1.创建类

1）加载类：执行静态代码块

2）初始化

①初始化成员变量

②执行代码块

3）创建类：构造函数

例子：

public class Test {

public Test() {

System.out.println("构造函数...");

}

static {

System.out.println("静态代码块...");

}

{

System.out.println("代码块...");

}

public static void main(String[] args) {

new T();

}

}

class T extends Test{

{

System.out.println("T代码块...");

}

static {

System.out.println("T静态代码块...");

}

public T() {

System.out.println("T构造函数...");

}

}

Test静态代码块...

T静态代码块...

Test代码块...

Test构造函数...

T代码块...

T构造函数...

类加载：父类先加载，子类后加载（静态成员、静态代码块）

若无父类信息，何来子类继承之说

将两个类的字节码文件加载至内存中才可以使用此类

对象初始化：在构造方法中，执行第一句代码之前（super(...)除外），会初始化

成员变量及执行非静态代码块（成员初始化、非静态代码块）

创建对象：执行构造方法

2.基本类型与包装类型比较时，会自动拆箱为基本类型比较

包装类型之间的比较时，不会自动拆箱

3.基本类型与引用类型作为参数传递时，都是副本传递

基本类型：值的副本

引用类型：引用地址的副本

4.Java中，e-10 => e的-10次幂

e：10

5.Java中基本类型都可以相互转换

除了boolean值，他不允许与任意类型进行转换

6.窄化转换：大类型转小类型（父转子）

扩展转换：小类型转大类型（子转父）

P55 Thinking In Java（Version Four）

7.Java所有方法都有返回值，只不过返回值为void时，return隐式存在

P69 Thinking In Java（Version Four）

1. 循环嵌套时break和continue时考虑使用标签（标号）
2. 一旦GC准备好释放对象占用的内存时，就会调用其finalize()方法，然后在下一次垃圾

回收动作发生之前，才会真正回收对象占用内存。

P87 Thinking In Java

1. 创建对象时初始化成员是在构造方法前还是后

考虑带参构造即可

若之后：那么在构造方法执行完后才初始化那么成员变量为空/0等值；

所以只能是在之前

1. 创建子类时，该对象包含一个基类的子对象，它跟直接用基类创建对象是一样的
2. Java SE5引入了可变参数
3. 一个.java文件就是一个编译单元
4. final参数主要向匿名内部类传递数据
5. 动态绑定/后期绑定：即在程序运行时可判断出对象的真实类型

除了static/final方法外，其他所有方法都是动态绑定

P151 Thinking In Java

1. 构造器实际上是static方法，只不过该static声明是隐式的

P157 Thinking In Java

17.执行清理动作时，与创建顺序相反

P161 Thinking In Java

18.用继承表达行为上的差异，用字段表达状态上的差异

P165 Thinking In Java

19.