**复习知识点**

|  |  |  |  |
| --- | --- | --- | --- |
| **题型** | **分值/题** | **数量** | **各题分值** |
| 1．选择题 | 2 | 20 | 40 |
| 2．填空题 | 2 | 5 | 10 |
| 3．程序阅读题 | 4 | 5 | 20 |
| 4．简答题 | 5 | 2 | 10 |
| 5．程序设计 | 10 | 2 | 20 |

1. Java语言的特点：简单、面向对象、平台无关、多线程、动态
2. 标识符命名规则。（选择题）
3. 开发与运行Java程序的三步骤：编辑源程序，编译生成字节码，解释运行字节码
4. java程序文件名命名规则（java源程序文件名与应用程序类名），及文件扩展名（选择题）、main方法：

1、Java 程序源文件的后缀必须是 .java，不能是其他文件后缀名。

2、如果 Java 程序源代码里定义了一个 public 类，则该源文件的主文件名必须与该 public 类（也就是该类定义使用了 public 关键字修饰）的类名相同。

3、如果 Java 程序源代码里没有定义任何 public 类，那么 Java 程序源程序的主文件名可以是任意的。

1. 整型（十进制、八进制、十六进制）合法常量的表示方法、字符型常量（单个字符、字符串常量）的表示方法、实型合法常量的表示方法。字符采用的编码（填空题）。
2. 混合数据类型运算的表达式求值：（填空题、选择题、阅读程序）
3. 掌握教材提到的常用运算符的运算规则，以及运算对象（如 %、/ 、逻辑运算符&&和||等）的特殊要求，包括复合赋值运算符（+=，-=，\*= 等）
4. 数据类型转换：自动转换条件，何种情况下需要强制转换及语法格式
5. 条件运算符的使用：

条件运算符（ ? : ）也称为 “三元运算符”。

语法形式：布尔表达式 ？ 表达式1 ：表达式2

运算过程：如果布尔表达式的值为 true ，则返回 表达式1 的值，否则返回 表达式2 的值

1. 掌握分支结构与循环结构程序设计（选择题、程序阅读题、编程题）
2. if语句和与else的配对，
3. switch语句
4. 循环语句的结构（for、while、do-while）
5. break和continue语句return语句使用的用法。
6. 以及它们的组合使用。
7. 一维数组的定义与使用；二维数组的定义与使用；数组元素的访问
8. Java面向对象编程

掌握类的定义（成员变量、成员方法和构造方法）和使用，对象创建

实例变量与局部变量的区别（简答题）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 变量 | 定义位置 | 初始化值 | 生命周期 | 作用范围 | 内存位置 |
| 类变量 | 类中方法外以static关键字声明 | 有默认的初始化的值 | 第一次访问时创建,在程序结束时销毁 | 整个类中有效 | 静态存储区 |
| 实例变量 | 类中方法外 | 有默认的初始化的值 | 对象的存在而存在,随着对象的消失而消失 | 整个类中有效 | 堆内存 |
| 局部变量 | 方法内 | 不赋值不能用 | 方法的调用而存在,方法调用完毕而消失 | 所在方法有用 | 栈内存 |

类与对象的关系（简答题）

类是对象的抽象，而对象是类的具体实例。类是抽象的，不占用内存，而对象是具体的，占用存储空间。类是用于创建对象的蓝图，它是一个定义包括在4102特定类型的对象中的方法和变量的软件模板。

package与import的用法

import表示的是导入包，两个不同包下的类在互相调用时，需要导入调用的这个类所在的包才能进行调用。

如：类A在apackage这个包下，类B在bpackage这个包下，A调用B的代码5261如下：

package apackage;

import bpackage.B;

public class A {

public static void main(String[] args) {

B b = new B();

}

}

package表示的是包名，当一个4102类定义在一个包下时，代码中需要引入包名来标注，使用方法为package 加包名;

方法的调用与参数传递，参数传递的方法（值传递与引用传递）（程序阅读题）

this与super、final的各自作用（简答题）

this访问本类中的属性和方法，super访问父类中的属性和方法。 this调用本类构造，必须放在构造方法的首行。super调用父类构造，必须放在子类构造方法首行。Final是的某个方法值只能被引用不能被修改

类的访问修饰符有哪些（选择题）

public.protected.default.private四个，定义类时，只能写一个。

类的私有成员和公共成员

公共成员用public来修饰，可以被所有类访问

私有成员用private来修饰，它只允许在自己类的内部被访问

成员的访问修饰符：private、protect、public、缺省修饰符，他们的访问权限各是什么（选择题）



1. 方法的重载的格式 ：P81
2. 构造方法的作用与定义，构造方法的重载与调用

作用：对类的初始化

1. 静态成员变量和静态成员方法的使用 （选择题，程序阅读题）
2. 继承与多态的概念（简答题）

继承：继承是面向对象最显著的一个特性。继承是从已有的类中派生出新的类，新的类能吸收已有类的数据属性和行为，并能扩展新的能力。

多态：多态指同一个实体同时具有多种形式。多态就是允许方法重名 参数或返回值可以是父类型传入或返回。

1. 子类的创建（注意，在子类构造方法中自动调用父类构造方法），上转型对象（程序阅读题）
2. 方法的重载与重写（也称为覆盖）的区别（简答题）

重载在同一类中，在一个类里面为一种行为提供多种实现方式，并提高可读性

重写在子类和父类间，父类方法无法满足子类的要求，子类通过方法重写满足要求



1. 抽象类的定义

抽象类的基本概念

抽象方法，只声明而未实现的方法我们称为抽象方法，所有的抽象方法都需要用abstract关键字声明，包含抽象方法的类也需要使用abstract关键字声明

1. 接口与接口的实现，利用接口实现多重继承
2. 接口与抽象类的区别（简答题）

抽象类要被子类继承，接口要被类实现。

接口只能做方法声明，抽象类中可以作方法声明，也可以做方法实现

接口里定义的变量只能是公共的静态的常量，抽象类中的变量是普通变量

接口是设计的结果，抽象类是重构的结果。  
抽象类和接口都是用来抽象具体对象的，但是接口的抽象级别最高

抽象类可以有具体的方法和属性，接口只能有抽象方法和不可变常量。

抽象类主要用来抽象类别，接口主要用来抽象功能。

1. 程序对错误与异常的处理方式，异常处理的基本结构try-catch-finally使用，throw和throws的使用及各自的作用（选择题, 程序阅读题,）

no Excaption finally Catch finally

1. Scanner的输入输出常用方法，能以键盘、文件实现输入。熟悉常用文件字符/字节输入输出流类、随机流类、缓冲流类。（选择题）

<https://blog.csdn.net/qq_43228135/article/details/86698260>

1.输入流：

1）FileReader(String filename);

2) FileReader(File filename);

2.输出流：

1）FileWriter(String filename);

2）FileWriter(File filename);

3）FileWriter(String filename,boolean append);

4）FileWriter(File filename,boolean append);

缓冲流 ：**BufferedReader(Reader in);**

**随机流 ：RandomAccessFile(String name,String mode);**

1. 线程的生命周期，创建线程的两种方法，start()、run()、sleep()方法的功能（选择题）

线程的生命周期：新建、就绪、运行、阻塞、死亡

创建线程的两种方法：继承Thread类，实现Runnable接口

run(); 只是调用了一个普通方法，并没有启动另一个线程，程序还是会按照顺序执行相应的代码。

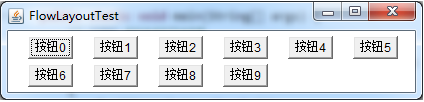
start(); 则表示，重新开启一个线程，不必等待其他线程运行完，只要得到cup就可以运行该线程。

sleep()； 就算线程的睡眠时间到了，他也不是立即会被运行，只是从睡眠状态变为了可运行状态，是不会由睡眠状态直接变为运行状态的

1. 常见布局管理器GridLayout， FlowLayout， BorderLayout的布局方式（选择题）

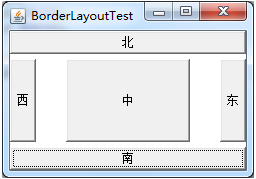
FlowLayout()

构造一个新的 FlowLayout，它是居中对齐的，默认的水平和垂直间隙是 5 个单位。



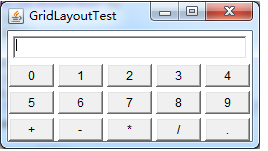
BorderLayout布局管理器

这是一个布置容器的边框布局，它可以对容器组件进行安排，并调整其大小，使其符合下列五个区域：北、南、东、西、中。每个区域最多只能包含一个组件



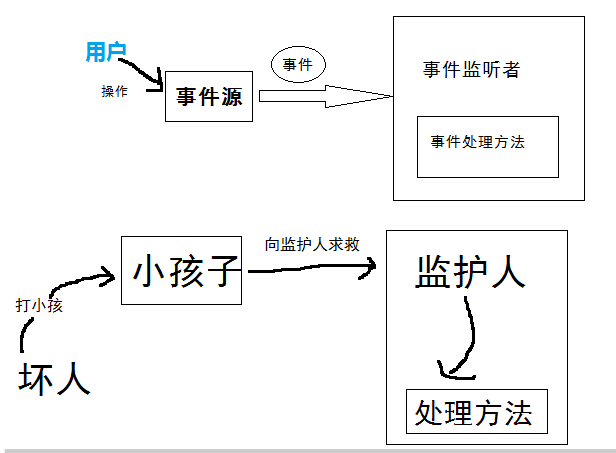
GridLayout布局管理器

建立一个表格的版面对象。rows代表有几行，cols代表有几列；hgap是组件之间的水平距离，vgap是组件之间的竖直距离



1. 掌握事件处理机制，熟悉为事件源注册监听器的方法，熟悉ActionEvent事件。（选择题）

事件处理机制：



注册监听器方法：

①编写事件处理类（事件监听者）

②根据需求给事件处理类实现监听接口

③在事件处理类中重写（实现），其事件处理函数

④在事件源类中指定该事件的监听器（响应者）是谁，即注册监听

ActionEvent，动作事件，不代表具体的动作，是一种语义。如按钮或菜单被鼠标单击，单行文本框中按下回车键等都可以看做是ActionEvent事件。如果用户的一个动作导致了某个组件最基本的动作发生了，这就是ActionEvent事件。

1. LinkedList、HashMap的使用（选择题）

ArrayList：数组集合。 查询、修改、新增（尾部新增）快，删除、新增（队列中间）慢，适用于查询、修改较多的场景。

LinkedList：双向链表集合。查询、修改慢（需要遍历集合），新增，删除快（只需要修改前后节点的链接即可），适用于新增、删除较多的场景。

HashMap：结合数组和链表的优势，期望做到增删改查都快速，时间复杂度接近于O(1)。当hash算法较好时，hash冲突较低。适用于增删改查所有场景。

编程题涉及相关内容：

1. 求数列之和（累积和）
2. 某个范围的素数
3. 某个范围的完全数
4. 求水仙花数
5. 统计字符串中某种字符的格式
6. 类的设计
7. 抽象类的设计并被子类中继承，以及抽象方法的重写。在应用程序中，创建子类对象并调用其方法。
8. 接口的设计并在类中实现，以及抽象方法的重写。在应用程序中，创建子类对象并调用其方法。
9. 类的两个组成部分：属性和方法。

类的属性是根据对象的特征进行定义的，反映的是对象的一些本质特征；类的方法分为两类，普通方法和构造方法，其中普通方法是根据对象的行为进行定义的，构造方法就是要重点学习的内容了。

Java语言中会默认提供给编程者一个无参数的构造方法，在使用默认的构造方法时，不能给构造方法提供任何参数。一旦给构造方法提供了参数，就是自定义了构造方法，此时，默认的构造方法会被覆盖而无法使用，用户只有使用自己定义的构造方法。

构造方法的格式是：

public 构造方法名（数据类型 参数名，…）{ }

有几点需要注意的地方：

1.构造方法不能定义返回值数据类型。

2.构造方法名必须和类名完全一致。构造方法的最重要的作用是实例化对象，同时也可以给属性设置初始值。

方法重载：现实生活中很多情况下都需要进行方法重载，在java语言中，普通方法和构造方法都可以进行重载，普通方法的重载比如像用碗装米、装水，又或者吃饭时吃的是蔬菜还是饺子；构造方法也可以重载，此时需要用代码来实例化每一个对象，比如贾宝玉出生时带玉，林黛玉出生时不带玉，此时也需要用到方法重载。想要实现方法重载有两个条件：

1.方法名必须完全一致

2.方法所带的参数个数、参数类型、参数顺序至少要有一个不同。

那么系统又是如何来调用重载后的方法呢？系统会根据方法所带的参数来决定调用不同的方法，假如同时存在多个方法，那么系统则根据参数的个数和种类来对相应的方法进行调用。

在java语言方法重载时，当参数名和下方的属性名完全一致时，java会自动采取就近原则进行调用，容易将属性和参数混淆，此时就会用到this关键字。

This关键字有三个作用：

1.this.属性名，当属性名和参数名完全一致时，可以用this关键字来做区分，加上this的表示属性，不加的表示参数。

2.this(参数值，…)，调用当前对应参数的构造方法。

3.this.普通方法名(参数值，…)，调用当前对象中的普通方法。

可是说到底this到底是谁呢？其实，this是用来表示对象的，谁在调用方法，方法中的this就表示谁。假设a对象在调用方法，这时候方法中的this就是a，这就好比生活中的“我”是一个道理，谁说出来就表示谁。