来源：小天使笔记本防盗软件官网（www.angeletsoft.cn）

第 1 章　Java 语言概述与面向对象思想　1
1.1 Java 语言的发展　1
1.1.1 Java 语言的产生　1
1.1.2 Java 语言的发展　1
1.2 Java 语言的特点　2
1.3 面向对象与面向过程的差异　3
1.3.1 面向过程思想回顾　4
1.3.2 面向对象思想介绍　4
1.4 面向对象程序设计中的主要概念和特征　4
1.4.1 主要概念　5
1.4.2 主要特征　5
*1.5 Java 与 C++的差异　5
1.6 本章小结 5
习题　5
第 2 章　Java 语言开发环境　6
2.1 JDK 6
2.1.1 JDK 的简介 6
2.1.2 JDK 的构成 6
2.1.3 JDK 的使用 6
2.2 IDE 8
2.2.1 IDE 简介　8
2.2.2 JBuilder　9
2.2.3 Eclipse　9
2.2.4 相关资源　9
2.3 Project 管理 9
2.3.1 Project 的含义 9
2.3.2 可行的 Project 组织模式　9
2.3.3 主要开发工具的 Project 目录　10
2.4 本章小结　10
习题　10

第 1 章　Java 语言概述与面向对象思想

1.1 Java 语言的发展

1.1.1 Java 语言的产生

上世纪 90 年代初期，Sun 公司在研究一种适用于未来的智能设备的编程语言，该语言要具有一些新的特性，以避免 C++的一些不足。

该语言起初命名为 Oak，来源于语言作者 Gosling 办公室窗外的一棵橡树（Oak）。后来在注册时候遇到了冲突，于是就从手中的热咖啡联想到了印度尼西亚一个盛产咖啡的岛屿，中文名叫爪哇，Java 语言得名于此。

随着 Internet 的迅速发展,Web 应用日益广泛,Java 语言也得到了迅速发展。1994 年,Gosling 用 Java 开发了一个实时性较高、可靠、安全、有交互功能的新型 Web 浏览器，它不依赖于任何硬件平台和软件平台。这种浏览器名称为 HotJava，并于 1995 年同 Java 语言一起，正式

在业界对外发表，引起了巨大的轰动，Java 的地位随之而得到肯定。此后的发展非常迅速。Java 编程语言的句法与 C++的句法相似，语义则与 Small Talk TM 的语义相似。Java 编程语言可被用来创建任何常规编程语言所能创建的应用程序。

设计 Java 编程语言的主要目标是:

提供一种易于编程的语言，从而消除其它语言在诸如指针运算和存储器管理方面影响健壮性的缺陷。

利用面向对象的概念使程序真正地成为完全面向对象的程序。

为使代码尽可能清晰合理、简明流畅提供了一种方法。

为获得如下两点益处提供一种解释环境：

提高开发速度——消除编译—链接—装载—测试周期；

代码可移植性——使操作系统能为运行环境做系统级调用。

为运行不止一个活动线程的程序提供了一种方式。

通过允许下载代码模块，从而当程序运行时也能动态支持程序改变。

为那些保证安全性而装载的代码模块提供了一种检查方法。

精心开发的 Java 核心技术为上述目标的实现提供了保证，其中包括如下几个主要技术：

Java 虚拟机

自动垃圾收集

代码安全性

1.1.2 Java 语言的发展

Java 语言的发展目标并不仅仅是一种编程语言，同时还要构建一种开发环境、一种应用环境、一种部署环境。

作为 Java 语言的最基本支持，Sun 公司在 1996 年发布了 Java 开发工具包 JDK 1.0（JDK 是 Java Develop Kit 的简称），其中包括了进行 Java 开发所需要的各种实用程序（编译、执行、文档生成器等等）、基本类库（相当于 C 语言的函数库以及 C++的类库）、程序实例等等。1998 年，Sun 公司发布了更新的 JDK 1.2，由于在技术思想方面与以前有很多改进，所以此后的 Java 技术一般称之为 Java 2。随后，针对不同的领域特征，Java 技术分为三种不同的平台（最新的称谓又去掉了意义模糊的 2，如 JavaSE）：

J2SE——标准 Java 平台

J2SE 是 Java 语言的标准版，指的就是 JDK（1.2 及其以后版本），包含 Java 基础类库和语法。它用于开发具有丰富的 GUI（图形用户界面）、复杂逻辑和高性能的桌面应用程序。

J2EE——企业级 Java 平台

J2EE 建立在 J2SE 之上，用于开发和实施企业级应用程序。它是一个标准的多层体系结构，可以将企业级应用程序划分为客户层、表示层、业务层和数据层，主要用于开发和部署分布式、基于组件、安全可靠、可伸缩和易于管理的企业级应用程序。

J2ME——嵌入式 Java 技术平台

J2ME 也是建立在 J2SE 之上，主要用于开发具有有限的连接、内存和用户界面能力的设备应用程序。例如移动电话（手机）、PDA（电子商务）、能够接入电缆服务的机顶盒或者各种终端和其他消费电子产品。

任何语言建立的应用程序的类型或多或少都与应用程序的运行环境有关，而 Java 语言一般可以建立如下的两种程序：

Applications

Applications 是一种独立的程序，它是一种典型的通用程序，可运行于任何具备 Java 运行环境的设备中。www.angeletsoft.cn

Applets

Applets 是一种贮存于 WWW 服务器上的用 Java 编程语言编写的程序，它通常由浏览器下载到客户系统中，并通过浏览器运行。Applets 通常较小，以减少下载时间，它由超文本标识语言（HTML）的 Web 页来调用。

Java 运行环境具有一些特殊性，或者有很多特殊的人为建立的运行环境，所以 Java 编程中经常建立各种组件，它们可以在特定环境中运行，如 Servlet、JavaBean、JSP 等。

在 Java 技术体系中，有很多免费或非免费的第三方 Java 组件，他们往往提供了某一方面的解决方案，可以应用在很多项目的开发过程中。

学习 Java 语言需要逐渐了解 Java 技术体系，从局部细节开始学习，从整体中进行认识并选择方向，从应用中进行巩固提高。

1.2 Java 语言的特点

Java 语言适用于 Internet 环境，是一种被广泛使用的网络编程语言，它具有如下的一些特点：

简单

Java 语言的语法规则和 C++类似，但 Java 语言取消了指针和多重继承，统一使用引用来指示对象（C++中有两种形式，实际上是两种产生对象的途径，而 Java 中只有一种），通过自动垃圾收集免去了程序设计人员对于内存块的释放工作。

面向对象（近于完全）

Java 语言为了提高效率，定义了几个基本的数据类型以非类的方式实现，余下的所有数据类型都以类的形式进行封装,程序系统的构成单位也是类。因而几乎可以认为是完全面向对象。

平台无关性（可移植、跨平台）

Java 虚拟机（JVM）是在各种体系结构真实机器中用软件模拟实现的一种想象机器，必要时候可以用硬件实现。

当然，这些虚拟机内部实现各异，但其功能是一致的——执行统一的 Java 虚拟机指令。

Java 编译器将 Java 应用程序的源代码文件（.java）翻译成 Java 字节码文件（.class），它是由 Java 虚拟机指令构成的。由于是虚拟机器，因而 Java 虚拟机执行 Java 程序的过程一般称为解释。

依赖于虚拟机技术，Java 语言具有与机器体系结构无关的特性，即 Java 程序一旦编写好之后，不需进行修改就可以移植到任何一台体系结构不同的机器上。

从操作系统的角度看，执行一次 Java 程序的过程就是执行一次 Java 虚拟机进程的过程。

面向网络编程

Java 语言产生之初就面向网络，在 JDK 中包括了支持 TCP/IP、HTTP 和 FTP 等协议的类库。

多线程支持

多线程是程序同时执行多个任务的一种功能。多线程机制能够使应用程序并行执行多项任务，其同步机制保证了各线程对共享数据的正确操作。

良好的代码安全性

运行时（Runtime）一词强调以动态的角度看程序，研究程序运行时候的动态变化，也用运行时环境一词表达类似的含义。

Java 技术的很多工作是在运行时完成的，如加强代码安全性的校验操作。

一般地，Java 技术的运行环境执行如下三大任务：

加载代码——由类加载器执行

类加载器为程序的执行加载所需要的全部类（尽可能而未必同时）。

校验代码——由字节码校验器执行

Java 代码在实际运行之前要经过几次测试。字节码校验器对程序代码进行四遍校验，这可以保证代码符合 JVM 规范并且不破坏系统的完整性。如－－检查伪造指针、违反对象访问权限

或试图改变对象类型的非法代码。

执行代码——由运行时的解释器执行

自动垃圾收集

许多编程语言都允许在程序运行时动态分配内存块，分配内存块的过程由于语言句法不同而有所变化，但总是要返回指向存储区起始位置的指针。

在 C，C++ 及其它一些语言中，程序员负责取消分配内存块。有时这是一件很困难的事情。因为程序员并不总是事先知道内存块应在何时被释放。当在系统中没有能够被分配的内存块时，可导致程序瘫痪，这种程序被称作具有内存漏洞。

当分配内存块不再需要时，程序或运行环境应取消分配内存块。

垃圾收集就是将不再需要的已分配内存块回收。

在其它一般的语言中，取消分配是程序员的责任。

Java 编程语言提供了一种系统级线程以跟踪存储区分配，来完成垃圾收集：

可检查和释放不再需要的存储块

可自动完成上述工作

可在 JVM 实现周期中，产生意想不到的变化

良好的代码健壮性

Java 能够检查程序在编译和运行时的错误。类型检查能帮助用户检查出许多在开发早期出现的错误。同时很多集成开发工具（IDE）的出现使编译和运行 Java 程序更加容易，并且很多集成开发工具（如 Eclipse）都是免费的。

1.3 面向对象与面向过程的差异

一种程序设计语言的产生，不仅是程序设计技术的改进，也包含了表达和认知思想的进步。以 C 语言为代表的部分早期语言，被称为面向过程的语言，不仅因为其程序设计的表达形式是以过程为基本元素，本质上更在于此时对计算机化的系统的理解的主导思想还是控制流或者数据流的，构成系统的要素是模块——处理逻辑。

面向对象语言的产生，是因为对于系统的理解或抽象到了更为高级的层次。此时的认知思想不仅更接近于现实世界，其抽象程度也很高。因而，既有易懂的一方面，也有难懂的另一方面，就看理解的境界了。

1.3.2 面向对象思想介绍

面向对象思想，对现实世界采用直观的理解，计算机化时候采用深度的抽象，简单地可以总结如下：

系统是由事物构成的，事物之间是有联系的，复杂的事物也是系统；

系统与系统、系统与事物、事物与事物之间是有明确界限（边界）的；

系统或事物的状态刻画可以用属性表示,属性一般是些简单的数据,如果复杂那就是事物了；

系统或事物的状态会发生变化，称为行为，产生变化是有原因的（内部的或外部的），变化的过程可能是复杂的；

不同的事物之间会具有共同的属性和行为，共同的极端情形就是完全包含。

基于以上的认识，一个运行时（动态）的具体系统或事物，是由几个更小的具体事物构成（极端的事物就是一个简单的属性数据），它们是不断发生变化的。如果对事物这一概念进行了有效的抽象，那么问题就迎刃而解。

首先，将任何一个具体的事物称为对象（Object）,它的极端情形就是过去的变量；事物是分类的，每一类事物都具有统一的属性和行为，即类型——抽象数据类型，简称为类（Class）；行为既然是过程，那么就抽象成函数，命名为方法，以示区别。

例如：

描述身高或姓名，各自只是一个简单的数据变量；
描述一个学生，可以使用学号、姓名、宿舍、班级等；那更换宿舍算什么呢！
那描述宿舍、班级，又要有许多个项目。

1.4 面向对象程序设计中的主要概念和特征
面向对象程序设计（Object Oriented Programming，OOP）语言中，为了进行更为高度的抽象，会引入一些现实世界中难于找到的概念，但对于一个程序语言来说确实很有价值的。以下暂时介绍的概念基本上都来源于对现实世界的抽象，要从程序设计的角色中去理解它们。面向对象程序设计使系统更易于理解，也使代码具有更好的重用性、可扩展性、易于管理和维护。
1.4.1 主要概念
1、类
类是对一类事物的抽象表示，其角色就相当于数据类型，当然可以算作复杂的数据类型。如学生、宿舍、班级。
2、对象
对象表示一个具体的事物，其角色就是变量，即一个复杂数据类型——xx 类的变量。如周瑜、张飞、瑜飞居，飞虎班。
1.4.2 主要特征
OOP 语言有三个特征：封装、继承及多态性。
1、封装
类的构成包括成员变量/对象与成员方法，这样将相关的数据与函数包装在一起，同其他的类相区分，就是封装。显然，避免了面向过程语言的平行缺陷，说明了类和成员之间的所属关系。进一步地，可以限制类的成员在外部的可见性，那么就将封装体现得更完美。
2、继承
当一种事物甲完全是另一种事物乙的特例，那么，一般地，类甲只是比类乙多出一些成员变量/对象与成员方法，称为类甲继承类乙，类甲称为（类乙的）子类，类乙称为（类甲的）父类。
父类也称为基类、超类，子类也称为导出类、派生类。
显然编写子类就没有必要重复书写父类中乙有的代码部分，这是 OOP 中最典型的代码重用。
3、多态
多态表示一个类的某种行为存在多种实现版本。简单的情况是在一个类中，给出多种不同的实现，复杂的情况是在多个子类中各自给出不同的实现。
*1.5 Java 与 C++的差异
1.6 本章小结
习题
1．解释 Java 语言的三种平台。
2．解释支撑 Java 语言目标的三种主要技术（自查资料完善）。
3．关于面向对象与面向过程的不同，从问题理解和问题表示两种角度，按照个人的理解和认识，例示其不同的地方。

第 2 章　Java 语言开发环境
2.1 JDK
2.1.1 JDK 的简介

可以从 Sun 公司的官方网站（http://java.sun.com）上下载 Java 开发工具包（JDK）。不必要求最新的版本，注意 Java SE、Windows 平台等信息。如：
jdk-1_5_0_17-windows-i586-p.exe
2.1.2 JDK 的构成

2.1.3 JDK 的使用
1、安装
过程简单，基本使用默认设置，注意安装目录（最好简单化）。
2、设置环境变量
环境变量——应用程序运行时候需要的一些相对固定值的参数。
例如，Java 开发工具等软件需要使用 JDK，那么必须知道 JDK 在系统中的什么位置，于是大家约定在操作系统中定义一个名称为 JAVA_HOME 的环境变量，其内容表示 JDK 的安装目录。
在 win2000/2003 中的方法：操作【我的电脑】—【属性】—【高级】—【环境变量】后，在【系统变量】区域进行新建、编辑等操作即可。
JAVA_HOME——必设的环境变量，表示 JDK 安装目录（如 C:\Java\jdk1.5.0_17）。（第一次是新建）
Path——名称程序查找路径。如果需要在命令行方式下使用 Java 的话，修改其内容，在前面增加 JDK 命令目录，不同项目中间使用分号分隔。（修改，增加%JAVA_HOME%\bin）
CLASSPATH——类库设置。（新建，.; C:\Java\jdk1.5.0_17 \lib\tools.jar）。
3、使用
创建程序
可以使用最简单的文本编辑器，编写如下的程序，保存的名字必须是 Hello.java。

```
package chap01;   //当前类所属包名称

public class Hello  //当前类名
  {
   public static void main(String[] args) //应用程序的主函数
     {
      System.out.println("Hello, 我的第一个 Java 应用程序！");  //输出函数
      }
   }
```

程序内容解释
在以上的程序中，必须了解并逐渐熟悉的关键点：
声明包名——chap01，指明类的位置（所属包）、相当于文件夹名称，用途在于方便类的组织和管理；
声明类名——Hello，与程序文件名相同（指必须的 public 类）；
类属性——public，描述类的可见性，即类和其它类、包的关系；
主函数——main，程序入口（仅 Applications 类型需要）；
主函数属性——public static，公有、静态（特殊的函数）；
主函数的参数——String[] args，命令行参数（如果执行程序时候给了参数，将会由系统封装成字符串数组传递到这里，程序内部可以从此获得并处理）；
输出函数——这是 Java 类库中提供的一个在标准输出设备（显示器、字符输出，对于操作系统来说是可以重定向的设备）显示文本的函数。注意其中出现的类名称 System、成员对象名称 out、成员方法名称 println，书写语法以及参数。从帮助中可以方便地查找到函数的

参数语法；

整个类的框架——括号等表达的结构。

注释格式——//表示行注释，本行后面的内容为注释；/**/表示块注释，其中的内容为注释。

以上的内容在一个 Java 应用程序中几乎是必需的。

典型源文件布局

一个 Java 源文件可包含三个“顶级”要素：

一个包声明（可选，没有表示顶级包，但有点麻烦）；

任意数量的导入语句 import，表示将要使用的外部 Java 类；

类和接口声明。

该三要素必须以上述顺序出现。即，任何导入语句出现在所有类定义之前；如果使用包声明，则包声明必须出现在类和导入语句之前。

如果需要或者说是遵循编程规范，那么在文件开头可以使用如下形式的文档注释：

```
/**
文档信息描述
*/
```

这种形式的注释可以被专门的文档生成工具所处理(由 javadoc 命令生成的 HTML 文件)。

程序编译

如下图中的命令 javac Hello.java。

注意编译后的文件变化。

程序运行

如下图中的命令 java chap01.Hello。

命令窗口与简单命令解释

有多种形式打开如下的命令窗口，注意其中常见的命令的含义以及其成功执行的前提。一条命令一般由 命令动词+空格+参数 构成，并以回车结束。命令动词中如果不包含路径那么将从默认的路径中查找命令程序，命令可能的参数内容以及形式取决于程序本身，如果需要，可以使用诸如 java -? 或 java -help 形式获得。

d:——切换当前分区；

cd——切换当前某分区下的工作文件夹，需要参数；

md——创建新的文件夹，需要参数；

copy——拷贝文件；

……

2.2 IDE

2.2.1 IDE 简介

IDE 的含义是 Java 集成开发环境（Integrated Development Environment，IDE）。

有很多 IDE 可供选择，典型的如 JBuilder、Eclipse 等。

2.2.2 JBuilder

JBuilder 是使用 Java 开发的，支持 J2EE 技术的 Java 集成开发环境。但 JBuilder 不是完全免费的。

JBuilder 的安装程序存放在机器中的 C:\java\JBX 中，两个目录相当于两张光盘，安装时候当出现安装项目列表时候只选第一个即可，安装后的处理按照其中的说明文件进行，解密环节或许需要执行若干次。

最好先安装 JDK。
如果出现解密命令失败，一般是 path 环境变量设置问题。
2.2.3 Eclipse
Eclipse 是基于 Java 的，开放源码的、可扩展的应用开发平台，它为编程人员提供了一流的 Java 集成开发环境（Integrated Development Environment，IDE）。是一个可以用于构建集成 Web 和应用程序的开发工具平台，其本身并不会提供大量的功能，而是通过插件来实现程序的快速开发功能。
Eclipse 是一个成熟的可扩展的体系结构。它为创建可扩展的开发环境提供了一个平台。这个平台允许任何人构建与环境或其他工具无缝集成的工具，而工具与 Eclipse 无缝集成的关键是插件。Eclipse 还包括插件开发环境（PDE），PDE 主要针对那些希望扩展 Eclipse 的编程人员而设定的。这也正是 Eclipse 最具魅力的地方。通过不断的集成各种插件，Eclipse 的功能也在不断的扩展，以便支持各种不同的应用。
虽然 Eclipse 是针对 Java 语言而设计开发的，但是它的用途并不局限于 Java 语言，通过安装不同的插件 Eclipse 还可以支持诸如 C/C++、PHP、COBOL 等编程语言。
Eclipse 利用 Java 语言写成，所以 Eclipse 可以支持跨平台操作，但是需要 SWT（Standard Widget Toolkit）的支持，不过这已经不是什么大问题了，因为 SWT 已经被移植到许多常见的平台上，例如 Windows、Linux、Solaris 等多个操作系统，甚至可以应用到手机或者 PDA 程序开发中。
2.2.4 相关资源
需要关注各家公司的官网、以及对应中文网站。
还有如 www.csdn.net 这样的综合性技术网站。
2.3 Project 管理
2.3.1 Project 的含义
Project 即项目或工程，几乎所有的 IDE，使用时候必须先建立 Project，才可以创建程序、编译程序、运行程序、修改程序、调试程序。
如果有一种非常理想和完善的 IDE 工具，那么它可以管理一个 Project 的所有文档资源和生命周期。最好是从软件工程的角度去思考和理解 Project。
2.3.2 可行的 Project 组织模式
学习程序设计语言，首先要学会管理自己的代码资源，使用合理的组织方式。
影响实际项目开发中的工程组织方式有多种因素：软件生命周期的不同阶段、项目的功能结构、项目关联人员角色、文档分类等。
在本书的学习中，建议每章组织为一个工程，章内所有例子、习题以及自己创建的程序能够分门别类（文件夹=包）进行组织。
平时管理好自己的劳动成果，要有代码资源的积累。
2.3.3 主要开发工具的 Project 目录
2.4 本章小结
习题
1. 建立一个总结文档，分章节记录以后学习实践过程中的一些关键的操作过程和技巧、问题的解决方法、总结等。以便以后查阅。
2. 练习 JDK 的安装、编译并执行 Hello 或自己编写的应用程序。
3. 改造 Hello 程序，自定义三个整型变量 a、b、c 并赋值，然后适当地处理保证 a 最大、c 最小，再按从大到小顺序输出。
编程时候主函数的代码中，除了输出函数之外，其它可以按照 c 语言的语法进行。
4. 安装 JBuilder，尝试创建工程、建立程序、编译、运行、修改、调试。

此过程中，关注其中的少量主菜单（文件、工程、运行）中的主要功能。
4. 查阅 JBuilder 帮助，通过 System 类最终找到输出函数 print 系列适当阅读。