



福建省高校计算机
统编教材（二级）

Delphi

程序设计学习与实验指导

福建省高校非计算机专业教材编写委员会 组织编写
宁正元 主编

厦门大学出版社

福建省高校计算机统编教材

Delphi 程序设计 学习与实验指导

主 编 宁正元

编写者 (以姓氏笔画为序)

刘雄恩

陈 琼

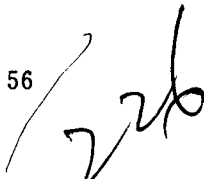
厦门大学出版社

图书在版编目(CIP)数据

Delphi 程序设计学习与实验指导/宁正元主编. —厦门:厦门大学出版社,2005.10
福建省高校计算机统编教材
ISBN 7-5615-2454-4

I. D… II. 宁… III. 软件工具-程序设计-高等学校-教材 IV. TP311.56

中国版本图书馆 CIP 数据核字(2005)第 112822 号



厦门大学出版社出版发行

(地址:厦门大学 邮编:361005)

<http://www.xmupress.com>

[xmup @ public. xm. fj. cn](mailto:xmup@public.xm.fj.cn)

福建沙县方圆印刷有限公司印刷

2005 年 10 月第 1 版 2005 年 10 月第 1 次印刷

开本:787×1092 1/16 印张:13.5

字数:345 千字 印数:0001-2000 册

定价:18.50 元

如有印装质量问题请与承印厂调换

前 言

本书是福建省高校计算机统编教材《Delphi 程序设计与应用教程》的配套学习指导书,用于辅助该课程的教学与实验,是理想的学习辅导材料。在篇章的安排上力求与教程保持一致,以便读者结合教程更好地掌握基本概念、操作和设计方法。

全书分为“学习指导与习题”、“实验指导”和“模拟试题”三篇,最后附有部分参考答案和“福建省高等院校计算机应用水平等级考试二级 Delphi 程序设计考试大纲”(修订稿)。第一篇对教程各章节知识点进行了概况和总结,突出重点、难点;精选了大量例题,并对它们进行了详细的解答;设计了丰富的习题。第二篇设计了 11 个实验指导,每个实验均明确指出实验目的,提供详细的实验示例(操作与设计步骤)和实验习题;第三篇模拟试题的设计与考试大纲的要求相吻合。

本书由福建农林大学计算机与信息学院宁正元教授主编,负责全书内容的安排和统稿。第一篇第 1、5、6、7 章和第二、三篇由福建农林大学刘雄恩编写,第一篇第 2、3、4 章由福建农林大学陈琼编写。

在本书编写过程中,福建省高校计算机水平考试委员会的专家们给予了许多指导和支持;厦门大学出版社宋文艳编审和眭蔚编辑对本书的出版付出了大量的劳动。在此表示诚挚的谢意。

鉴于作者的学识和水平,书中疏漏和不当之处恳请读者指正。

作者

2005 年 9 月 23 日

目 录

第一篇 学习指导与习题

第 1 章 Delphi 简介	1
本章知识点	1
例题精解	2
习题	2
第 2 章 Object Pascal 语言基础	4
本章知识点	4
例题精解	20
习题	23
第 3 章 Delphi 中的面向对象	42
本章知识点	42
例题精解	46
习题	47
第 4 章 运用组件编程	50
本章知识点	50
例题精解	58
习题	61
第 5 章 程序调试和异常处理	75
本章知识点	75
例题精解	77
习题	78
第 6 章 Windows 深入编程初步	83
本章知识点	83
例题精解	90
习题	97
第 7 章 数据库应用编程	102
本章知识点	102
例题精解	105
习题	113

第二篇 实验指导

实验一 Delphi 程序设计初识	115
实验二 选择分支程序设计	119
实验三 循环处理程序设计	125
实验四 数组的运用	131
实验五 过程与函数	139
实验六 自定义数据类型	145
实验七 面向对象程序设计	151
实验八 常用组件的使用(一)	155
实验九 常用组件的使用(二)	162
实验十 图形与多媒体编程	170
实验十一 简单数据库应用编程	177

第三篇 模拟试题

模拟试题一	182
模拟试题二	187

部分参考答案	193
--------	-----

附录 福建省高等院校计算机应用水平等级考试二级 Delphi 程序设计考试大纲(修订稿)	205
---	-----

第一篇 学习指导与习题

第1章 Delphi 简介

一、本章知识点

1.1 Delphi 的基本特点

领会 Delphi 是一个完全集成化、面向对象、可视化，能编制各种 Windows 程序，功能强大又易学易用的软件开发平台。

1.2 Delphi 的集成环境

Delphi 的卓越性能也体现在它的集成开发环境（IDE）上。通过简单实例的开发过程，了解其 IDE 的基本组成和特点。

Delphi 的 IDE 由以下几个部分组成：

- 标题栏；
- 菜单栏；
- 工具栏；
- 组件板；
- 对象观察器（Object Inspector）；
- 对象树结构观察器（Object TreeView）；
- 窗体设计窗口（Form）；
- 代码编辑器（Code Editor）。

1.3 Delphi 的文件种类

一个 Delphi 源程序是由多个文件组成的，这些文件组合起来称为一个项目。一个编制中的应用程序就是一个项目，每个项目由以下几种文件组成：

- 项目文件 (*.dpr)；
- 单元文件 (*.pas)；
- 窗体文件 (*.dfm)；

- 配置文件 (*.dof, *.cfg);
- 资源文件 (*.res);
- 中间编译单元 (*.dcu);
- 备份文件 (*.~*)。

二、例题精解

【例 1-1】如何在 Delphi 的 IDE 中创建一个新的程序项目？

解：首先依据需要，保存现有程序项目的所有文件，然后新建项目。可按如下步骤实现：

(1) 执行【File】→【Save All...】菜单项命令或点击工具栏上的“Save All”快捷键，在相继打开的保存文件对话框中完成项目文件、单元文件等的保存，其他文件如窗体文件、资源文件、配置文件等，IDE 会自动保存。

(2) 执行【File】→【New】菜单项命令或点击工具栏上的“New”快捷键，在打开的新建项目对话框中选择所要建立项目的类型后，确认即可，IDE 将自动初始化所建项目并打开相应文件和设计器窗口。

【例 1-2】一个 Delphi 项目中有下列文件：Myprog.dpr、Myprog1.dfm、Myprog1.pas、Myprog1.dcu、Myprog.res，其中，哪个（些）不能用记事本打开来浏览和编辑？

解：能用记事本打开的文件应是由字符组成的文本文件，Delphi 的项目文件 (*.dpr)、单元文件 (*.pas) 是文本文件，窗体文件 (*.dfm) 也默认保存为文本文件，而资源文件 (*.res) 不是文本文件，编译单元后产生的中间文件 (*.dcu) 显然也不是文本文件，因此，其中的 Myprog1.dcu、Myprog.res 不能用记事本打开来浏览和编辑。

三、习题

一、选择题

1. Delphi 中使用的编程语言是 ()。

- A) Object Pascal
- B) Pascal
- C) Delphi
- D) C++

2. 下列方法中，() 是唯一不能打开 Delphi 帮助文件的方法。

- A) 通过执行 Delphi 主菜单“Help”下的菜单项
- B) 通过执行 Delphi 主菜单“Tools”下的菜单项
- C) 通过执行系统的【开始】→【程序】→【Borland Delphi 7】→【Help】下的菜单项
- D) 在代码编辑器窗口中，使用热键 Ctrl + F1 或 F1

3. 下列叙述中，() 是唯一正确的。

- A) Delphi 是一种跨平台的高级程序设计语言
- B) 在 IDE 中, 执行“Run”菜单项或按 F9 键, 编写好的程序即可运行, 这表明源程序无须编译就可以直接运行
- C) 在 Delphi 的 IDE 中新建一个 Application 程序项目后, 程序员不必编写任何代码, 该程序已可执行
- D) Delphi 的源程序中, 只能使用 Object Pascal 语言
4. Delphi 的单元文件经编译后产生的文件的扩展名是 ()。
- A) EXE B) DCU C) DPR D) DFM
5. 在 Delphi 的 IDE 中打开一个程序项目, 是以打开扩展名为 () 的文件开始的。
- A) PAS B) DCU C) DPR D) DFM

二、编程题

1. 试对 Delphi 集成开发环境作适当设置。如通过选择执行【Tools】→【Editor Options】菜单项打开的对话框, 将编辑器中的颜色和字体设置为个人喜好的格式。

2. 编制一个应用程序 (Application), 在窗体设计器窗口上放置一个按钮组件 (Button), 将按钮的 Caption 属性改为“退出”, 并在其 OnClick 事件代码中写下:

```
Application.Terminate;
```

保存项目和单元文件后, 编译和运行程序, 最后关闭项目的所有文件。

3. 打开上一题建立的程序项目, 在代码编辑窗口中找到“退出”按钮的 OnClick 事件处理过程, 在“Application.Terminate;”一句的前面加上行注释“//”(连续两个斜杆符号), 使该语句在运行时不被执行, 然后插入如下语句:

```
Close;
```

再保存、编译和运行程序, 注意程序的行为。(如有疑惑可保留, 待学习到第四章后便可顿悟。)

找到代码位置的方法提示如下:

方法一: 执行 IDE 的【Search】→【Save All...】菜单项命令, 在打开的查找对话框中查找“Terminate”。

方法二: 在窗体设计器上选中“退出”按钮组件, 然后在对象观察器窗口的 Events 页上鼠标双击 OnClick 处, 代码编辑器窗口上光标即定位在该按钮的 OnClick 事件处理过程处。

方法三: 在窗体设计器上双击“退出”按钮组件, 代码编辑器窗口上光标即定位在该按钮的 OnClick 事件处理过程处。

var

<变量标识符表 1>:<类型标识符 1>;

<变量标识符表 2>:<类型标识符 2>;

.....

<变量标识符表 n>:<类型标识符 n>;

2. 常量

在程序中，其值保持不变的量称为常量。常量有两种表示形式：字面常量和符号常量。

(1) 字面常量

字面常量按照其数据类型可分为整型常量、实型常量、布尔常量和字符常量等。

整型常量可以用十进制表示，也可以用十六进制形式表示，十六进制数以字符\$开头。

实型常量有一般表示形式和指数计数法两种，Object Pascal 中的实型常量表示和日常书写的形式基本相似。布尔常量只有 False 和 True 两个值，False 表示逻辑值“假”，True 表示逻辑值“真”。字符常量必须使用定界符定界，Object Pascal 的字符定界符有两种：单引号和#。其中#引导一个整数，整数对应字符的 ASCII 码。字符串常量是用单引号定界的一串字符，当字符串内有单引号时，要用两个连续单引号表示。

(2) 符号常量

符号常量就是使用标识符来表示常量，符号常量的意义在于减少常量值差错机会与修改程序的工作量，并提高程序的可读性。符号常量需预先定义。其定义的一般形式为：

const

<常量标识符>=表达式;

以保留字 const 后开始常量声明；“=”号左边为常量标识符；“=”号右边的表达式可以由常量、部分在程序编译时可计算的函数及先定义的常量标识符等构成，表示其值。

3. 类型常量

类型常量和变量一样需要先定义再使用，以保留字 const 开头。通过常量定义所给出的类型常量的用法与变量完全一样，允许在程序中对类型常量重新赋值。一个类型常量定义部分可包含多个类型常量定义，每个类型常量定义末尾均有分号，其一般形式为：

const

<标识符 1>: <类型标识符 1>=表达式 1;

<标识符 2>: <类型标识符 2>=表达式 2;

.....

<标识符 n>: <类型标识符 n>=表达式 n;

2.2 保留字和指令字

1. 保留字

保留字是 Object Pascal 中具有特殊语法含义的单词，这些保留字不能被定义为标识符。在帮助文件中查找 Reserved Words，可以了解 Object Pascal 中定义的 65 个保留字及其用法。

2. 指令字

Object Pascal 语言中定义的指令字为编译提供指示意义。各指令字有它专门的用途，如 absolute 指令字用于指示相同类型的另一标识符存放在同一段内存区域，overload 用于指示函数或过程的重载。指令字可以用作标识符而被重新定义，当重新定义后，在所定义的作用域内它就失去了原来的意义。

其中，private、protected、public、published、automated、on、at 是指令字，但应当视其为保留字，不能定义和它们同名的标识符。

2.3 简单数据类型

Object Pascal 是一种强类型语言，拥有丰富的数据类型，并允许程序员自定义新的数据类型。Object Pascal 中数据类型分为简单类型、字符串类型、结构类型、指针类型、过程与函数类型和可变类型等，如图 2-1 所示。

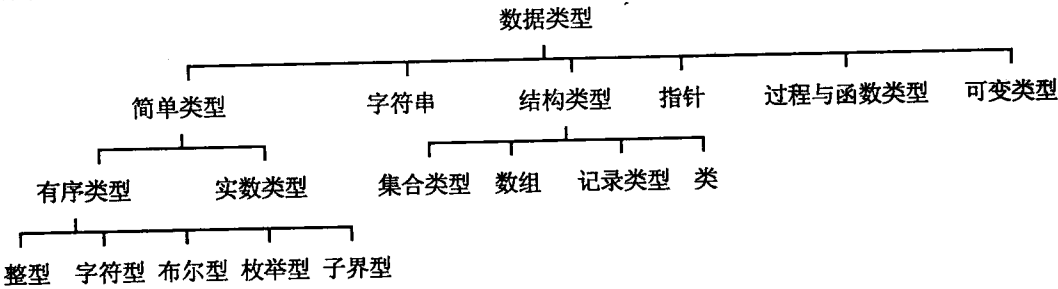


图 2-1 数据类型分类

简单数据类型包括有序类型 (Ordinal) 和实数类型 (Real)，有序类型又包括整型、字符型、布尔型、枚举型和子界型。

1. 有序类型

(1) 整型

常见整数类型见表 2-1。

表 2-1 Object Pascal 语言中的整数类型

整数类型	取值范围
Integer	Signed 32 bit
Cardinal	Unsigned 32 bit
Shortint	Signed 8 bit
Smallint	Signed 16 bit
Int64	Signed 64 bit
Byte	Unsigned 8 bit
Word	Unsigned 16 bit

(2) 字符型

Object Pascal 语言的字符数据类型包括以下三种:

- **AnsiChar**: 标准的单字节 ANSI 字符。
- **WideChar**: 两字节的 Unicode 字符。Unicode 字符集的前 256 个字符与 ANSI 字符集相同。
- **Char**: 相当于 AnsiChar。

(3) 布尔型

布尔型包括 Boolean、ByteBool、WordBool、LongBool 四种。其中 Boolean 和 ByteBool 为单字节, WordBool 为双字节, LongBool 为四字节。Object Pascal 预定义了两个常量标识符 False 和 True, 可以赋值给 Boolean 型数据, 对应的序数值为 0 和 1。ByteBool、WordBool 和 LongBool 型数据值为 0 时表示 False, 非 0 时表示 True。

(4) 枚举型

枚举型由一组有序的标识符组成, 枚举型说明中列出了所有这种类型可以包括的常数。声明枚举型的语法如下:

```
type typeName = (val1, ..., valn);
```

其中 typeName 和 vali 是有效的标识符, vali 是枚举常数标识符。在默认的情况下, 枚举值是从 0 开始的有序整数, 取值范围为 0~255。

(5) 子界型

当用户需要限制一个变量的取值范围时, 可以通过子界型定义一个有序类型的子集。其声明形式如下:

```
type
```

```
类型标识符 = 上界值..下界值;
```

其中上、下界值是某种已定义的有序类型的值, 上界值小于或等于下界值。该类型是子界型的基类型, 子界型的取值范围在上、下界之间。

2. 实数类型

常见实数类型见表 2-2。

表 2-2 Object Pascal 语言中的实数类型

实数类型	取值范围	有效位数	字节数
Real	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$, $-1.7 \times 10^{308} \dots -5.0 \times 10^{-324}$	15~16	8
Single	$1.5 \times 10^{-45} \dots 3.4 \times 10^{38}$, $-3.4 \times 10^{38} \dots -1.5 \times 10^{-45}$	7~8	4
Double	$5 \times 10^{-324} \dots 1.7 \times 10^{308}$, $-1.7 \times 10^{308} \dots -5.0 \times 10^{-324}$	15~16	8
Extended	$3.6 \times 10^{-4951} \dots 1.1 \times 10^{4932}$, $-1.1 \times 10^{4932} \dots -3.6 \times 10^{-4951}$	19~20	10
Comp	$-2^{63} \dots 2^{63}-1$	19~20	8
Currency	-922337203685477.5808..922337203685477.5807	19~20	8

3. 日期时间类型

日期时间型 TDateTime 在 Object Pascal 中实际是双精度浮点型 (Double)，其原型是：

```
TDateTime = Double;
```

```
TDate = TDateTime;
```

```
TTime = TDateTime;
```

Object Pascal 中以 1899 年 12 月 30 日 12:00 时为准，计为 0，每增加 1 天，日期时间类型数据增 1。因此，整数部分表示距离该日该时的天数。

2.4 结构类型

结构类型包括集合类型、数组类型、记录类型、文件类型、类类型、类引用类型、接口类型。

1. 集合类型

集合类型是一组相同类型元素的集合，这些类型必须是有序类型，如整型、布尔型、字符型、枚举型和子界型。Object Pascal 中规定基类型只能是不超过 256 个有序值的集合，集合的元素的序数值必须介于 0 和 255 之间。

2. 数组类型

数组类型是一组相同数据类型元素的有序集合，每个数据元素都有一个唯一的索引号。与集合类型不同的是，数组类型的数据元素可以重复。数组类型分为静态数组和动态数组。数组有一维和多维两种形式。

(1) 静态数组

静态数组是指定了大小的数组。定义静态数组的语法形式如下：

```
type Array [indexType1, ..., indexTypen] of baseType;
```

其中，indexType 是有序类型，其范围不能超过 2 GB，indexType 通常以 Integer 为子界，也可以是其他有序类型的子界；baseType 声明数组元素的数据类型。

(2) 动态数组

动态数组在定义时不确定数组的大小或长度。定义动态数组的语法形式如下：

```
type 数组名: Array of baseType;
```

在访问数组之前用 SetLength 过程为数组动态或重新分配存储空间。语句格式为：

```
SetLength(动态数组名, 元素个数);
```

该语句为动态数组分配指定元素个数的存储空间，下标从 0 开始。如要释放动态数组占用的存储空间，可以将 nil 赋值给该动态数组变量，或通过调用过程 SetLength(动态数组名, 0) 实现。

多维动态数组通过使用嵌套的“Array of ...”结构定义。

3. 字符串类型

字符串是一个字符系列。字符串类型有：

- AnsiString: Pascal 缺省的字符串类型，由 AnsiChar 字符组成。默认情况下其长度

没有限制，与以 NULL 结束的字符串相兼容。

- ShortString: 该类型是为了兼容 Pascal 的字符串类型，长度限制在 255 个字符内。
- WideString: 功能上类似于 AnsiString，但它由 WideChar 字符组成。
- PChar: 指向以 NULL 结束的字符串的指针。

缺省情况下，如果用“String”来定义字符串，编译器默认为 AnsiChar。加上编译指令{\$H-}时，String 为 ShortString。

4. 记录类型

记录是一组不同类型数据的集合，其定义格式如下：

```
type
    记录类型标识符 = record
        域 1: 类型 1;
        域 2: 类型 2;
        .....
        域 n: 类型 n;
    end;
```

每个数据元素称为“域”。访问记录域，可以使用“变量名.域名”的形式，也可以使用开域语句 with 来直接访问记录的域。

5. 指针类型

指针类型的变量存储内存空间的地址。指针所指单元可以是任何类型数据，通过指针可以灵活地访问内存中的数据。指针类型的声明格式如下：

```
type 指针类型标识符 = ^数据类型;
```

6. 可变类型

可变类型是指在程序运行期间确定且可以改变的数据类型，相当于 C++ 的无类型(Void)。这些数据在编译期间不能确定数据类型，比固定类型的数据占用更多的存储空间和操作时间。

默认情况下，可变类型可以是除了结构类型和指针类型之外的任何其他类型。可变类型的标识符是 Variant。

2.5 运算符

运算符是对各种数据类型进行运算的符号。例如，有能进行加、减、乘、除的运算符，有能访问一个数组的某个单元地址的运算符等。

按照操作数数目来分，运算符可以分为单目运算符和双目运算符。

有些运算符要根据给定的操作数的数据类型来做相应的处理。例如，运算符 not 对于整型的操作数来说，表示按位取反；对于逻辑类型的操作数来说，表示逻辑非。

1. 赋值运算符

给一个变量赋值，用“:=”运算符，形式如下：

```
变量 := 表达式;
```

2. 算术运算符

算术运算符的操作数是整数或实数，见表 2-3。

表 2-3 Object Pascal 语言中的算术运算符

运算符	作 用	操作数类型	结果类型
+	正号运算符，单目运算	整数或实数	整数或实数
	两个整数或实数相加		
-	负号运算符，单目运算，求负值	整数或实数	整数或实数
	两个整数或实数相减		
*	两个整数或实数相乘	整数或实数	整数或实数
/	两个整数或实数相除	整数或实数	实数
div	两个整数整除，结果取商的整数部分	整数	整数
mod	两个整数相除，取余数，余数的符号与被除数相同	整数	整数

3. 关系运算符

关系运算符用来比较两个操作数，结果为布尔型数据。关系运算符见表 2-4。

表 2-4 Object Pascal 语言中的关系运算符

运算符	作 用	结果类型
=	比较两个操作数是否相等，相等结果为 True，否则为 False	布尔型
<>	比较两个操作数是否不等，不等结果为 True，否则为 False	布尔型
>	左操作数大于右操作数，则结果为 True，否则为 False	布尔型
>=	左操作数大于或等于右操作数，则结果为 True，否则为 False	布尔型
<	左操作数小于右操作数，则结果为 True，否则为 False	布尔型
<=	左操作数小于或等于右操作数，则结果为 True，否则为 False	布尔型

4. 逻辑运算符

逻辑运算符有 not、and、or、xor 四个，逻辑运算的操作数都是布尔型数据，结果也为布尔型。逻辑运算的含义见表 2-5。

表 2-5 逻辑运算表

A	B	not A	A and B	A or B	A xor B
False	False	True	False	False	False
False	True	True	False	True	True
True	False	False	False	True	True
True	True	False	True	True	False

5. 日期时间运算符

日期时间运算符见表 2-6。此外，关系运算符也适用于日期时间型数据。

表 2-6 Object Pascal 语言中的日期时间运算符

运算符	作 用	操作数类型	结果类型
+	D+N: 表示当前日期时间之后 N 天的日期时间	日期时间型+整型或实型	日期时间型
-	D-N: 表示当前日期时间之前 N 天的日期时间	日期时间型-整型或实型	日期时间型
	D1-D2: 表示两个日期时间的间隔	日期时间型-日期时间型	整型或实型

6. 位运算符

位运算符见表 2-7。

表 2-7 Object Pascal 语言中的位运算符

运算符	作 用	操作数类型	结果类型
shl	A shl n: 表示将整型数 A 的各位向左移 n 位, 高位溢出, 低位补 0	整数	整数
shr	A shr n: 表示将整型数 A 的各位向右移 n 位, 低位溢出, 高位补 0	整数	整数
not	逐位取反, 单目运算。其规则为: 0 取反为 1, 1 取反为 0	整数	整数
and	逐位相与, 双目运算。其规则为: 0 与 0 的值为 0, 0 与 1 的值为 0, 1 与 0 的值为 0, 1 与 1 的值为 1	整数	整数
or	逐位相或, 双目运算。其规则为: 0 或 0 的值为 0, 0 或 1 的值为 1, 1 或 0 的值为 1, 1 或 1 的值为 1	整数	整数
xor	逐位相异或, 双目运算。其规则为: 0 异或 0 的值为 0, 0 异或 1 的值为 1, 1 异或 0 的值为 1, 1 异或 1 的值为 0	整数	整数

7. 集合运算符

集合运算主要是判断两个集合之间的关系。集合运算符见表 2-8。

表 2-8 Object Pascal 语言中的集合运算符

运算符	作 用	操作数类型	结果类型
+	集合并	集合型	集合型
-	集合差	集合型	集合型
*	集合交	集合型	集合型
<=	左集合是否包含于右集合	集合型	布尔型
=	左集合是否等于右集合	集合型	布尔型
>=	左集合是否包含右集合	集合型	布尔型
<>	左集合是否不等于右集合	集合型	布尔型
in	左集合是否与右集合有从属关系	集合型	布尔型