HZ BOOKS
华章教育

普通高等教育“十一五”国家级规划教材

计算机基础课程系列教材

Visual Basic 程序设计教程

第4版

郭志强 邱李华 曹青 等编著

机械工业出版社
China Machine Press

普通高等教育"十一五"国家级规划教材

计算机基础课程系列教材

Visual Basic 程序设计教程

第4版

郭志强 邱李华 曹青 等编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Visual Basic 程序设计教程 / 郭志强等编著 . —4 版 . —北京：机械工业出版社，2017.1
（计算机基础课程系列教材）

ISBN 978-7-111-55879-8

I. V… II. 郭… III. BASIC 语言 - 程序设计 - 高等学校 - 教材 IV. TP312.8

中国版本图书馆 CIP 数据核字（2017）第 014962 号

本书以 Visual Basic 6.0 为开发环境，深入浅出地介绍了程序设计的基本概念和基础知识、结构化程序的基本结构（顺序、选择和循环结构）、数组、过程、Visual Basic 常用控件、Windows 界面设计要素、计算机图形设计基本概念、文件、数据库基础及应用程序开发等。

本书内容全面，介绍了大学生应该掌握的多种常用算法，并提供了大量实用、有趣的实例，增强学生学习兴趣的同时开阔了学生视野。另外，每章配有大量知识点全面且难易结合的上机练习题，对学生巩固所学知识提供了强有力的帮助。本书配套出版的习题集紧密结合教材编写，包含了大量各种类型的练习题，同时附有参考答案，有利于学生进行课外自主练习和能力培养。

本书可作为高等学校计算机及其相关专业教材、全国计算机等级考试的参考用书，也可作为广大计算机爱好者的入门自学读物。

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：佘　洁　　责任校对：董纪丽
印　　刷：北京市荣盛彩色印刷有限公司　　版　　次：2017 年 3 月第 4 版第 1 次印刷
开　　本：185mm × 260mm　1/16　　印　　张：22
书　　号：ISBN 978-7-111-55879-8　　定　　价：39.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：(010) 88378991　88361066　　投稿热线：(010) 88379604
购书热线：(010) 68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

前　言

Visual Basic 源自于 BASIC 编程语言，是一种由微软公司开发的可视化程序设计语言。它基于 Windows 开发环境，以事件驱动为机制，采用图形化用户界面（GUI），具有简单、易学、易用的优点，深受程序专业开发人员和初学者的喜爱。

Visual Basic 不但继承了传统的结构化程序设计语言的功能，而且引入了最新的面向对象程序设计思想。随着 Windows 版本的变化，Visual Basic 语言的版本也在逐步升级，它的功能也越来越强大。使用 Visual Basic 既可以编写各种小的客户端程序，或轻松地创建 ActiveX 控件，又可以方便快捷地使用 ADO 连接数据库，创建功能强大的数据库应用程序。

目前 Visual Basic 已经成为许多高等学校首选教学使用程序设计语言，也是全国计算机等级考试指定的程序设计语言之一。

2002 年 1 月，我们出版了《Visual Basic 程序设计教程》及配套习题集。

2006 年 9 月，教育部高等学校计算机科学与技术教学指导委员会正式制定了《关于进一步加强高等学校计算机基础教学的意见暨计算机基础课程教学基本要求（试行）》（以下简称《要求》），对计算机程序设计基础课程教学提出了“一般要求”和“较高要求”。在充分领会《要求》精神的基础上，我们对原教材进行了修订，形成了第 2、3 版。第 2、3 版教材涵盖了《要求》中有关 Visual Basic 程序设计的“一般要求”和“较高要求”涉及的所有内容，为不同办学层次的学校和不同专业提供了选择余地。第 2、3 版突出了教改特色，适应了各高校计算机课程改革的新要求和新动向，被许多高等学校选为教材，深受广大师生的喜爱，是普通高等教育“十一五”国家级规划教材。

本书为《Visual Basic 程序设计教程》第 4 版。第 4 版秉承了前面版本的特点，注重对学生基本概念、基本理论、基本技能的培养，条理清晰，深入浅出，实例丰富。同时，结合一线教师多年在教学实践过程中遇到的问题和其他高校教师反馈的意见，对第 3 版进行了修订，主要体现在以下几个方面：

1）强化了面向对象程序设计的基本概念。面向对象程序设计方法在当今应用程序的创建中用得越来越多，学生有必要对面向对象程序设计的基本概念、架构和设计方法有一个较全面的了解。

2）完善了数据库基本概念和相关知识介绍。数据库在各种信息系统中得到了广泛的应用，为了让学生快速掌握数据库应用程序的设计方法，本版加大了 SQL 的描述比重，引入了 ADO 对象的介绍，并通过实例深入浅出地介绍了数据库设计和应用程序的开发过程。

3）所有的例题和练习题在最新的 Windows 10 环境下进行了测试，做到了完美的兼容。

4）更正了以前版本中错误和不适当的概念描述。

5）在例题中增加了更多的注释语句，方便学生理解程序。

6）对较难的上机练习题，增加了更多提示，减轻了学生的困惑。

7）文字描述更加简练，易读易用，即使对于初学者，阅读起来也比较容易。

8）例题和习题更加丰富，增加了更多具有实用性和趣味性的例题和上机练习题。

9）完善了部分上机练习题的视频演示，视频以 swf 文件形式给出（通过华章网站 www.hzbook.com 下载）。

编　者

教学建议

内容	教学要求	讲课	上机	小计
第 1 章 程序设计基础	了解程序设计语言的编辑、解释、编译、连接的概念 了解源程序、目标程序和可执行程序的概念 了解算法的概念及表示 了解结构化程序设计的三种基本结构及面向对象程序设计基本概念	2		2
第 2 章 Visual Basic 简介	了解 Visual Basic 简单工程结构和工程文件的管理 了解可视化编程的基本概念 掌握属性的设置，以及方法、事件过程的使用 掌握窗体、命令按钮、标签、文本框等常用控件的使用	4	4	8
第 3 章 Visual Basic 程序设计代码基础	了解 Visual Basic 的基本数据类型 掌握常量与变量的概念及定义方法 了解常用内部函数的使用 掌握运算符、表达式的书写及求值规则 了解代码的书写规则及语法格式的约定	4	4	8
第 4 章 顺序结构程序设计	掌握赋值语句的使用方法 掌握数据输入、输出的基本方法	3	4	7
第 5 章 选择结构程序设计	掌握单行结构条件语句、块结构条件语句、多分支选择语句的语法结构和使用方法 掌握选择结构嵌套的使用	3	4	7
第 6 章 循环结构程序设计	掌握 For…Next 循环结构、Do…Loop 循环结构的基本语法和使用方法	4	6	10
第 7 章 数组	掌握数组的基本概念，以及静态数组、动态数组的定义和使用 掌握数组的基本操作 了解控件数组的概念、创建及简单使用	4	4	8
第 8 章 过程	了解过程的概念和分类 掌握 Function 过程、Sub 过程的定义和调用方法 掌握过程调用中的参数传递原理和方法 掌握过程的嵌套调用 了解变量和过程的作用域与变量的生存期	6	4	10
第 9 章 Visual Basic 常用控件	了解 Visual Basic 常用控件的使用	4	4	8
第 10 章 界面设计	掌握下拉式菜单和弹出式菜单的设计方法 掌握工具栏的设计方法 掌握对话框的设计方法	2	2	4
第 11 章 图形设计	了解 Visual Basic 图形设计的基本概念和方法 了解 Visual Basic 常用图形控件的使用 了解画点、画线、画圆等基本绘图方法	2	2	4

（续）

内容	教学要求	讲课	上机	小计
第 12 章 文件	了解文件结构与文件的分类 掌握对顺序文件、随机文件、二进制文件的打开、关闭以及读写操作	4	4	8
第 13 章 数据库	了解数据库的基本原理和概念 掌握 INSERT、UPDATE、DELETE 语句的基本语法结构和使用方法 掌握 SELECT 语句的基本语法结构和常用子句的使用方法 了解 ADO 控件及对象 掌握使用 ADO 对象连接并操纵数据库的方法	6	6	12

说明：根据教材涵盖的内容，建议课程安排的总学时为 96 学时，其中讲课和上机各占 48 学时，教师可以根据实际教学大纲要求及后续课程的安排进行适当的调整。

目　录

第 1 章
程序设计基础

要使计算机能够按人的要求完成一系列的操作，就需要在人和计算机之间建立一种二者都能识别的特定语言，这种特定语言就是程序设计语言。使用程序设计语言编写的用来使计算机完成某种特定任务的一系列命令的集合构成程序，编写程序的工作则称为程序设计。Visual Basic 是一种程序设计语言。本书将介绍 Visual Basic 6.0 程序设计语言的基础知识，以及如何使用 Visual Basic 6.0 进行简单程序设计。

1.1 程序设计语言

计算机语言种类繁多，可以从应用特点和对客观系统的描述等多个方面对其进一步分类。例如，从应用范围来分，可以分为通用语言与专用语言；从程序设计方法来分，可以分为面向过程与面向对象语言；从程序设计语言与计算机硬件的联系程度分，可以分为机器语言、汇编语言和高级语言。其中，机器语言、汇编语言依赖于计算机硬件，被统称为低级语言，高级语言与计算机硬件基本无关，因此可以说程序设计语言经历了由低级向高级发展的过程。

随着计算机的发展，出现了许许多多的高级程序设计语言。例如，早期出现的 BASIC、Pascal、FORTRAN、C 等高级语言，采用的是面向过程的程序设计方法，而较新出现的 Visual Basic、Visual C++、Java 等，采用的是面向对象的程序设计方法。面向过程的语言致力用计算机能够理解的逻辑来描述需要解决的问题和解决问题的具体方法和步骤；面向对象的程序设计语言将客观世界的事物抽象成对象，通过面向对象的方法更利于用人理解的方式对复杂系统进行分析、设计与编程。同时，面向对象能提高编程的效率，通过封装技术、消息机制可以像搭积木一样快速开发出一个全新的系统。面向对象的语言已经成为当前最流行的一类程序设计语言。Visual Basic 6.0 是一种面向对象的高级程序设计语言。

在所有的程序设计语言中，除了用机器语言编写的源程序可以在计算机上直接执行外，用其他语言编写的源程序都需要使用相应的翻译工具对其进行翻译，才能被计算机所理解并执行，这种语言翻译工具称为语言处理程序或翻译程序，用不同的程序设计语言编写出来的源程序，需要使用不同的语言处理程序。通过语言处理程序翻译后的目标代码称为目标程序。

对高级语言源程序进行翻译可以有两种方式：解释方式和编译方式，相应的翻译工具分别称为解释程序和编译程序。在解释方式下，解释程序对源程序进行逐句分析，如果没有

错误，则将该语句翻译成相应的机器指令并立即执行，如果发现有错误，则立即停止执行。解释方式不生成可执行程序，其工作过程如图 1-1 所示。

图 1-1 解释方式的工作过程

在编译方式下，编译程序对整个源程序进行编译处理，产生等价的目标程序。通常在目标程序中还可能调用一些其他语言编写的程序和标准程序库中的标准子程序，因此需要使用连接程序将目标程序和有关的其他程序库组合成一个完整的可执行程序，产生的可执行程序可以脱离源程序和语言处理程序独立存在，且可以重复运行。编译方式的工作过程如图 1-2 所示。

图 1-2 编译方式的工作过程

1.2 程序设计

程序设计就是使用某种程序设计语言编写一些代码来驱动计算机完成特定功能的过程。为了有效地进行程序设计，至少应当具备两个方面的知识：一个是要掌握一种或一种以上的程序设计语言；另一个是要掌握解题的方法和步骤，也就是说，在遇到一个需要求解的问题后，怎样将它分解成一系列计算机可以实现的操作步骤，这就是“算法”需要研究的问题。可以说，程序设计的灵魂是算法，而语言只是实现算法的工具。有了正确的算法，就可以利用任何一种语言编写程序，使计算机进行工作，得出正确的结果。因此本书在正式介绍 Visual Basic 语言之前，先简要介绍算法的概念和算法的表示。

1.2.1 算法

1. 什么是算法

算法是对解决某一特定问题的操作步骤的具体描述。广义地说，算法就是为解决某个问题而采取的方法和步骤。例如，厨师炒菜的操作步骤就是“烹调算法”；期末考试前的复习计划就是“复习算法”；到医院看病，先挂号，再问诊、检查、诊断，然后取药等，这就是“看病算法”。

在这里我们只讨论计算机算法，计算机中的算法就是为计算机解决问题而设计的有明确意义的操作步骤的有限集合。

计算机算法可分为两大类：数值计算算法和非数值计算算法。数值计算算法的目的是求数值解，如求方程的根、求函数的定积分等；非数值计算算法包括的范围很广，最常见的是用于管理领域，如用于文字处理和图形图像处理以及信息的排序、分类、查找等的算法。

2. 算法的特性

算法应具有以下特性：

- 有穷性：算法中执行的步骤总是有限次数的，不能无休止地执行下去。
- 确定性：算法中的每一步操作必须含义明确，不能有二义性。
- 有效性：算法中的每一步操作都必须是可执行的。
- 有 0 个到若干个输入：算法常需要对数据进行处理，因此，算法常需要数据输入。

- 有 1 个到若干个输出：算法的目的是用来解决一个给定的问题，因此，它应向人们提供解题的结果。

3. 算法的表示形式

描述算法的方法有多种，常用的有自然语言、流程图、NS 图、伪代码和 PAD 图等，其中最普遍的是流程图。下面简要介绍用自然语言和用流程图表示算法。

（1）用自然语言表示算法

自然语言就是人们日常使用的语言，因此用自然语言表示的算法较容易理解。

例如，将两个变量 X 和 Y 的值互换。

交换存在直接交换和间接交换两种方式。例如，两个人交换座位，只要各自去坐对方的座位就行了，这种交换就是直接交换。再如一瓶酒和一瓶醋互换，就不能直接从一个瓶子倒入另一个瓶子，必须借助一个空瓶子，先把酒倒入空瓶子，再把醋倒入已倒空的酒瓶子，最后把酒倒入已倒空的醋瓶子，这样才能实现酒和醋的交换，这种交换就是间接交换。

计算机中交换两个变量的值不能用直接交换的方法，而必须采取间接交换的方法，因此，需要引入一个中间变量 Z。用自然语言表示该算法，可以描述为：

步骤 1：将 X 的值存入中间变量 Z 中：$X \rightarrow Z$。

步骤 2：将 Y 的值存入变量 X 中：$Y \rightarrow X$。

步骤 3：将中间变量 Z 的值存入 Y 中：$Z \rightarrow Y$。

用自然语言表示算法，虽然容易表达，但文字冗长，而且往往不严格，对于同一段文字，不同的人会有不同的理解，容易产生二义性，因此，除了很简单的问题外，一般不用自然语言表示算法。

（2）用流程图表示算法

流程图用一些图框、流程线以及文字说明来描述操作过程。用流程图来表示算法，直观、形象、容易理解。

传统流程图采用了美国国家标准化协会（American National Standard Institute，ANSI）规定的一些符号，常见的流程图符号表示如下：

▭（起止框）：表示流程开始或结束。

▱（输入 / 输出框）：表示输入或输出。

▭（处理框）：表示基本处理功能的描述。

◇（判断框）：根据条件是否满足，在几个可以选择的路径中，选择某一路径。

↓→（流程线）：表示流程的路径和方向。

○（连接点）：表示流程图中向其他地点或来自其他地点的输出或输入。

图 1-3 为交换两个变量的传统流程图。

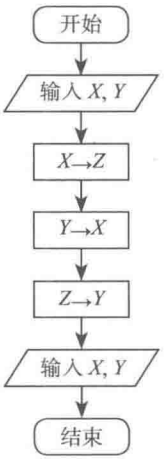

图 1-3 交换两个变量的传统流程图

1.2.2 程序设计的基本结构

在一些高级语言中设置了无条件转移语句，当程序执行到此语句时，就会无条件地转移去执行某条语句。对于编制一些小程序来说，无条件转移语句使用起来很方便，可以转到程序的任意位置去执行，但是，在长期的程序设计实践中人们发现，当设计的程序较大而且无条件转移语句稍多时，就会给程序的阅读、修改、维护带来很多的麻烦。任意地转移会使程

序设计思路显得非常没有条理性且难以理解。于是人们设想，能否使用一些基本的结构来设计程序，无论多么复杂的程序，都可以使用这些基本结构按一定的顺序组合起来。这些基本结构的特点都是只有一个入口、一个出口。由这些基本结构组成的程序就避免了任意转移、难以阅读的问题。

1966 年，Bohra 和 Jacopini 提出了 3 种基本结构，认为算法和程序都可以由这 3 种基本结构组成。这 3 种基本结构分别是顺序结构、选择结构和循环结构。

（1）顺序结构

顺序结构是最简单的一种基本结构，计算机在执行顺序结构的程序时，按语句出现的先后次序依次执行。图 1-4 是用传统流程图表示的顺序结构。图 1-4 的虚线框内是一个顺序结构，其中，A 和 B 表示操作步骤。计算机先执行 A 操作，再执行 B 操作。

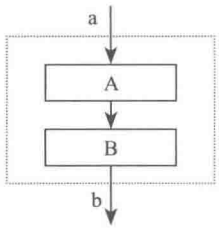

图 1-4 顺序结构流程图

（2）选择结构

当程序在执行过程中需要根据某种条件的成立与否有选择地执行一些操作时，就需要使用选择结构。图 1-5 是用传统流程图表示的选择结构，图 1-5 的虚线框内是一个选择结构。这种结构中包含一个判断框，根据判断给定的条件是否成立，从两个分支路径中选择执行其中的一个。从图 1-5 可以看出，无论执行哪一个分支路径都通过汇合点 b。b 点是该基本结构的出口点。

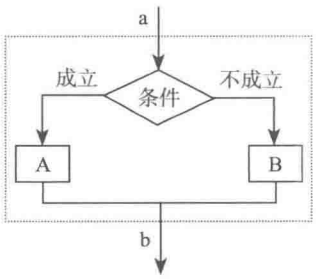

图 1-5 选择结构流程图

（3）循环结构

循环结构用于规定重复执行一些相同或相似的操作。要使计算机能够正确地完成循环操作，就必须使循环能够在执行有限次数后退出，因此，循环的执行要在一定的条件下进行。根据对条件的判断位置不同，可以有两类循环结构：当型循环和直到型循环。

1）当型循环结构。图 1-6 表示了当型循环结构。当型循环的执行过程是：当程序运行到 a 点，从 a 点进入当型循环时，首先判断条件是否成立，如果条件成立，则执行 A 操作，执行完 A 操作后，再判断条件是否成立，若仍然成立，再执行 A 操作。如此反复执行，直到某次条件不成立时为止，这时不再执行 A 操作，而是从 b 点退出循环。显然，在进入当型循环时，如果一开始条件就不成立，则 A 操作一次都不执行。

2）直到型循环结构。图 1-7 表示了直到型循环的结构。直到型循环的执行过程是：当程序运行到 a 点，从 a 点进入直到型循环时，首先执行 A 操作，然后判断条件是否成立，如果条件不成立，则继续执行 A 操作，再判断条件是否成立，若仍然不成立，再执行 A 操作。如此反复执行，直到某次条件成立时为止，这时不再执行 A 操作，而是从 b 点退出循环。显然，在进入直到型循环时，A 操作至少执行一次。

以上 3 种基本结构的共同特点如下：

- 只有一个入口、一个出口。
- 每一个基本结构中的每一部分都有机会被执行到。也就是说，对每一个框来说，都应当有一条从入口到出口的路径通过它。
- 结构内不存在“死循环”（即无终止的循环）。

已经证明，由以上 3 种基本结构组成的算法，可以解决任何复杂的问题，并且由基本结

构构成的算法属于“结构化”的算法，不存在无规律的转移。

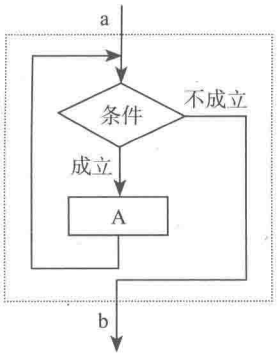

图 1-6 当型循环结构

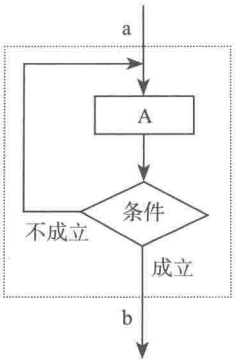

图 1-7 直到型循环结构

1.3 结构化程序设计

结构化程序设计是软件发展的一个重要里程碑，它采用自顶向下、逐步求精及模块化的设计思想，将一个系统问题按功能划分为若干模块，并使各模块之间的关系尽可能简单、在功能上相对独立，每一模块内部均是由顺序、选择和循环 3 种基本结构组成，而模块功能的实现是由子程序（函数）来完成的。

虽然结构化程序设计方法具有很多优点，但它仍是一种面向过程的程序设计方法，它把数据和处理数据的过程分离为相互独立的实体，一般只突出了实现功能的操作方法及过程的实现，其中心思想是用计算机能够理解的逻辑来描述和表达待解决的问题及其具体的解决流程，而被操作的数据处于实现功能的从属地位，程序模块和数据结构是松散地耦合在一起的。使用这种方法编写的程序在执行时是线性的，顺序是不能改变的。如果在执行过程中，用户需要输入什么参数或用户做出选择，程序将等待用户的输入。只有用户提供足够的数据，程序才能继续执行下去。像早期出现的 BASIC、FORTRAN、Pascal、C 等都是进行结构化程序设计的语言。

使用结构化程序设计方法编写小的程序比较有效。然而，随着程序规模与复杂性的增长，程序中的数据结构变得越来越重要。当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于“老问题”的新方法都要重新编写处理过程，带来了额外的开销，致使程序的可重用性变差。

由于上述缺陷，面向过程的结构化程序设计思想已不能完全满足现代化软件开发的要求，一种全新的软件开发技术应运而生，这就是面向对象程序设计（Object Oriented Programming，OOP）。

1.4 面向对象程序设计

面向对象程序设计是一种计算机程序设计架构，它按照人们对现实世界的习惯认识和思维方式来设计和组织程序，它强调系统的结构应该直接与现实世界的结构相对应，应该围绕现实世界中的事物来构造系统。因此，面向对象程序设计将现实世界中的任何事物都看作对象，通过在对象之间建立相互关系来解决实际问题。

面向对象程序设计一般会涉及以下基本概念。

1. 对象

现实世界中的每一种实体都是对象（Object）。在面向对象程序设计中，对象是现实世界中各种实体的抽象表示。“对象”中封装了描述该对象的属性（数据）和方法（行为方式），是数据和代码的组合。对象中的属性描述了对象的特征；对象中的方法决定要向哪个对象发消息、发什么消息，以及收到消息时如何进行处理等。整个程序由各种不同类型的对象组成，各对象既是一个独立的实体，又可以通过消息相互作用。

2. 类和类的实例

类（Class）是对具有相同属性和行为的一组对象的抽象。类描述了属于该类的所有对象的属性和方法，也就是说，对象的属性和方法是在定义类时被指定的。

类是生成对象的模板，每一个属于某个类的特定对象称为该类的一个实例（Instance），通常简称为对象。例如，猫科动物是类，每只老虎或猫是该类的一个实例。

Visual Basic 为用户提供了丰富的类，能满足用户绝大多数的需求，如 Button 类、Label 类、TextBox 类等。用户通过这些类的实例（即对象）完成相应的编程工作。在编程时，程序员经常使用工具箱中的控件进行界面设计，每一个控件都对应一个类，每一个具体控件就是对应类的实例，即对象。

用户还可以创建自己的类，为它们定义属性、方法和事件，然后利用自己定义的类创建相应的对象。

3. 封装

对象中包含了描述该对象的属性（数据）和方法（行为方式），这种技术叫做封装(Encapsulation)。对象的这种封装性可以将对象的内部复杂性与应用程序的其他部分隔离开来，这样，在程序中使用一个对象时就不必关心对象的内部是如何实现的，而每一个对象仅有若干接口为应用程序所使用。封装使对象的内部实现与外界应用分割开来，可以有效地防止外界对对象内部数据和代码的破坏，也避免了程序各部分之间数据的滥用。

4. 继承

在面向对象程序设计中，可以在已有类的基础上通过增加新特征而派生出新的类，这种机制称为继承（Inheritance）。其原有的类称为基类（Base Class）或父类，而新建立的类则称为派生类或子类。

例如，生物是一个类，而动物就是生物的一个派生类，猫科动物又是动物的一个派生类。

在继承机制中，可以在基类的基础上增加一些属性和方法来构造出新的类。当定义新的类时，如果将新类说明为某个类的派生类，则该派生类会自动地继承其基类的属性和方法。如果基类的特征发生了改变，则其派生类将继承这些改变的特征。继承性可以使得在一个类上所做的改动，能够自动反映到它的所有派生类中。

通过继承，基类的内容在新类中可以直接使用而不必重新定义，这显然减少了软件开发的工作量，也实现了代码的重用，这正是面向对象程序设计的优点。

5. 多态

多态（Polymorphism）性是面向对象程序设计的另一重要特征。在通过继承而派生出的一系列类中，可能存在一些名称相同但实现过程和功能不同的方法。

多态性有两个方面的含义：一种是将同一个消息发送给同一个对象，但由于消息的参数不同，对象表现出不同的行为，这种多态性是通过“重载”来实现的；另一种是将同一个消息发送给不同的对象，各对象表现出的行为各不相同，这种多态性是通过“重写”来实现的。

例如，Move 方法是多态性的一个很典型的例子。当 Move 方法被一个窗体对象执行时，