高等学校计算机公共基础课规划教材

Visual Basic程序设计教程

诸海生　任　超　编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高等学校计算机公共基础课规划教材

Visual Basic 程序设计教程

诸海生　任　超　编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书是为初学程序设计和对可视化程序设计没有经验的读者编写的，主要内容包括 Visual Basic 程序设计基础、程序控制结构、窗体和常用控件、数组、过程、菜单、通用对话框、多窗体界面、图形设计、文件处理、数据库应用基础等。本书包括了 Visual Basic 程序设计的基本内容，也涵盖了全国计算机等级考试二级 Visual Basic 考试的绝大部分知识点。

本书在结构上考虑了初学者的学习特点，尽量将 Visual Basic 语言成分、界面元素和算法的难度分散。对于复杂一些的例题，则采用问题分析、解题思路、设计算法、界面设计、代码实现等分步讲解的模式，明确各个步骤要解决的问题，帮助读者理清程序设计的过程。为提高读者兴趣，本书还设计了一些涉及简单动画的例题与习题。

本书适合作为高等学校非计算机专业学习 Visual Basic 程序设计的教材，也可供其他对程序设计有兴趣的读者和准备参加全国计算机等级考试二级 Visual Basic 考试的读者复习、参考。

图书在版编目（CIP）数据

Visual Basic 程序设计教程 / 诸海生，任超编著.
—北京：中国铁道出版社，2011.2
高等学校计算机公共基础课规划教材
ISBN 978-7-113-12250-8

Ⅰ. ①V… Ⅱ. ①诸… ②任… Ⅲ. ①BASIC 语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2010）第 231599 号

书　　名：Visual Basic 程序设计教程
作　　者：诸海生　任　超　编著

策划编辑：秦绪好
责任编辑：周海燕　　**编辑部电话**：400-668-0820
编辑助理：贾淑媛
封面设计：付　巍　　**封面制作**：李　路
版式设计：于　洋　　**责任印制**：李　佳

出版发行：中国铁道出版社（北京市宣武区右安门西街 8 号　　邮政编码：100054）
印　　刷：北京新魏印刷厂
版　　次：2011 年 2 月第 1 版　2011 年 2 月第 1 次印刷
开　　本：787mm×1092mm　1/16　印张：19.25　　字数：462 千
印　　数：3 000 册
书　　号：ISBN 978-7-113-12250-8
定　　价：29.00 元

FOREWORD

前　言

Visual Basic 是一门便于学习、便于掌握的具有可视化功能的高级语言，是一个很好的学习高级语言程序设计和可视化程序设计的入门工具。

本书是介绍利用 Visual Basic（简称 VB）进行可视化程序设计的基础教材。本书是按照下面的思路编写的。

1．程序设计方法与界面设计方法并重

考虑到 Visual Basic 的学习者往往不是从事专业程序设计的人员，所以，本书总体思路是程序设计方法与界面设计方法并重。在程序设计方法方面只介绍常用的基本方法和算法，略去例如递归等稍难一些的技术。在界面设计方面，使用较多的篇幅介绍各种常用控件及其应用。例题中，除非常简单的界面外，对程序中所使用的控件都加以说明。上机练习也大多给出界面图示，希望为学生设计界面起到一个引导作用。

2．尽可能将学习算法设计的难度和学习 Visual Basic 语言成分及界面元素的难度分散

根据长期高级语言程序设计课程的教学实践，发现学生在初学程序设计时的主要困难出现在算法设计方面。目前，很多教材的算法设计与算法的实现（用所学的高级语言实现）是穿插在一起介绍的，学生往往把对算法的理解和对语言成分的理解混在一起，增加了学习的难度。Visual Basic 除了这两部分内容之外，还有很多控件需要介绍，使得在解决一个问题时，要涉及三方面的知识点，更增加了难度。编写本书的另一个思路是尽可能将这三方面的难度分散，在全面介绍语言成分和常用控件之前，先讲解设计算法的基本思维方式。为此，在学习程序基本结构之前，比较详细地介绍了几种最常见的基本算法，讲解例题，突出学习设计算法的步骤和思维方式，并经过习题训练学生这方面的能力，希望能够把学习算法的设计和用 Visual Basic 语言实现算法的难度分散，从而降低学习的整体难度，为后面的学习打下基础。具体做法是：在第 1 章中介绍算法的基本概念、算法的描述方法（流程图），在第 3 章中介绍几个基本算法（与实现无关），它们是构成后面章节介绍的算法的基础，在第 4 章再讨论如何使用 Visual Basic 的各种语句实现第 3 章介绍的算法。在后续章节中也尽量贯穿这一思路。

例题的介绍也尽量体现这种组织方式，对于复杂一些的例题采用下面的模式讲解：

① 问题分析：分析问题的求解内容，建立模型。

② 解题思路：介绍解决问题的方案，确定所要使用的技术手段，明确所使用的数据和数据的组织形式。如果比较简单，就合并到问题分析中去。

③ 设计算法：通过流程图给出解题的具体步骤。

④ 界面设计：列出主要控件的用途、主要属性，给出界面的图示。

⑤ 实现：给出程序代码。

⑥ 必要时给出程序说明，对程序做解释。

对于比较简单的例题，可能省略上述中的一项或数项，以避免烦琐。

3. 使学生尽快地掌握简单界面的创建，看到学习成果，提高学习兴趣

Visual Basic 的特点之一是可以很快地创建一个 Windows 环境下的应用程序界面。本书力图使学生可以尽快地掌握简单界面的创建，看到学习成果，提高学习兴趣，这样做也有助于学生克服学习困难。所以，在教材的结构上尽量把对常用控件的介绍提前。整个教材对程序设计与 Visual Basic 语言的介绍和对 Visual Basic 界面元素的介绍按章穿插进行，试图避免因长时间学习程序设计与 Visual Basic 语言而带来的枯燥。为提高学生兴趣，还设计了一些涉及简单动画的例题与习题。

4. 通过大量例题和习题，帮助学生加深对所学知识的理解和掌握

Visual Basic 程序设计是一门实践性比较强的课程，对于非计算机专业的学生来说，阅读程序实例和动手编程是学习程序设计的有效手段。本书各章都配有较多例题和习题，便于学生复习、巩固所学知识，锻炼编程能力，也有助于学生自学。在介绍了常用控件之后，习题中尽量做到界面与算法相结合。

5. 涵盖全国计算机等级考试二级 Visual Basic 考试的知识点

本书在选材上参考了全国计算机等级考试二级 Visual Basic 考试大纲，内容涵盖了全国计算机等级考试二级 Visual Basic 考试的绝大部分知识点。

本书共分 11 章，主要内容包括：对象和事件驱动的概念、Visual Basic 程序开发环境、Visual Basic 程序设计基础、程序控制结构、窗体和常用控件、数组、过程、菜单、通用对话框、多窗体界面、图形设计、文件处理、数据库应用基础等。

本书所有例题的程序代码都在 Visual Basic 6.0 企业版上调试通过。

本书适合作为高等学校非计算机专业学生学习 Visual Basic 程序设计的教材，也可供其他对程序设计有兴趣的读者学习、参考。

为方便教学，本书配有电子课件，有兴趣的读者可到 http://edu.tqbooks.net 网站下载。

由于编者水平有限，缺点和错误在所难免，敬请读者批评指正。

编 者

2010 年 9 月

CONTENTS

目 录

第 1 章 Visual Basic 程序设计概述

程序是对计算任务的处理对象和处理过程的描述，计算机程序设计就是为计算机编写程序的过程，包括设计、编写和调试程序的方法和过程。程序设计涉及程序设计方法、程序设计语言等各方面知识。本章首先介绍计算机程序设计语言、程序设计的基本方法，然后介绍 Visual Basic 6.0 集成开发环境的构成与使用。

学习目标

- 了解程序设计语言和算法的基本概念，知道 Visual Basic 6.0 的版本和安装方法。
- 理解结构化程序设计方法，对象、事件驱动的概念和面向对象程序设计的思想。
- 掌握用流程图描述算法的方法，掌握 Visual Basic 6.0 集成开发环境的使用。

1.1 程序设计语言与程序设计

程序设计语言是人与计算机交流的媒介，程序通过程序设计语言表示出来。程序设计是设计、编写、调试程序的过程，为了保证程序的质量，程序设计应遵循一定的科学方法进行。

1.1.1 程序设计语言

人与人之间要通过语言进行交流，人与计算机之间也需要一种语言来进行交流，人们用这种语言来描述需要计算机执行的操作，这种描述体现为程序的形式，因此这种语言又称程序设计语言。程序设计语言需要具有描述要求计算机所进行的各种操作和所处理的各种数据的能力，还要能够描述操作的流程顺序和对流程的控制。

随着计算机技术的不断发展，人们设计了多种计算机程序设计语言，程序设计语言的发展也经历了由低级到高级的发展历程。在这一发展过程中，程序设计语言的表达能力在不断加强，而用程序设计语言设计程序的难度却在不断降低。

1. 机器语言

计算机只能识别由一串“0”和“1”组成的二进制编码所代表的命令，这种命令称为机器指令，一条机器指令规定了 CPU 的一种基本操作。所有机器指令的集合构成了这种 CPU 的指令系统，规定了 CPU 所能进行的所有基本操作。每一种 CPU 都有一套自己的指令系统，CPU 不同，其指令系统一般也不同。机器语言与指令系统相对应，它用二进制编码来描述要进行的操作和要处理的数据，是计算机能够直接识别的语言。

机器语言的特点是，计算机可以直接执行用机器语言编写的程序，程序运行的速度最快，

占用系统资源最少。但程序的编写难度最大，程序不易阅读，修改、调试也很不方便，不能在采用另一种 CPU 的计算机上运行。

2. 汇编语言

为了便于阅读和记忆，人们采用被称为“助记符”的英文缩写符号和地址符号来代替机器指令的二进制编码，这种由助记符构成的指令称为“汇编指令”，汇编指令的集合及其规则就构成了“汇编语言”。用汇编语言编写的程序叫汇编语言源程序。但计算机不能识别汇编语言，所以必须把汇编语言源程序中的汇编指令翻译成机器指令，完成这一工作的程序称为“汇编程序”。

由于汇编指令与机器指令基本是一一对应的，不同 CPU 的汇编指令系统也不同，因此用汇编语言编写的程序一般也不能移植。

用汇编语言编写的程序的执行效率比较高，占用的系统资源比较少，其编写难度、阅读、修改和调试的难度虽然比机器语言的难度要低一些，但仍与人的自然语言和数学语言有较大差异，其通用性和可移植性仍比较差。

机器语言与汇编语言统称为“低级语言”，它们都与硬件密切相关，所以也称为“面向机器的语言”。

3. 高级语言

高级语言是更接近于人的自然语言和数学语言的计算机语言。通常所说的程序设计语言往往是指高级语言。高级语言与具体的计算机硬件无关，它的表达方式有利于人们对解题过程进行描述。

与低级语言相比，用高级语言编写程序的难度大大降低，编写程序的效率大幅提高，程序的阅读、修改和调试也更加容易。但程序的执行效率降低了，占用的系统资源也更多了。由于与计算机硬件无关，高级语言编写的程序具有更好的通用性和可移植性。

高级语言使用不直接对应于机器指令的语句来实现特定的操作。语句的集合以及相关的语法规则构成了高级语言。高级语言往往还配有事先编写好的多个子程序可在程序中直接使用，大大方便了程序的编写。

最早出现的高级语言是 20 世纪 50 年代后期推出的 Fortran 语言，之后又陆续出现了多种高级语言，其中比较常见的有 BASIC、Pascal、COBOL、C、C++、Java 等。Visual Basic 也是一种高级语言。

高级语言源程序不能在计算机上直接运行，必须把它翻译成机器指令序列才能在计算机上运行。翻译的方式有两种：编译方式和解释方式，实现翻译的程序分别称为“编译程序”和“解释程序”。

高级语言源程序之所以能在不同种类计算机上运行，是因为该种计算机配有把这种高级语言源程序翻译成自己机器语言程序的编译程序或解释程序，并把它翻译成了自己的机器语言指令序列后再加以运行。这就如同不同地区的人们想要看懂英语电影，就必须把英语对白翻译成本地语言后再放映，或者在每次放映英语电影期间，配备一个同声翻译，一句一句地翻译。这两种翻译方法中，前者对应于高级语言的编译方式，后者对应于高级语言的解释方式。

编译是指把高级语言源程序翻译为在功能上等价的本计算机的机器语言程序——目标代码程序。在此之后，在计算机上执行的是目标代码程序，并且可以多次执行。执行目标代码程序期间不需要源程序和编译程序的参与。但是，一旦对源程序做了修改，则需要重新编译一次，

产生新的目标代码程序，然后才能执行。所以，编译方式的特点是：一次编译，多次执行；一旦修改，重新编译。

解释方式不产生目标代码程序。与人类语言的同声翻译类似，在执行源程序时，解释程序对源程序的语句逐条翻译，翻译一句，执行一句，重复的语句也要重复翻译。源程序全部翻译完毕，程序的执行也就结束了。下次执行时，还需要解释程序重新逐语句翻译。源程序修改后，仍用同样的方式逐句翻译执行。因此，每次执行程序时，都需要源程序和解释程序的存在。解释方式的特点是：每次执行，重新翻译；翻译一句，执行一句。一般来说，编译执行比解释执行的效率更高。

BASIC 语言采用解释方式，Fortran、Pascal、C 等语言采用编译方式。Visual Basic 既可以在集成开发环境中解释运行，又可以编译成目标代码程序后在操作系统下直接运行。

1.1.2　算法

设计计算机程序的一个重要任务是设计算法，算法不仅包含对每个运算步骤的描述，还要有对各个步骤执行次序的控制，即对程序运行的过程进行控制。算法是否正确，将决定程序是否能得到正确结果。

1．算法的概念

任何解决问题的过程都由一定的步骤组成。计算机算法简单地说就是指用计算机解决问题所采用的基本方法和操作步骤。

例如，输入两个数 x 和 y，要求输出其中较大的一个，就可以采用下面的操作步骤：

① 输入 x，y；

② 判断：x > y，是则继续执行下一步，否则跳到第④步；

③ 输出 x，程序结束；

④ 输出 y，程序结束。

上述 4 个步骤就是输出 x、y 中较大数的算法。

可见，算法不仅包括每个步骤的具体操作，还要确定各个步骤的先后顺序，即对操作流程的控制。

算法的设计与算法的实现是不同的。算法的设计是要确定解决问题的方法与步骤，要保证在逻辑上是正确的、可行的；算法的实现是要用一种程序设计语言，在语法规则的限制下，按照设计好的算法编写计算机程序，最终获得算法所预期的结果。这时，要选择合适的语句或语句序列来实现算法所描述的每个操作和算法所规定的流程控制。通常，同一个算法可以用多种不同的高级语言来实现。

2．算法的特征

一个计算机算法应该具有以下特征：

① 有穷性：构成算法的操作步骤应该是有穷的，而且每一步可以在有限时间内完成。

② 确定性：算法中的每一步必须是确定的，不能有模糊的含义，不能产生二义性。

③ 有效性：算法中的每一步应能有效地执行，并产生有效和确定的结果。

④ 有 0 个或多个输入：即如果必要，在执行算法时，可以从外界获得信息。

⑤ 有 1 个或多个输出：算法的目的是为了求解问题，问题的解就是算法的输出。

3. 算法的描述

算法中的各个步骤和流程的控制需要用适当的形式表示出来，这不仅便于算法的修改与保存，也便于算法的设计。

（1）自然语言描述

用自然语言描述算法容易掌握，描述的算法通俗易懂。上面关于输出 x、y 中较大数的描述就是自然语言描述算法的例子。用自然语言描述算法的主要缺点是容易产生“二义性”，如果描述不够严谨，对同一算法，不同人可能会有不同的理解。而且，用自然语言表达操作步骤的语句往往比较烦琐冗长，不容易表达清楚算法的逻辑流程。此外，算法的表示与计算机高级语言的表达形式也有较大差异，对初学者来说，把自然语言描述的算法写成高级语言源程序有一定困难。

（2）伪码描述

伪码是一种与高级语言类似，又比高级语言简单的符号系统。伪码的组成没有固定标准，可以自己约定，以能够完整、准确描述算法内容为准，一般把某种高级语言加以简化，再附加少量约定，称为“类某某语言”，作为描述算法的伪码。伪码没有复杂的语法规则，书写比较自由，算法的修改比较容易。伪码的表达能力比较强，描述算法比较严谨，与高级语言的表达形式比较接近，把算法写成高级语言源程序比较容易。但对未学过程序设计的初学者来说，要掌握用伪码描述算法这一方式有一定困难。

（3）图形描述

图形描述是介于自然语言描述和伪码描述之间的算法描述方式。人们定义了一些基本的图形符号，然后用这些图形符号再辅以自然语言和数学语言的简单表述来描述算法。图形描述方式是一种比较直观、容易理解也容易掌握的算法描述方式。

图形描述方式有多种，比较常用的有流程图、N-S 图、PAD 图等。本书采用流程图来描述算法。流程图的基本图形符号如图 1-1 所示。

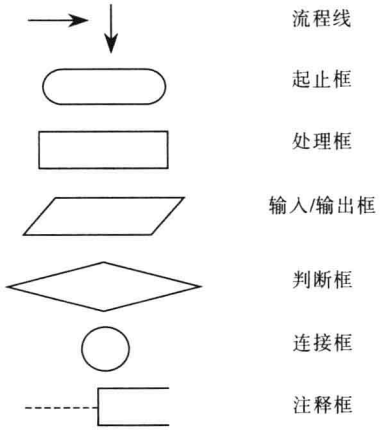

图 1-1　流程图的元素

其中：

流程线：是有向线，用于表示流程的路径和方向。

起止框：用于描述控制流程的开始与结束。

处理框：用于描述输入/输出之外的其他具体操作，框中用文字或数学符号来表示操作的具体内容。处理框有一条流入的流程线和一条流出的流程线。

输入/输出框：用于表示输入或输出操作，框内写明输入或输出的具体内容。输入输出框有一条流入的流程线和一条流出的流程线。

判断框：用于描述判断条件和流程转移关系。有一条流程线流入，两条流程线流出。框内写条件，根据条件的不同，选择不同的流向。

连接框：用于描述多张流程图的连接，可在框中标注标识来说明连接关系。

注释框：注释框本身不构成算法的一部分，只用于对算法中某个操作加以说明，说明文字写在框中。

图 1-2 是前面所介绍的“输出 x、y 中较大数”算法的流程图，其中“T”表示判断框中的条件成立，“F”表示判断框中的条件不成立。这个算法比较简单，更多的例子将在第 3 章中介绍。

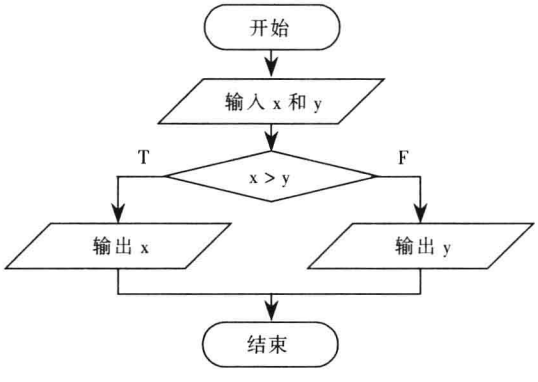

图 1-2　输出 x、y 中较大值的算法

流程图的缺点是对流程线的走向没有限制，流程的转移过于灵活，设计算法时流程转移如果过于随意，就会造成流程控制的混乱，导致算法不易理解，不易修改。

除了上述 3 种算法描述方式外，还可以用某种高级语言来直接描述算法。

1.1.3　结构化程序设计

早期的程序设计缺乏理论指导，程序流程的转移比较随意，当要处理的问题规模比较大、逻辑比较复杂时，程序就很难理解，并且增加了程序修改、调试和维护的难度，降低了程序的可靠性。为避免这种现象，人们开始研究程序设计方法，并于 20 世纪 60 年代末提出了“结构化程序设计”的思想。结构化程序设计方法要求设计者按照一定的原则和方法来设计程序，强调程序结构的规范化，使程序结构清晰，容易阅读，容易理解，容易修改，容易维护。

结构化程序设计的基本内容包括：

（1）自顶向下，逐步求精的分析方法

遇到一个大而复杂的问题时，首先对问题整体进行分析，按组织或功能把问题分解为若干

个相对简单的子问题，如果子问题仍然复杂，再进一步分解，直到子问题足够简单、容易处理为止，最后形成具有层次结构的分析结果。在不断分解的过程中，每一次分解都是对上一次分解的细化，是逐步求精的过程。

（2）模块化设计

程序由若干个功能相对独立的模块组成。上层模块调用下层模块，每个模块解决一个子问题，实现一个相对独立的功能，最底层模块实现最具体、最基本的功能。模块之间通过参数传递信息，模块内的数据对其他模块是隐蔽的。

（3）结构化编码

每个模块单独编码。编码时采用 3 种基本控制结构，保证程序结构的规范化。这 3 种基本结构是顺序结构、选择结构和循环结构。

顺序结构的流程图如图 1-3 所示。顺序结构按照各个处理的先后顺序依次执行。

选择结构的流程图如图 1-4（a）所示。它由一个条件和 2 个处理组成，根据条件是否满足决定执行哪个处理。也可以省略其中的一个，如图 1-4（b）所示。

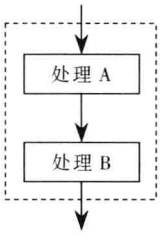

图 1-3　顺序结构

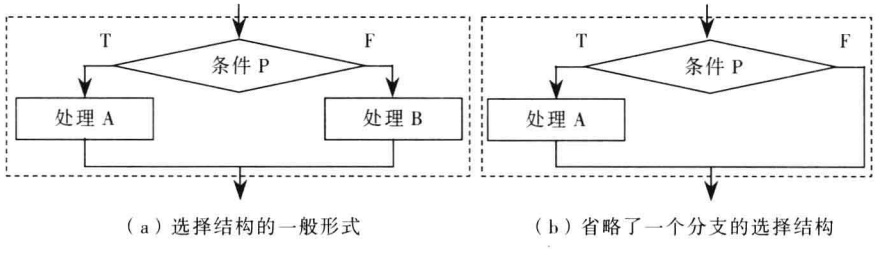

图 1-4　选择结构

循环结构的流程图如图 1-5 所示，它是一种重复执行的结构。根据条件是否满足，决定是否重复执行其中的处理。循环结构分为 2 种：当型循环和直到型循环。前者先判断条件，当条件满足时执行处理；后者先执行处理，后判断条件，条件不满足时重复，直到条件满足为止。

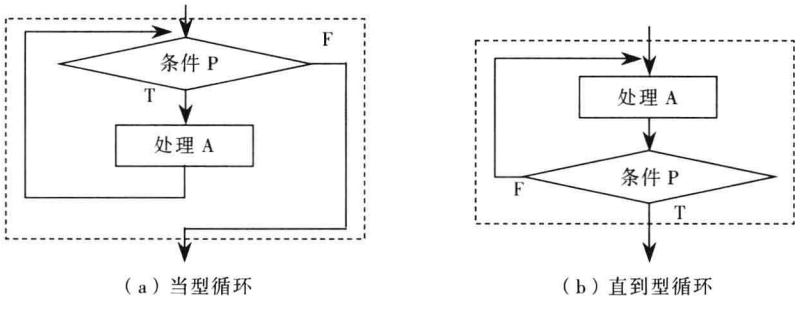

图 1-5　循环结构

利用这 3 种基本结构和它们的嵌套、组合可以解决任何复杂问题。

1.1.4　面向对象程序设计

自顶向下，逐步求精是一种基于功能分解的分析方法。它把复杂问题逐步分解，然后抽象为一系列子程序，通过子程序的调用达到模块化，最终组成完整的程序。这种程序设计的思想

被称为“面向过程的”或“过程化的”程序设计。结构化程序设计方法是设计过程化程序的有效方法。

解决实际问题时，不仅要考虑解题的步骤，还要考虑解题的对象，即需要处理的数据。实际上，程序是通过对数据的操作来解决问题的。

过程化的程序设计更注重解决问题的过程，它以操作为中心，对操作与要处理数据的描述是分离的，而在现实世界的问题中，操作往往是针对数据的，是围绕数据进行的。

在实际问题中，人们更直接看到的是组成问题的一个个对象，而不是一个个功能，所以，面向过程的程序对问题的描述与人们实际观察到的问题有一定差异。

面向对象的思想则是一种完全不同的设计和构造软件的思维方式。面向对象的思想不是把问题分解为过程，而是把问题分解为对象，而对象既具有自己的特征，又具有一定的行为能力，这与人们习惯的思维方式比较吻合，能更直接地描述客观世界。因此，软件的可维护性、可扩充性和可重用性也就更好，可以提高软件开发效率。

面向对象的主要内容有：

（1）对象和类

面向对象方法从问题所涉及的对象入手，以对象为中心构成程序。对象既包含描述对象的数据（称为对象的属性），也包含了针对这些数据所进行的操作（称为对象的方法）。类则是对相同性质对象特征（属性与方法）的描述，一个类刻画一组具有相同特性的对象，是对象的集合，而对象则是类的实例。

（2）消息

通过传递消息来进行对象之间的联系，对象可以向其他对象发送消息，请求服务，也可以响应其他对象发来的消息。

（3）封装

把对象的属性和方法包装在一起，隐蔽对象内部的实现细节，外部只有通过对象的方法才能处理对象内部的数据。封装隐藏了对象内部的复杂性，简化了对象的使用，可以像一个部件一样在程序中使用对象。

（4）继承

在现实世界中，有些对象是另一类对象的子集，例如，小学生、中学生都是学生的子集，小轿车、货车都是汽车的子集。子集一般具有其父集的全部或部分特征，当然一般还具有不同于父集的特征。面向对象中的继承是指定义一个类时，可以从另一个类或多个类继承特征。继承是实现代码复用的一种重要机制。

（5）多态

在面向对象的程序设计中，多态性是指在同一个类或不同类中，可以定义名称相同但操作不同的多个方法。多态性的主要好处是易于实现程序高层代码的复用，使程序容易扩充。

1.2　对象的概念

对象是面向对象程序设计的核心，一个面向对象的程序是由若干个描述对象特征的类组成的。