在类体中书写的先后位置无关。
成员变量在定义时有默认值。
在方法体中定义的变量和方法的参数被称为局部变量。
局部变量只在定义它的方法内有效。
局部变量在定义时没有默认值。

- 成员变量又分为实例成员变量（简称实例变量）和类变量（也称静态变量）。

如果成员变量的类型前面加上关键字 static，这样的成员变量称做是类变量或静态变量。
其他的变量统称为实例变量。
类中的方法也可分为实例方法和类方法. 方法声明时，方法类型前面不加关键字 static 的是实例方法、加关键字 static 的是类方法。

- 实例方法和类方法的区别

1）对象调用实例方法

当对象调用实例方法时，该方法中出现的实例变量就是分配给该对象的实例变量；该方法中出现的类变量也是分配给该对象的变量，只不过这个变量和所有的其他对象共享而已。

2）类名调用类方法

从而类方法不仅可以被类创建的任何对象调用执行，也可以直接通过类名调用。和实例方法不同的是，类方法不可以操作实例变量，这是因为在类创建对象之前，实例成员变量还没有分配内存。

- 实例变量和类变量的区别

不同对象的实例变量互不相同
所有对象共享类变量
通过类名直接访问类变量
构造方法是一种特殊方法，它的名字必须与它所在的类的名字完全相同，而且没有类型。

方法返回的数据类型可以是 Java 中的任何数据类型之一，当一个方法不需要返回数据时，返回类型必须是 void。

- 方法重载的意思是：一个类中可以有多个方法具有相同的名字，但这些方法的参数必须不同，即或者是参数的个数不同，或者是参数的类型不同。方法的返回类型和参数的名字不参与比较

方法重载是一种多态的体现。

- 定义类需要注意的地方：

1)对成员变量的操作只能放在方法中，方法可以对成员变量和该方法体中声明的局部变量进行操作。在声明成员变量时可以同时赋予初值，但是不可以在类体中有单独的赋值语句，但局部变量必须赋初值。
2)实例方法既能对类变量操作也能对实例变量操作，而类方法只能对类变量进行操作。
3)一个类中的方法可以互相调用，实例方法可以调用该类中的其他方法，类中的类方法只能

调用该类的类方法。

- 构造方法：

构造方法是一种特殊方法，它的名字必须与它所在的类的名字完全相同，而且没有类型。
允许一个类中编写若干个构造方法，但必须保证他们的参数不同，即参数的个数不同，或者是参数的类型不同。
他的作用是在创建对象时使用，主要是用来初始化各个成员变量，以便给类所创建的对象一个合理的初始状态。

对于基本数据类型的参数，向该参数“传值”，传递的是值的拷贝.
对于参数是引用类型时，“传值”传递的是变量的引用而不是变量所引用的实体。Java 的引用型数据包括对象、数组和接口。

方法重载的意思是：一个类中可以有多个方法具有相同的名字，但这些方法的参数必须不同，即或者是参数的个数不同，或者是参数的类型不同。

如果一个类想要使用的那个类和它不在一个包中，要使用 import 语句完成使命。

- 访问限制修饰符有 private、protected 和 public，都是 Java 的关键字，用来修饰成员变量或方法。

用关键字 private 修饰的成员变量和方法称为私有变量和私有方法。
对于私有成员变量或方法，只有在本类中创建该类的对象时，这个对象才能访问自己的私有成员变量和类中的私有方法。
用 public 修饰的成员变量和方法被称为共有变量和共有方法 。
我们在任何一个类中用类 Tom 创建了一个对象后，该对象能访问自己的 public 变量和类中的 public 方法（也可以通过类名来操作成员变量、方法） 。
当在另外一个类中用类 Tom 创建了一个对象后，如果这个类与 Tom 类在同一个包中，那么该对象能访问自己的友好变量和友好方法。
在任何一个与 Tom 同一包中的类中，也可以通过 Tom 类的类名访问 Tom 类的类友好成员变量和类友好方法。
类声明时，如果在关键字 class 前面加上 public 关键字，就称这样的类是一个 public 类 。
可以在任何另外一个类中,使用 public 类创建对象。
如果一个类不加 public 修饰，这样的类被称作友好类。
在另外一个类中使用友好类创建对象时，要保证它们是在同一包中。

第五章

- 继承是一种由已有的类创建新类的机制。利用继承，我们可以先创建一个共有属性的一般类，根据该一般类再创建具有特殊属性的新类，新类继承一般类的状态和行为，并根据需要增加它自己的新的状态和行为。由继承而得到的类称为子类，被继承的类称为父类（超类）。

Java 的类按继承关系形成树形结构这个树形结构中，根节点是 Object 类（Object 是 java.lang 包中的类），即 Object 是所有类的祖先类

- 子类创建对象时，子类的构造方法总是先调用父类的某个构造方法，完成父类部分的创

建；然后再调用子类自己的构造方法，完成子类部分的创建。如果子类的构造方法没有明显地指明使用父类的哪个构造方法，子类就调用父类的不带参数的构造方法 。

使用 super 调用父类的构造方法

可以使用 final 将类声明为 final 类。final 类不能被继承，即不能有子类。

- 简述上转型对象和接口回调

假设，A 类是 B 类的父类，当用子类创建一个对象，并把这个对象的引用放到父类的对象中时，称对象 a 是对象 b 的上转型对象。
接口回调是指：可以把实现某一接口的类创建的对象的引用赋给该接口声明的接口变量中，那么该接口变量就可以调用被类重写的接口方法。实际上，当接口变量调用被类重写的接口方法时，就是通知相应的对象调用这个方法。

- 上转型对象的使用

1.上转型对象不能操作子类新增的成员变量；不能调用子类新增的方法。
2.上转型对象可以访问子类继承或隐藏的成员变量，也可以调用子类继承的方法或子类重写的实例方法。
3. 如果子类重写了父类的某个实例方法后，当用上转型对象调用这个实例方法时一定是调用了子类重写的实例方法。

- abstract 类有如下特点

和普通的类相比，abstract 类里可以有 abstract 方法。也可以没有。对于 abstract 方法，只允许声明，不允许实现，而且不允许使用 final 修饰 abstract 方法。
对于 abstract 类，不能使用 new 运算符创建该类的对象，只能产生其子类，由子类创建对象。
如果一个类是 abstract 类的子类，它必须具体实现父类的所有的 abstract 方法。

使用关键字 interface 来定义一个接口，Java 使用了接口，一个类可以实现多个接口。

- 接口的变量和方法的构成规则：

接口中的变量自动都是 public、static，final，
接口中的方法默认为 public abstract；接口也产生 class 文件。
接口中的方法不能被 static 和 final 修饰，因为要重写所有接口中的方法。
接口中没有构造函数，方法可以抛出异常。
一个类通过使用关键字 implements 声明自己实现一个或多个接口。

- abstract 类与接口的比较

接口和 abstract 类的比较如下：
1．abstract 类和接口都可以有 abstract 方法。
2．接口中只可以有常量,不能有变量；而 abstract 类中即可以有常量也可以有变量。
3．abstract 类中也可以有非 abstract 方法,接口不可以。

- 内部类的使用规则：

声明内部类如同在类中声明方法或变量一样，一个类把内部类看作是自己的成员。
外嵌类的类体中可以用内部类声明的对象，作为外嵌类的成员。
外嵌类的成员变量在内部类中仍然有效，内部类中的方法也可以调用外嵌类中的方法。
内部类的类体中不可以声明类变量和方法。
外嵌类和内部类在编译时，生成两个.class 文件。

Java 把 InputStream 抽象类的子类创建的流对象称作字节输入流；OutputStream 抽象类的子类创建的流对象称作字节输出流。Java 把 Reader 抽象类的子类创建的流对象称作字符输入流；Writer 抽象类的子类创建的流对象称作字符输出流。

使用输入流通常包括 4 个基本步骤：

(1)设定输入流的源
(2)创建指向源的输入流
(3)让输入流读取源中的数据
(4)关闭输入流。

- 简述程序、进程和线程区别。

程序是一段静态的代码，它是应用软件执行的蓝本。
进程是程序的一次动态执行过程，它对应了从代码加载、执行至执行完毕的一个完整过程，这个过程也是进程本身从产生、发展至消亡的过程。
线程是比进程更小的执行单位，一个进程在其执行过程中，可以产生多个线程，形成多条执行线索，每条线索，即每个线程也有它自身的产生、存在和消亡的过程。

- 线程的一个完整的生命周期要经历如下的四种状态：

1.新建: 当一个 Thread 类或其子类的对象被声明并创建时，新生的线程对象处于新建状态。
2．运行 :线程必须调用 start（）方法（从父类继承的方法）通知 JVM，这样 JVM 就会知道又有一个新一个线程排队等候切换了。一旦轮到它来享用 CPU 资源时，此线程的就可以脱离创建它的主线程独立开始自己的生命周期了。
3．中断（阻塞）:
4．死亡 :处于死亡状态的线程不具有继续运行的能力。线程释放了实体。

综合题：
选择结构
循环结构
类及类的继承
文件读写
多线程编程