### Eclipse

是Java实现的智能编辑工具

开源、免费、绿色、灵活（基于插件---plugin）

开普勒 -> -> Mars -> Neon -> Oxygen

Intelli J – IDEA --- 收费，版本控制、依赖管理、重构

复习：

1. final
2. 常量：定义好之后值不可改变

B. 最终方法：可以重载、可以继承、不能重写

C. 最终类：不能被继承

2. abstract

A. 抽象类：不能实例化，有构造方法。不一定有抽象方法。不能用final修饰。子类在继承抽象类之后要实现抽象类中的所有的抽象方法

B. 抽象方法：可以重载，必然重写。不能用static/final/private修饰

3. 接口

A. 用interface定义接口，用implements让类实现一个接口

B. Java中支持类和接口的多实现，支持接口与接口之间的多继承

C. 接口中都是抽象方法，默认用public abstract修饰

D. 接口中的属性默认是用public static final修饰

## 内部类

### 方法内部类

定义在方法中类。

只能在定义它的方法中使用。方法内部类可以使用外部类中的一切属性和方法，可以使用当前方法中的常量。方法内部类中可以定义普通的属性和方法，也可以定义静态常量，但是不能定义静态变量和静态方法。方法内部类可以用final/abstract修饰，可以继承别的类或者实现别的接口

如果某个方法中的某一段逻辑要被重复使用又不想让别的方法调用，此时可以将这段逻辑封装到一个方法内部类中

### 成员内部类

定义在类内方法外的类

成员内部类可以使用外部类的一切属性和方法。可以定义普通的属性和方法，也可以定义静态常量，但是不能定义静态变量和静态方法。

Outer.Inner oi = new Outer().new Inner();

### 静态内部类

用static修饰的内部类

不能使用外部类中的非静态方法和非静态属性，可以使用外部类中的静态属性和静态方法。静态内部类中可以定义一切的属性和方法。

Outer.Inner oi = new Outer.Inner();

总结：

1. 除了静态内部类以外，其余的内部类中都不允许定义静态变量和静态方法

2. 除了静态内部类以外，其余的内部类都可以使用外部类中的非静态属性和非静态方法

### 匿名内部类

匿名内部类本质上是继承了对应的类或者是实现了对应的接口

只要一个类可以被继承，就可以存在匿名内部类形式

任何一个接口都可以产生匿名内部类

扩展：内部接口

类中可以定义类和接口，接口中可以定义类和接口

类中定义的接口以及接口中定义的类和接口都默认是static修饰

## 包

用package来定义包 --- 一个Java文件中只能有一个package语句，而且唯一的package必须放在Java文件的第一行

用import来导包

\* 表示导入当前包下而不包含子包下的类

java --- jdk的原生包

javax --- 是Java的扩展包

org --- 第三方厂商提供的扩展包

java.lang --- 核心包，包含了基本的类---在程序启动的时候，java.lang包下的类都会自动加载到内存中---在使用的时候不必须要手动导包

java.util --- 工具包

java.applet --- 图形界面

java.math --- 数学运算

java.io --- 数据传输

java.net --- 网络传输

java.nio --- 高并发

java.text --- 格式化

## 垃圾分代回收机制

针对的堆内存。

对象在创建完成之后是存储在堆内存中---由于在Java中每种类型的大小是固定的，所以所有的内存分配是由JVM自动完成的而不是由程序员来手动分配的。所以Java程序员在书写代码的时候不需要考虑内存的分配和回收问题。---导致如果内存出现问题，Java程序员是无法处理的。

对象使用完成之后不一定是立即回收的。GC（Garbage Collector）负责解析无用对象。GC是在不定的某个时刻启动。

堆内存分为了新生代和老生代。 新生代分为了伊甸园区和幸存区。对象刚创建的时候是放到了伊甸园区，在伊甸园区经过一次回收如果依然存在则挪到幸存区。如果在幸存区中经过了多次回收则会放到老生代。老生代的扫描频率要远远低于新生代。---老生代的对象如果被收回可能会导致程序的卡顿甚至崩溃。

发生在新生代的回收---初代回收(minor gc)

发生在老生代的回收---完全回收(full gc)

### 类加载机制(了解)

加载（将class文件转化为字节码放到内存中）-校验（检查字节码的合法性以及字节码和JVM的兼容性）---准备（为所有的类变量来分配空间并标记一个默认值）---解析（常量的符号引用替换）---初始化（为静态变量赋值并执行静态代码块）

类加载器---ClassLoader

启动/引导类加载器---Bootstrap ClassLoader--加载核心类

扩展类加载器 --- Extension ClassLoader

应用/系统类加载器 --- Application/System ClassLoader

类加载遵循双亲委派机制

API --- Application Programming Interfaces --- 应用程序接口 --- 实际上是Java提供的一系列的接口以及对应的实现类

Object String Pattern 包装类 数学类 日期类 3

异常 集合 映射 3.5

File IO 线程 套接字 --- 4.5

反射、注解、（动态代理/工厂模式） --- 1

JDK8特性 JVM参数 --- 1

## Object

顶级父类。所有类的对象都可以使用Object对象来接住。是Java中唯一的一个没有父类的类。

Object o = “abc”;

如果一个类在定义好之后没有指明父类，那么默认继承Object

### 重要方法

clone()---克隆一个新的对象。如果这个对象要想被克隆，那么这个对象对应的实际的类要实现Cloneable---这个接口中没有任何的属性和方法，仅仅用于标志这个类产生的对象可以被克隆。

finalize()---建议gc回收垃圾对象---System.gc();

getClass()---获取对象的实际类型

Bin---二进制

Oct---八进制

Dec---十进制

Hex---十六进制

重写equals：---要求掌握2-3个属性的重写

1. 判断地址是否一致

2. 判断参数是否为null

3. 判断对象和参数的类型是否一致

4. 判断属性是否一致

练习：重写Student类的equals方法

class Student {

private String name;

private String no;

}

public class Test {

public static void main(String[] args){

int a = 5;

int b = 7;

swap(a, b);

System.out.println(a);

System.out.println(b);

Person p1 = new Person("Amy");

Person p2 = new Person("Sam");

swap(p1, p2);

System.out.println(p1.name); // Sam

System.out.println(p2.name); // Amy

}

public static void swap(Person p1, Person p2){

Person temp = p1;

p1 = p2;

p2 = temp;

p1.name = "Amy";

p2.name = "Sam";

}

public static void swap(int i, int j){

int temp = i;

i = j;

j = temp;

}

}