Visual Basic

程序设计简明教程

第 二 版

Visual Basic 程序设计简明教程

第一章 Visual Basic程序设计概述

（1学时）

1.1 引例

1.2 VB主要功能和特点

1.3 VB集成开发环境

1.1 引例1.1

一行内容在窗体左、右移动；

移动两种方法:手动和自动，当自动时，文字闪烁显示；

当内容超出窗体,进行反弹。

1.1 引例 2.1

对输入的字符进行转换的程序。

转换规则：

- 大写字母转换成小写字母，
- 小写字母转换成大写字母
- 空格不转换
- 其余转换成“*”。

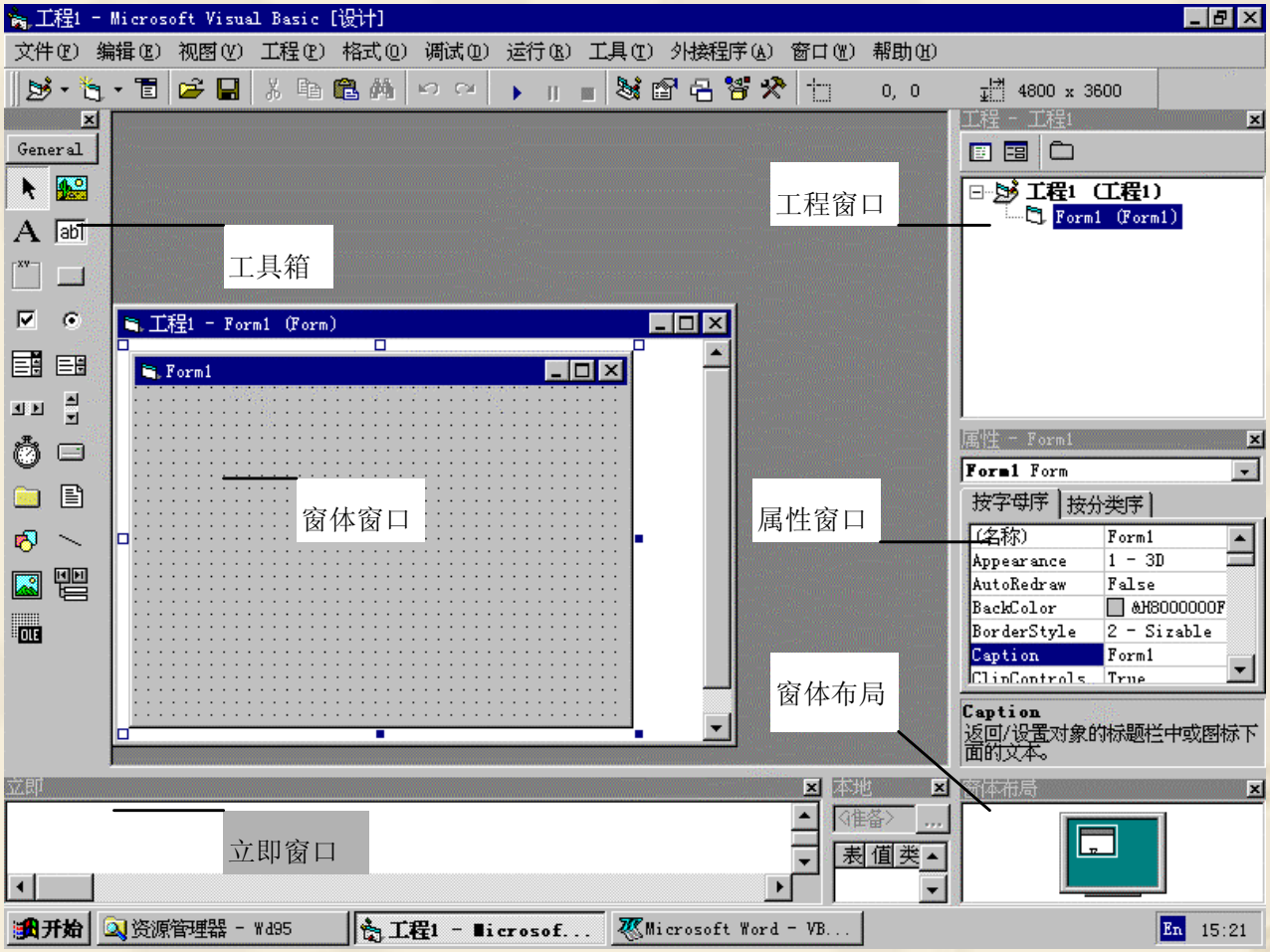
要求：每输入一个字符，马上就进行判断和转换。

通过上述两例说明VB面向对象、可视化、事件驱动的特点。

1.2 VB主要功能和特点

1. 具有面向对象的可视化设计工具;
2. 事件驱动的编程机制;
3. 提供了易学易用的应用程序集成开发环境;
4. 结构化的程序设计语言;
5. 支持多种数据库系统的访问;
6. Active技术;
7. VB 6.0在开发环境上、网络功能等的增强;
8. 完备的help联机帮助功能。

1.3 VB集成开发环境


工程1 - Microsoft Visual Basic [设计]
文件(F) 编辑(E) 视图(V) 工程(P) 格式(O) 调试(D) 运行(R) 工具(T) 外接程序(A) 窗口(W) 帮助(H)
General
工具箱
工程窗口
工程1 - Form1 (Form)
Form1
窗体窗口
属性窗口
窗体布局
立即窗口
工程 - 工程1
工程1 (工程1)
Form1 (Form1)
属性 - Form1
Form1 Form
按字母序
按分类序
(名称) Form1
Appearance 1 - 3D
AutoRedraw False
BackColor &H8000000F
BorderStyle 2 - Sizable
Caption Form1
ClipControls True
Caption
返回/设置对象的标题栏中或图标下面的文本。
立即
本地
窗体布局
开始
资源管理器 - Wd95
工程1 - Microsof...
Microsoft Word - VB...
En 15:21

1. **主窗口**

应用程序窗口，由标题栏、菜单栏和工具栏组成。

2. **窗体(form)窗口**

设计VB程序的界面。

3. **代码(code)窗口**

编辑窗体、标准模块中的代码。

4. **属性(properties)窗口**

所有窗体或控件的属性设置。

5. **工程资源管理器(project expror)窗口**

保存一个应用程序所有的文件。

6. **工具箱(toolbox)窗口**

显示各种控件的制作工具，供用户在窗体上设计。

第二章 VB简单的程序设计（3学时）

2.1 程序设计方法的发展

1. 初期的程序设计

高运行效率、少占用内存为目标。

2. 结构化程序设计

程序的可读性、可维护性为目标。

程序= 算法+数据结构 的面向过程的程序设计。

3. 面向对象的程序设计

降低程序的复杂性、提高软件的开法效率和改善工作界面为目标。

程序=对象+消息 的面向对象的程序设计。

2.2 对象的有关概念

1. 类

类是创建对象实例的模板，包含了创建对象的属性描述和行为特征的定义。

2. 对象

是类的一个实例，继承了类的属性、方法。

VB中常用的对象有：窗体、控件等。

3. 对象的建立和命名

4. 对象的三要素

（1）属性

对象的性质，即用来描述和反映对象特征的参数。

（2）方法

对象的行为，即将一些通用的过程编写好并封装起来，作为方法供用户直接调用。

（3）事件

响应对象的动作称为事件，它发生在用户与应用程序交互时。如单击控件、鼠标移动、键盘按下等。

- 事件过程

指附在该对象上的用户编写的程序代码,是事件的处理程序。

- 事件驱动过程

是图形用户界面的本质，由用户控制而代码作出响应。

5. VB程序的执行步骤如下：

（1）启动应用程序，装载和显示窗体；

（2）窗体（或窗体上的控件）等待事件的发生；

（3）事件发生时，执行对应的事件过程；

（4）重复执行步骤（2）和（3）；

(5) 直到遇到END结束语句结束程序的运行；

或按“结束”强行停止程序的运行。

2.3 建立简单的应用程序

以例2.1为例：

1. 建立用户界面的对象；
2. 对象属性的设置；
3. 对象事件过程及编程；
4. 程序运行和调试；
5. 保存文件。

2.4 基本控件和窗体

一、常用属性

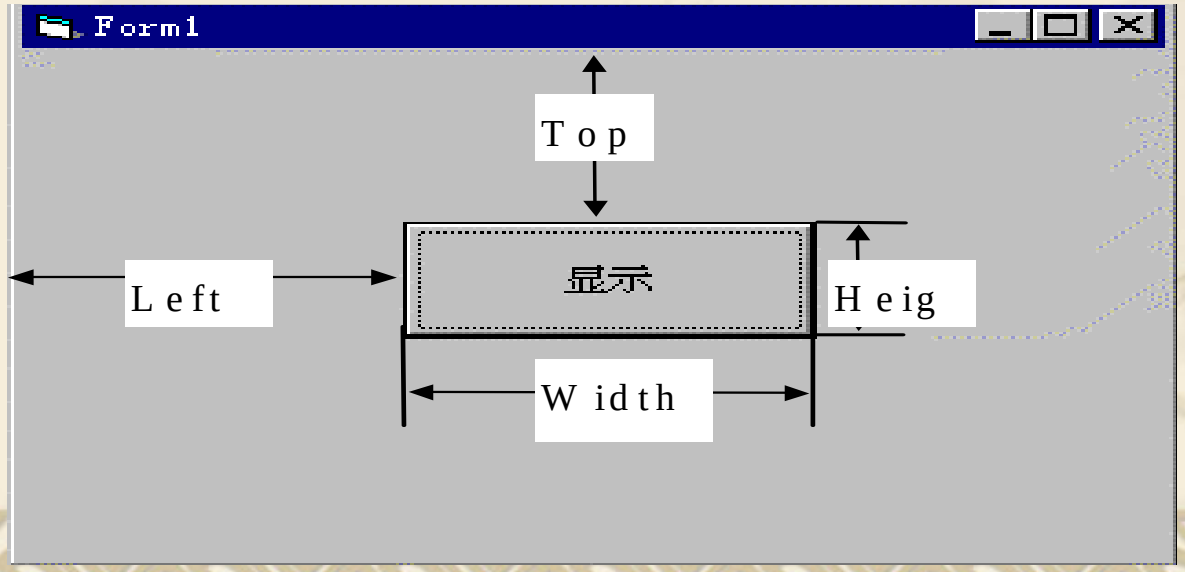
1. Name名称属性

创建的对象名称，有默认的名。在程序中，控件名是作为对象的标识而引用，不会显示在窗体上。

2. Caption标题属性

该属性决定了控件上显示的内容。

3. Height、width、top和left属性



4. Enabled属性

控件是否可操作。当设置为false时，呈暗淡色，禁止用户进行操作。

5. Visible属性

控件是否可见。当设置为false时，用户看不到，但控件本身存在。

6. Font属性

Fontname：字体；Fontsize：字体大小；Fontbold：是否是粗体；

Fontitalic：是否斜体；Fontstrikethru：

Fontunderline：是否带下划线。

例2.2 在窗体是建立两个命令按钮Command1和Command2，Commanc的Font通过属性窗口设置，其余属性通过代码实现

7. Forecolor前景颜色属性

设置控件的前景颜色（即正文颜色).。其值是一个十六进制常数，用户可以在调色板中直接选择所需颜色。

8. Backcolor背景颜色属性

9. Backstyle背景风格属性

0-transparent：透明显示，即控件背景颜色显示不出来。

10. BorderStyle边框风格属性

0-None：控件周围没有边框 。

1-Fixed Single：控件带有单边框。

11. Alignment属性

控件上正文水平对齐方式

0：正文左对齐；1：右对齐；2：居中。

12. AutoSize属性

控件是否根据正文自动调整大小， False

13. WordWarp属性

AutoSize True时，WordWarp才有效。

按正文字体大小在垂直方向上改变显示区域的大小。

14. TabIndex属性

决定了按Tab键时，焦点在各个控件移动的顺序。各个控件默认TabIndex值就是以控件建立时 的顺序，第一个为0。

15. 控件默认属性

反映该控件最重要的属性，使用时可省略属性名。

注意：Command的默认属性为Default,当该属性为True，当按Enter键，该控件起作用。

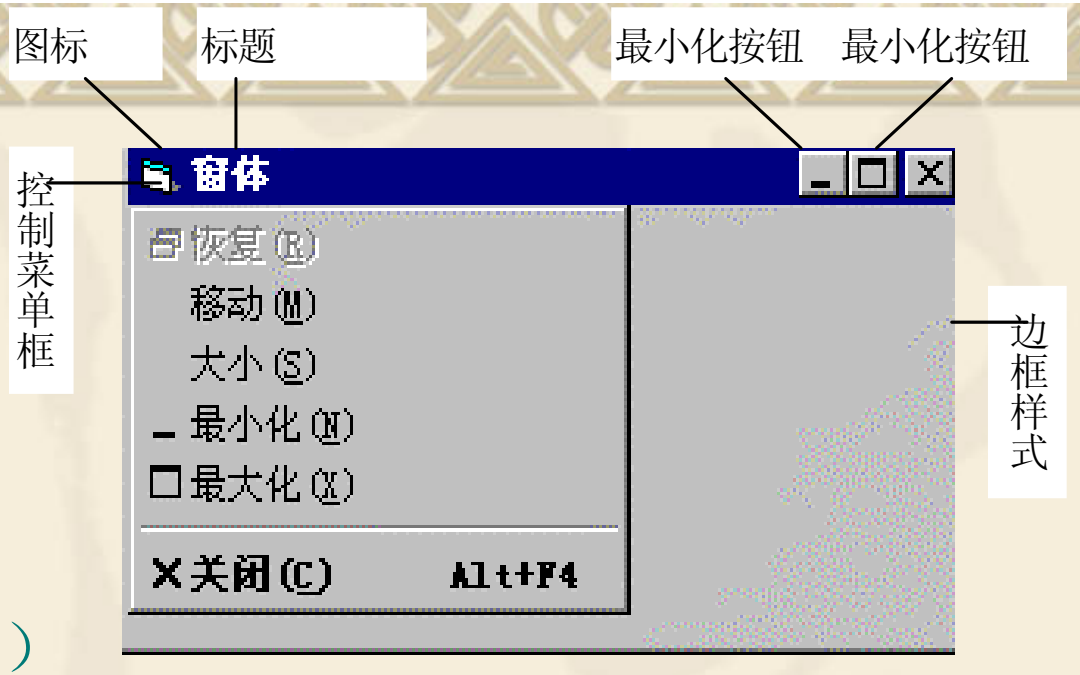
例2.3

控件设置

默认控件名 Name	标题 Caption	有关属性设置
Form1	颜色、对齐、鼠标属性例	MousePointer=99,MouseIcon=Key04.ico
Label1	左对齐	Alignment=0， BorderStyle =1
Label2	居中	Alignment=1，BorderStyle =1
Label3	自动	AutoSize=True， WordWarp=False， BorderStyle =1
Label4	背景白	BackColor= &H00FFFFFF&， BorderStyle =0
Label5	前景红	ForeColor= &H000000FF&， BorderStyle =0

效果

二. 窗体



1. 属性

Caption标题

MaxButton、MinButton

Icon、ControlBox

Picture、AutoRedraw

BorderStyle （0 1 2 3 4 5 ）

WindowsState （0-正常、1-最小化、2-最大化）

2. 事件

Click、DblClick和Load

Load 当装入窗体时激发，通常用于对属性和变量初始化

3. 方法

Print、Cls和Move等

例2.4 上述属性、Click、DblClick和Load 事件演示。

三、标签 Label

用于显示文本(输出)信息，不能作为输入信息的界面。

例2.5 显示浮雕效果的文字，实现方法：字颜色、标签位置

四、文本框 TextBox

是一个文本编辑区域，可在该区域输入，编辑和显示正文内容。

1. 其他属性

- Text：正文内容；
- Maxlength：设置正文的最多字符个数，0任意长度值；
- MultiLine：是否为多行，默认为一行，False；
- ScrollBars：MultiLine为True时，该属性才有效，表示滚动条的形式
  0-None、1-Horizontal、2-Vertical、3-Both；
- Locked：是否可被编辑属性，False，表示为可编辑。
- PassWord：口令字符； Text属性返回输入的数据，屏幕显示该字符；
- SelStart、SelLength、SelText：选中文本的起始、长度、内容。

例2.6 复制选中的文本

2. 常用事件

- Change：当改变文本框的Text属性时会引发该事件。
- KeyPress(KeyAscii As Integer)：同上，并可返回一个KeyAscii参数 KeyAscii 为13，按回车键;为0 去除刚输入的字符。
- LostFocus：当控件失去焦点时发生。
- GotFocus：当控件获得焦点时发生。

3. 常用方法

[对象.]SetFocus：是把光标移到指定的文本框对象中。

4. 文本框的应用 数据过滤

例2.7 要求输入合法的数字数据。当输入结束时（按Tab键），对于输入正确的数据，显示正确信息；对输入的非数字数据，则响铃(Beep)、显示错误信息、清除文本框中的内容，并使焦点重新回到文本框。

- Text2_ LostFocus 当输入结束按Tab键时，该事件激发。
- IsNumeric(Text2)：判断是否输入非数字数据。
- Text2.SetFocus：使焦点重新回到文本框。

五、命令按钮CommandButton

接受用户输入的命令。输入命令可以有三种方式：

鼠标单击、Tab键焦点到该按钮、快捷键。

1. 其他属性

- Caption：命令按钮显示的内容，可设置快捷键，例 &Ok，显示Ok。
- Value： 检查该按钮是否按下。该属性在设计时无效。
- Picture ：按钮可显示图片文件(.bmp和.Ico)，当Style为1时有效。
- Style：确定显示的形式，0 只能显示文字，1 文字、图形均可 。
- ToolTipText ：设置工具提示，和Picture结合使用。

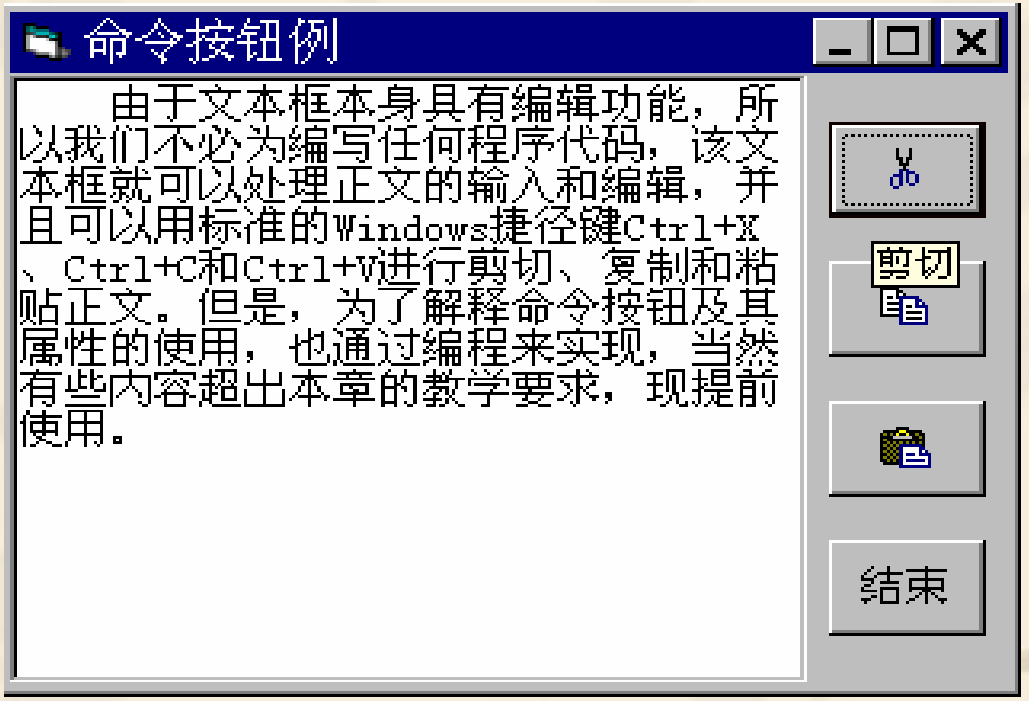
2. 事件

Click

3. 例 2.8

建立一个允许剪切、复制和粘贴的简单便笺板程序。

可增加内容：利用MouseMove事件决定命令按钮的有效性，即当选中内容，“剪切”、“复制”按钮有效，否则无效。



六、常用方法

方法是面向对象的，故使用的形式为： [对象.]方法

1. Print方法

形式： [对象.]Print[{Spc(n)|Tab(n)}][表达式列表][；|，]

作用：在对象上输出信息

对象：窗体、图形框或打印机(Printer)，省略对象在窗体上输出。

Spc(n)函数：插入*n*个空格，允许重复使用。

Tab(n)函数：左端开始右移动n列，允许重复使用。

；(分号)：光标定位上一个显示的字符后。

，(逗号)：光标定位在下一个打印区的开始位置处。

无；，时换行。

开始打印的位置是由对象的CurrentX和CurrentX属性决定，缺省为打印对象的左上角0，0。

注意：**Print**方法在**Form _Load**事件过程中起作用，必须设置窗体的**AutoRedraw**为**True**。

例2.9 显示图形

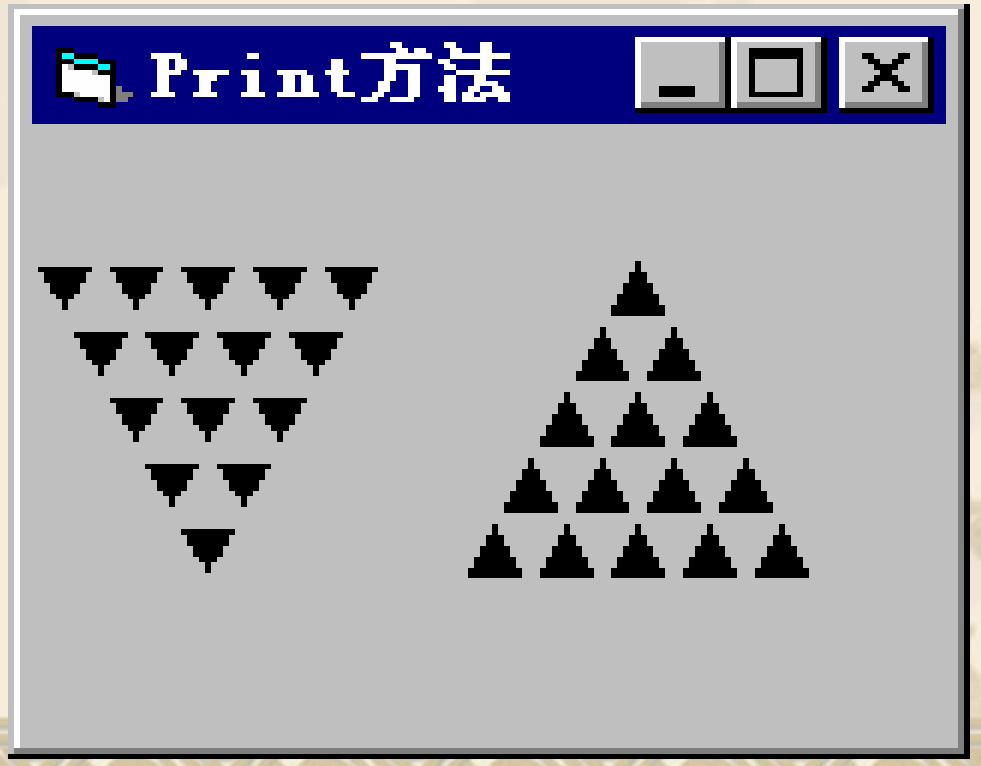
```
Private Sub Form_Click()
  For i = 1 To 5
    Print Tab(i); String(6 - I, "▼");    Spc(6); String(I, "▲")
  Next i
End Sub
```

注意：

String(*number*, *character*)

返回指定长度重复字符的字符串。

考虑：
若把Spc(6)换成Tab(6)，效果如何？



2. Cls方法

形式：[对象.]Cls

作用：清除运行时在窗体或图形框中显示的文本或图形。

注意：不清除在设计时的文本和图形。

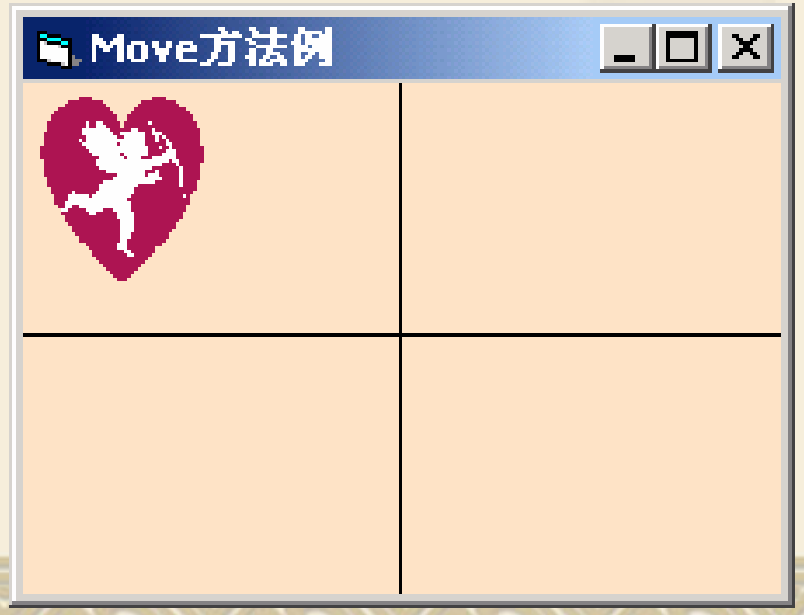
3. Move方法

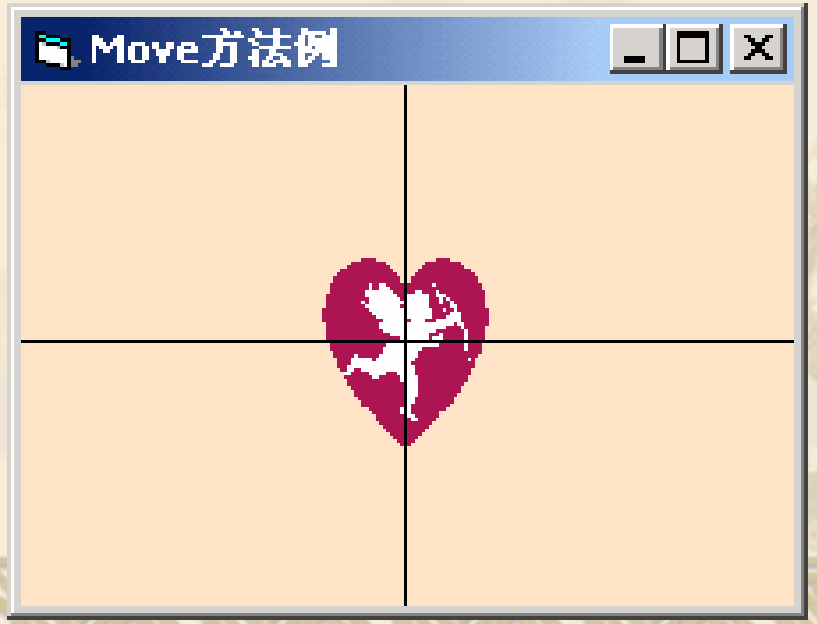
形式：[对象.]Move 左边距离[，上边距离[，宽度[，高度]]]

作用：移动窗体或控件，并可改变其大小。

对象：可以是窗体及除时钟、菜单外的所有控件

例2.10 图形移到窗体的中心。





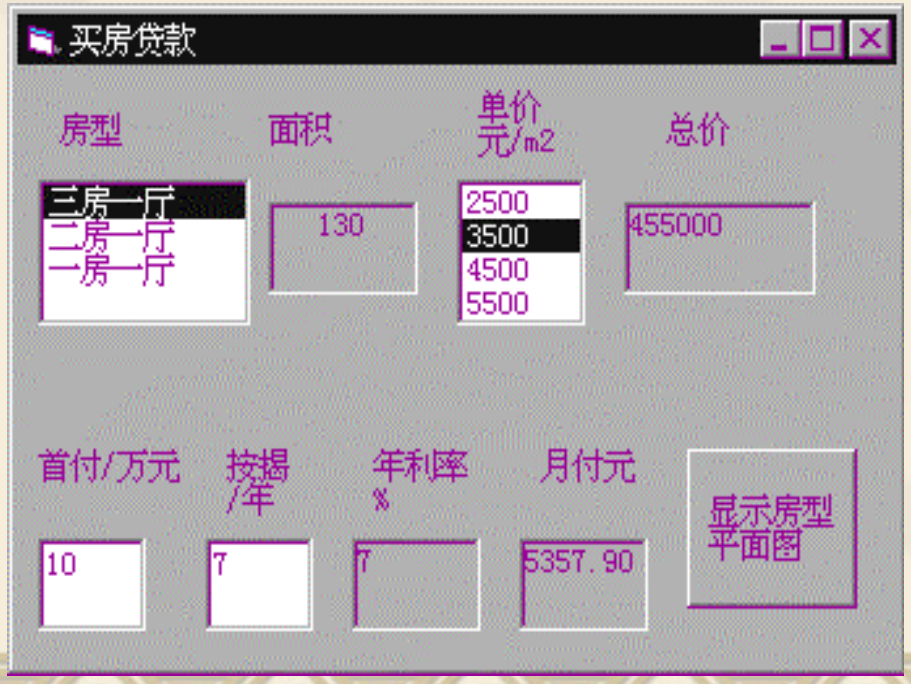
七、应用举例

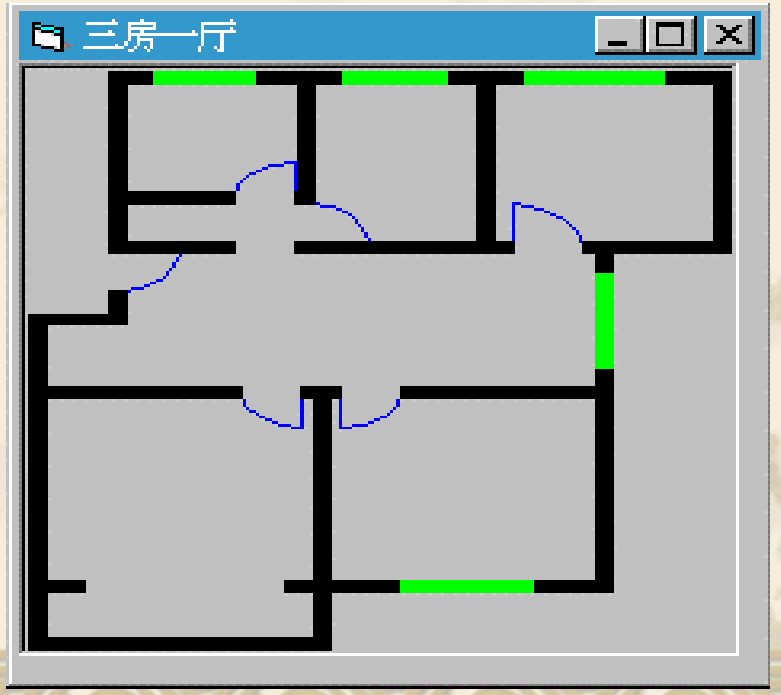
例2-11　编一计算月支付贷款的程序。现要求根据房产商提供的信息，买房者选择房型、面积、单价、按揭期等信息，计算每月支付代款的程序。

窗体1：列表框选择数据、标签显示数据、文本框输入数据；

窗体2：图形框显示房屋的平面图。

Pmt函数计算月支付贷款。





2.5 工程的管理及环境的设置

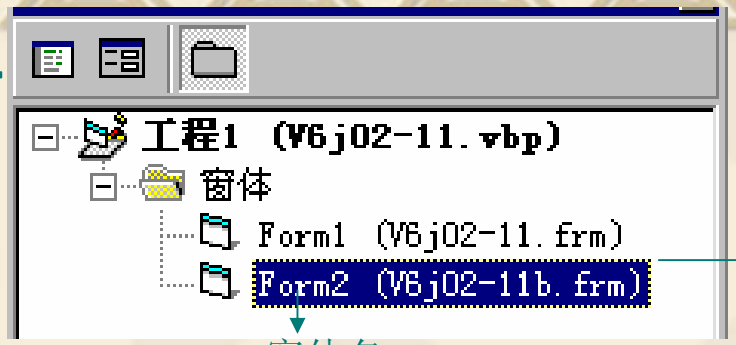
一、工程的组成

工程文件包含了一个应用程序的所有文件:

- 窗体文件(.frm)：控件及属性、事件过程和自定义过程;
- 窗体的二进制数据文件(.frx)，自动产生同名.frx文件;
- 标准模块文件(.bas);
- 类模块的文件(.cls);
- 资源文件(.res);
- ActiveX控件的文件(.ocx).

二、创建、打开和保存工程

三、添加、删除和保存文件



1. 窗体名和窗体文件名概念

窗体名：窗体的Name属性，在代码中用来引用该窗体，同一工程不能有相同的窗体名；

窗体文件名：窗体存放在磁盘上的文件名，该文件包含了该窗体的所有控件属性和代码，同一文件夹不能有相同的文件名。

2. 在工程中添加现存的窗体时，只是对该窗体文件的引用纳入工程。因此，如果更改窗体并保存它，会影响包含此窗体文件的任何工程。

3. 在工程中删除窗体时，仅将此窗体从工程里删除掉，但是窗体文件仍存在于磁盘上。但是，如果在VB之外删除一个窗体文件，VB不能更新此工程文件，当打开此工程时，将显示一个文件丢失的错误信息。

4. 对窗体文件改名方法

1）打开工程

“另存为…Form”实现文件的复制——保存工程文件。

2）利用编辑程序打开工程文件，修改FORM=XXX的文件名。

2.6 生成可执行文件和制作安装盘

以工程文件例2.11为例。

1. 生成可执行文件

文件/生成V6B2-11.exe文件，然后关闭Visual Basic6.0。

2. 制作安装盘

- 在向导的指导下运行V B6.0中文版工具“Package & Deployment”；
- “打包”：把工程中用到的各种类型的文件进行打包压缩后，存放到特定的目录下。
- “展开”：再把这些打包的文件展开到用户可以携带的安装介质软盘、光盘等。

3. 检测安装程序

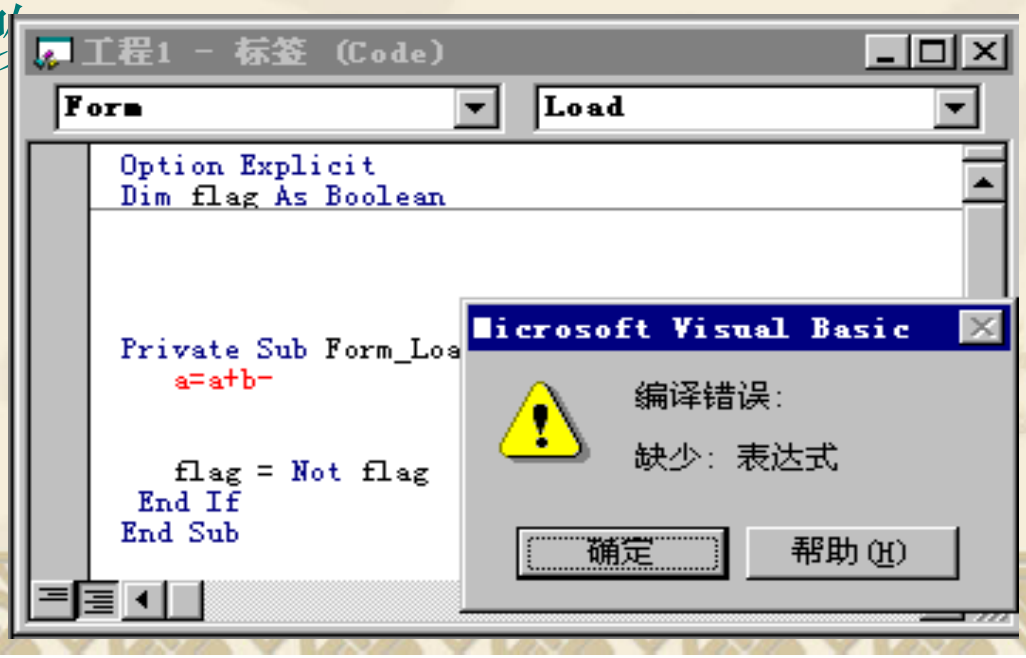
在没有VB 6.0系统的环境下，执行安装盘中的Setup.exe文件，将发行盘进行安装。

2.7 程序调试

一、错误类型

（1）编辑错误

在编辑代码时，VB会对键入的代码直接进行语法检查。当发现代码存在打字错误，遗漏关键字或标点符等语法错误，VB 在Form窗口中弹出一个子窗口，提示出错信息，出错的那一行变成红色。这时，用户必须单击“确定”按钮，关闭出错提示窗，然后对出错行进行修改。

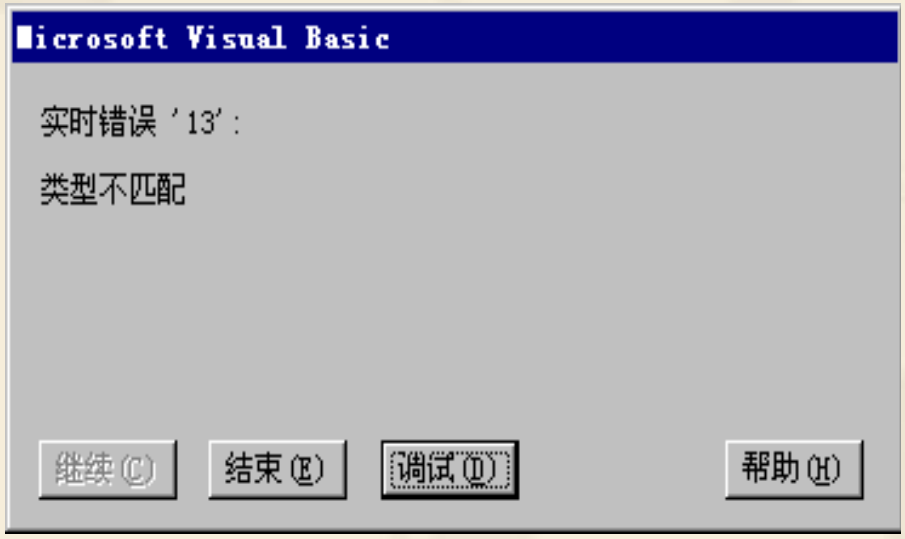


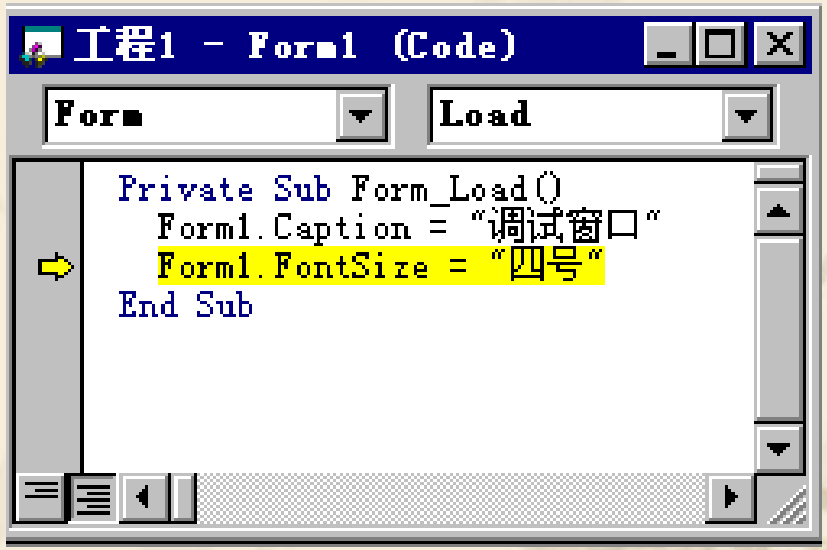
（2）编译错误

编译错误指按了“启动”按钮，VB开始运行程序前，先编译执行的程序段时，产生的错误。此类错误由于用户未定义变量、遗漏关键字等原因产生。这时，Visual Basic也弹出一个子窗口，提示出错信息，出错的那一行被高亮度显示。

（3） 运行错误

运行时错误指VB在编译通过后，运行代码时发生的错误。这类错误往往是指令代码执行了一非法操作引起的。例如类型不匹配、试图打开一个不存在的文件等。





（4） 逻辑错误

程序运行后，得不到所期望的结果，这说明程序存在逻辑错误。这类错误往往是程序存在逻辑上的缺陷所引起。例如，运算符使用不正确、语句的次序不对、循环语句的起始、终值不正确等。通常，逻辑错误不会产生错误提示信息，故错误较难排除，需要程序员仔细地阅读分析程序以及调试。

二、调试错误

(1) VB的三种模式

- 设计模式[设计]
  进行程序的界面设计、属性设置、代码编写等。
- 运行模式[运行]
  执行“运行/启动”命令，可以查看程序代码，但不能修改。
- 中断模式[中断]

当在运行模式，按了“中断”命令或当程序出现运行时错误时进入中断模式。可以查看代码、修改代码、检查数据。

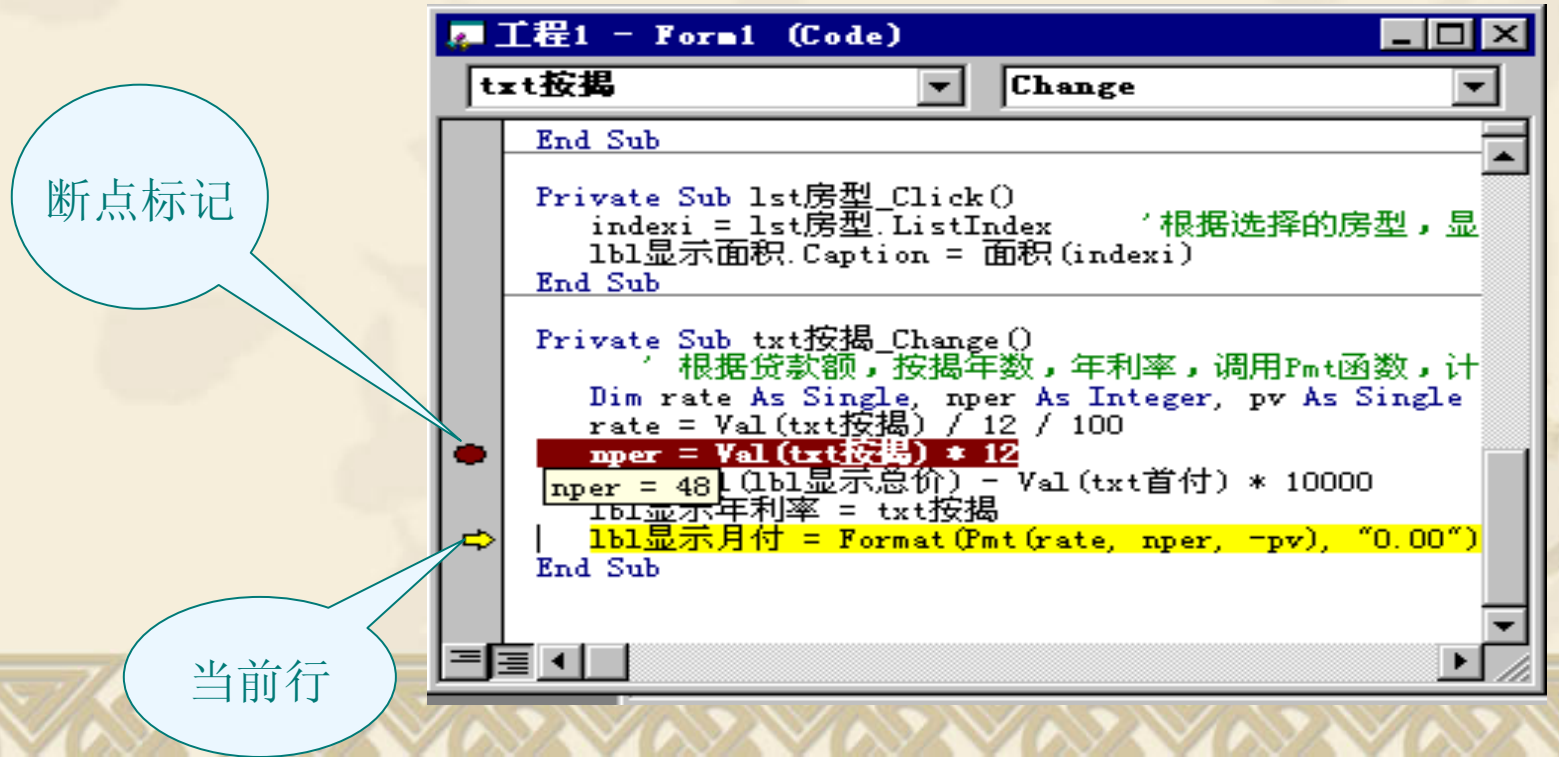
(2) 调试窗口

有三个调试窗口，它们是：“立即”窗口、“监视”窗口、和“本地”窗口。可单击视图菜单中的对应命令打开这些窗口。

（3）设置断点和逐语句跟踪

断点是告诉VB挂起程序执行的一个标记，当程序执行到断点处即暂停程序的运行，进入中断模式。设置或删除断点的步骤：

中断模式下，直接查看某个变量的值，只要把鼠标指向所关心的变量处，稍停一下，就在鼠标下方显示该变量的值。



2.8 常 见 错 误

1. 使用中文标点符号

系统产生“无效字符”，以红色显示。

2. 字母和数字形状相似

小写字母“l”和数字“1”形式相同、小写字母“o”与数字“0”。

3. 对象名称(Name)属性写错

Text1、Text2、Command1

txtInput、txtOutput、cmdOk

4. 对象的属性名、方法名、标准函数名写错

尽量使用自动列出成员功能。正确的系统按规定的大小写表示。

5. 无意形成控件数组

建立控件时小心使用“复制”、“粘贴”按钮。

6. 打开工程时找不到对应的文件

保存文件时先保存窗体.frm文件、再保存.vbp文件，注意路径。

第三章 VB语言基础

（2学时）

3.1 编 码 基 础

1. VB代码不区分字母的大小写

系统保留字自动转换每个单词的首字母大写；

用户自定义行以第一次为准。

2. 语句书写自由

一行可书写几句语句,之间用冒号分隔；

一句语句可分若干行书写，用续行符_连接一行<=255个字符。

3. 注释有利于程序的维护和调试

Rem开始 或 单撇号 '

4. 保留行号与标号

3.2 数 据 类 型

标准数据类型

表 1-3-1　　Visual Basic 的数据类型

数据类型	关键字	类型符	前缀	存贮大小(字节)	举例
字节型	Byte	无	b	1	125
逻辑型	Boolean	无	f	2	True　False
整型	Integer	%	i	2	－32768　32767
长整型	Long	&	l	4	－2123456677
单精度型	Single	!	s	4	－3.4E19　1.4E-10
双精度型	Double	#	dbl	8	－1.75686267D36　1.123456789
货币型	Currency	@	c	8	$12.345
日期型	Date	无	dt	8	03/25/1999
字符型	String	$	str	字符串	“abcdefg”
对象型	Object	无	对象	4	Command
变体型	Variant	无	v	按需分配	任一值

3.3 变量与常量

一、变量和常量的命名规则

变量：在程序运行中其存储的值可以改变。

常量：在程序运行中其值不可以改变。

命名规则：

1. 以字母或汉字开头，后可跟汉字、字母、数字或下划线组成，长度小于等于255个字符；

2. 不要使用VB中的关键字；

3. VB中不区分变量名的大小写；

4. 为了增加程序的可读性，可在变量名前加一个缩写的前缀来表明该变量的数据类型。

二、变量声明

1. 用Dim语句显式声明变量

形式：Dim 变量名 [AS 类型]

Dim 变量名类型符

例　Dim iCount As integer , sAllsum As single

等价于　Dim iCount%, sAllsum!

2. 隐式声明

未进行上述的声明而直接使用，其类型为 Variant 类型。

建议不使用。

注意：在通用声明处加 Option Explicit 语句可强制显式声明变量。

三、常量

1. 用户声明常量

形式：

Const 常量名 [AS 类型] = 表达式

省略 [AS 类型]，常量的类型由表达式值的类型决定。

为使与变量名区分，一般常量名使用大写字母。

例 Const MAX=100

2. 系统提供的常量

系统定义常量位于对象库中，可通过“对象浏览器”查看。

例：vbNormal vbMinimized、vbbCrLf等

3.4 运算符和表达式

一、运算符

1. 算术运算符

例：5+10 mod 10 \ 9 / 3 +2 ^2　结果：10

运算符	优先级	例	结　果
^	1	ia^2	9
—	2	— ia	— 3
*	3	ia*ia*ia	27
/	3	10/ia	3.3333333333
\	4	10\ia	3
Mod	5	10 Mod ia	1
+	6	10 + ia	13
—	6	ia— 10	— 7

2. 字符串运算符

& 、+ 字符串连接

" 123 " + " 456 " 结果 " 123456 "

" 123 " & " 456 " 结果 " 123456 "

区别：+ 两边必须是字符串，& 不一定

例如：

```
"abcdef" & 12345          ' 结果为 "abcdef12345 "
"abcdef " + 12345         ' 出错
"123" & 456               ' 结果为" 123456 "
“123” + 456               ' 结果为 579
```

注意：

```
"123 " + True             ' 结果为 122
```

True转换为数值-1，False转换为数值0

3. 关系运算符

将两个操作数进行大小比较，结果为逻辑量。

字符串比较,则按字符的ASCII码值从左到右一一比较，直到出现不同的字符为止.。

例：" ABCDE " > " ABRA "　　结果为 False

"男字" > "女字" 按汉字的拼音字母比较

运算符	例	结　果
=	"ABCDE" = "ABR"	False
>	"ABCDE" > "ABR"	False
>=	"bc" >="abcdef"	True
<	23<3	False
<=	"23"<"3"	True
<>	"abc"<>"ABC"	True

4. 逻辑运算符

将操作数进行逻辑运算，结果是逻辑值：

条件表达式1 And 条件表达式2 条件表达式均为T， 结果为T；

条件表达式1 Or 条件表达式2 条件表达式有一个为T 结果为

运算符	说明	优先级	说　明	例	结果
Not	取反	1	当操作数为假时，结果为真	Not F	T
And	与	2	操作数均为真时，结果才为真	T And F T And T	F T
Or	或	3	操作数中有一个为真时，结果为真	T Or F F Or F	T F
Xor	异或	3	操作数相反时，结果才为真	T Xor F T Xor T	T F

二、表达式

1. 组成

变量、常量、函数、运算符和圆括号。

2.书写规则

(1) 运算符不能相邻。例**a+ -b**是错误的。

(2) 乘号不能省略。例x 乘以y 应写成：x*y。

(3) 括号必须成对出现，均使用圆括号。

(4) 表达式从左到右在同一基准上书写，无高低、大小。

3. 不同数据类型的转换

运算结果的数据类型向精度高的数据类型靠。

Integer<Long<Single<Double<Currency

4. 优先级

算术运算符>=字符运算符>关系运算符>逻辑运算

表达式书写举例

$$\frac{abcd}{efg}$$

a*b*c*d/e/f/g　　或　　a*b *c*d/（e*f*g）

$$\sin 45^\circ + \frac{e^{10} + \ln 10}{\sqrt{x + y + 1}}$$

sin(45*3.14/180)+(exp(10)+log(10))/sqr(x+y+1)

例：用人单位招聘秘书：年龄小于40岁的女性，学历专科或本科，

年龄<40，性别=“女”，(学历=“专科”，学历=“本科”)

And　　And　　Or

考虑：若分别写成：

年龄<40 And 性别= "女" And (学历= "专科" And 学历= "本科")

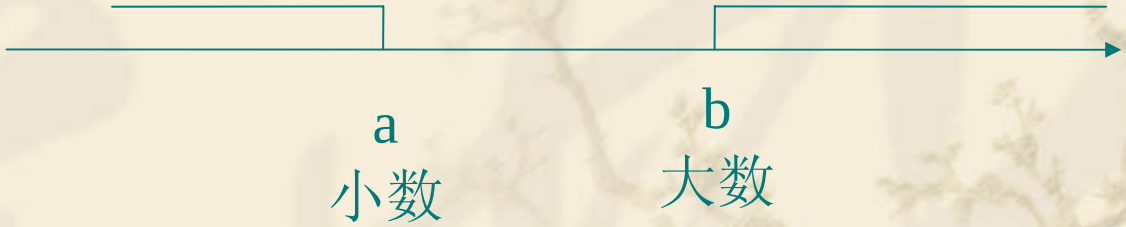
年龄<40 Or 性别= "女" Or (学历= "专科" Or 学历= "本科")

例：表示算术表达式：　$3 \le x \le 7$

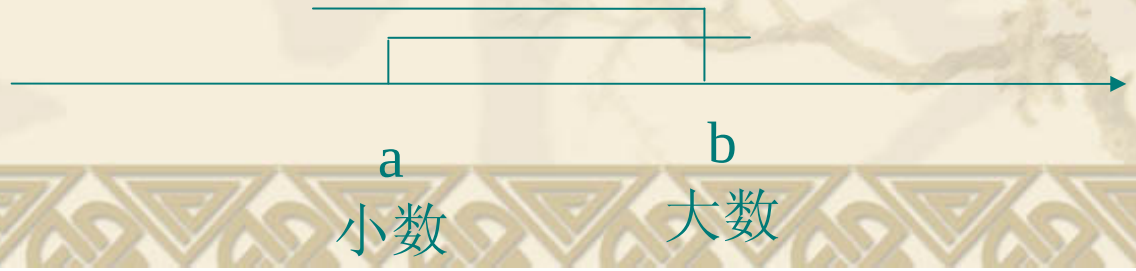
正确的VB表达式：　3〈=x And x <=7

错误的VB表达式：　3<=x<=7　　或 3<=x Or x<=7

比a小，比b大 用 Or



比a大，比b小 用 And



3.5 常用内部函数

1. 数学函数

Rnd函数返回0和1（包括0但不包括1）之间的双精度随机数。

每次运行时，要产生不同序列的随机数，执行Randomize 语句。

例产生1~100的随机数： Int(Rnd *100)+1

2. 转换函数

Chr(66) → "B"　　Asc ("B") → 66

Asc(Chr(99)) → 99　　Chr(Asc("K")) → "K"

Ucase$("abcdefg") → "ABCDEFG"

3. 日期函数

Time返回系统时间、**Date**返回系统日期、**Now**返回系统日期和时间

DateAdd（要增减日期形式，增减量，要增减的日期变量）

DateDiff（要间隔日期形式，日期1，日期2）

例如求离你毕业还有多少天：

DateDiff（"d",date, #2006/07/01# ）假定2006年毕业

4. 字符串编码和函数

(1)字符串编码

Windows系统对字符采用了DBCS编码，中文2字节，西文1字节

VB中字符Unicode编码，任何字符2字节。

Len(“中国good ”)为6；LenB(“中国good ”)为12。

StrConv() 进行DBCS与Unicode之间转换。

(2)字符串函数

Mid$("ABCDEFG",2,3) ⟹ "BCD"

InStr(2, "ABCDEFG", "EF") ⟹ 5

S=Split("123,56,ab",",") ⟹ S(0)="123”,S(1)="56”,S(2)="ab"

Join(S,"") ⟹ "123 56 ab"

Replace("ABCDABCD","CD","123") ⟹ “AB123AB123”

5. 格式输出函数

例3.1 利用Format函数显示有关的日期和时间。


```
Private Sub Form_Click( )
    FontSize = 12
    MyTime = #9:21:30 PM#
    MyDate = #7/21/1997#
    Print Tab(2); Format(MyDate, "m/d/yy")
    Print Tab(2); Format(MyDate, "mmmm-yy")
    Print Tab(2); Format(MyTime, "h-m-s AM/PM")
    Print Tab(2); Format(MyTime, "hh:mm:ss A/P")
    Print Tab(2); Format(Date, "dddd,mmmm,dd,yyyy")
    Print Tab(2); Format(Now, "yyyy年m月dd日 hh：
      mm")
    Print FormatDateTime(Now)  ' VB6.0新提供的函数
End Sub
```

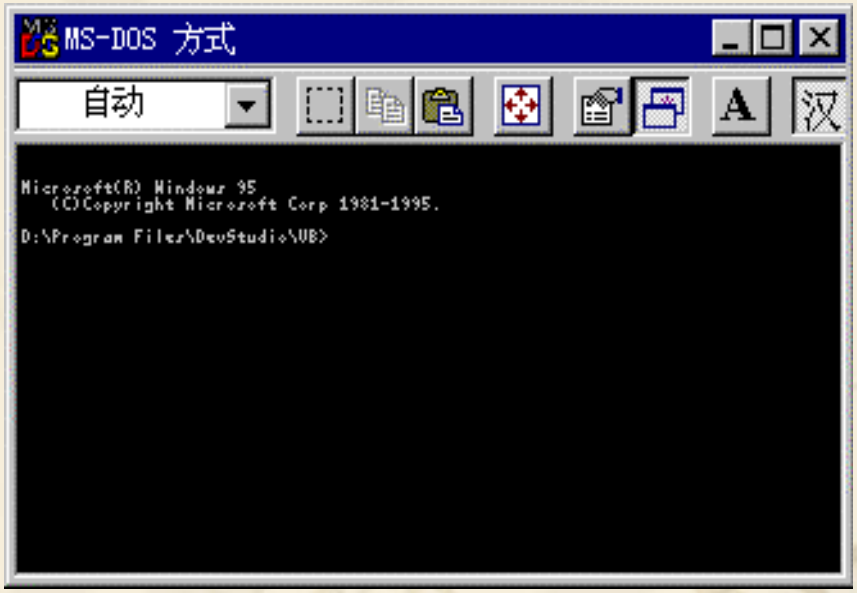
6. Shell函数

Shell函数负责执行一个可执行文件，返回一个Variant，如果成功，代表这个程序的进程ID，若不成功，则会返回0。

形式：Shell(命令字符串,窗口类型)

窗口类型值为1表示正常窗口，缺省窗口最小化为图标。

i = Shell("c:\windows\calc.exe")　　j = Shell("c:\command.com", 1)



提示：如果不知道应用程序的路径名，可用Start命令启动程序

i = Shell(start & "calc.exe")　；　i = Shell(start & " vb6.exe ")

3.6 常 见 错 误

1. 逻辑表达式书写错，在VB没有造成语法错而形成逻辑错

例如，数学表达式 $3 \leqslant x < 10$ VB表达式写为 3<=x<10。

问题在于VB中的逻辑量与数值量可相互转换。

2. 同时给多个变量赋值，在VB没有造成语法错而形成逻辑错

例如：Dim x%，y%，z%

x=y=z=1

3. 标准函数名写错

4. 变量名写错

检查方法：在通用声明段加Option Explicit

5. 语句书写位置错

在通用声明段只能有Dim语句，不能有赋值等其他语句

第四章　基本的控制结构

（8学时）

4.1　顺序结构

4.2　选择结构

4.3　循环结构

4.4　其他辅助控制语句

4.5　常用算法（一）

4.6　常见错误　（顺序、条件、循环）

4.1 顺 序 结 构

一、赋值语句

形式： [LET] 变量名 = 表达式

作用： 将表达式的值赋值给变量名。

一般用于给变量赋值或对控件设定属性值。

例： sRate!=0.1

Text1.Text ＝ "欢迎使用Visual Basic 6.0 "

当表达式的类型与变量的类型不一致时，强制转换成左边的精度，如

iA% = 10 / 3 iA中的结果为3

注意：虽然赋值号与关系运算符等于号都用“=”表示，VB系统会根据所处的位置自动判断是何种意义的符号。

二、与用户交互函数和过程

1. InputBox函数

InputBox(提示[，标题][，缺省][，x 坐标位置][，y坐标位置])

其中：提示： 提示信息；标题： 对话框标题；缺省：输入区缺省值

函数返回字符类型。

例要在屏幕上显示图示的对话框：相应的语句如下：

```
Dim  strName  As  String * 40
strName= InputBox("请输入你的姓名" + vbCrLf + "然后单击确定" , "输入框" )
```

也可以使用如下语句：

```
Dim strName  As  String * 40，  strS1  As  String * 40
strS1 = "请输入你的姓名" + Chr(13) + Chr(10) + "然后单击确定"
strName= InputBox(strS1，  "输入框" ，  ，  100，100)
```

当键盘输入“王晓明” 后，变量strName获得键盘输入的值。

2. MsgBox函数和MsgBox过程

函数形式：变量[%] = MsgBox(提示[,按钮][，标题])

过程形式：MsgBox 提示[,按钮][,标题]

按钮值如下：

分组	内部常数	按钮值	描　述
按钮数目	vbOkOnly	0	只显示 Ok 按钮
	vbOkCancel	1	显示 Ok，Cancel 按钮
	vbAboutRetryIgnore	2	显示 About，Retry，Ignore 按钮
	vbYesNoCancel	3	显示 Yes，No，Cancel 按钮
	vbYesNo	4	显示 Yes，No 按钮
	vbRetryCancel	5	显示 Retry，Cancel 按钮
图标类型	vbCritical	16	关键信息图标 红色 STOP 标志
	vbQuestion	32	询问信息图标 ？
	vbExclamation	48	警告信息图标 ！
	vbInformation	64	信息图标　　i

函数返回所选按钮整数

内部常数	返回值	被按下的按钮
vbOk	1	Ok
vbCancel	2	Cancel
vbAbout	3	About
vbRetry	4	Retry
vbIgnore	5	Ignore
vbYes	6	Yes
vbNo	7	No

例4.1 编一帐号和密码检验程序。

要求：

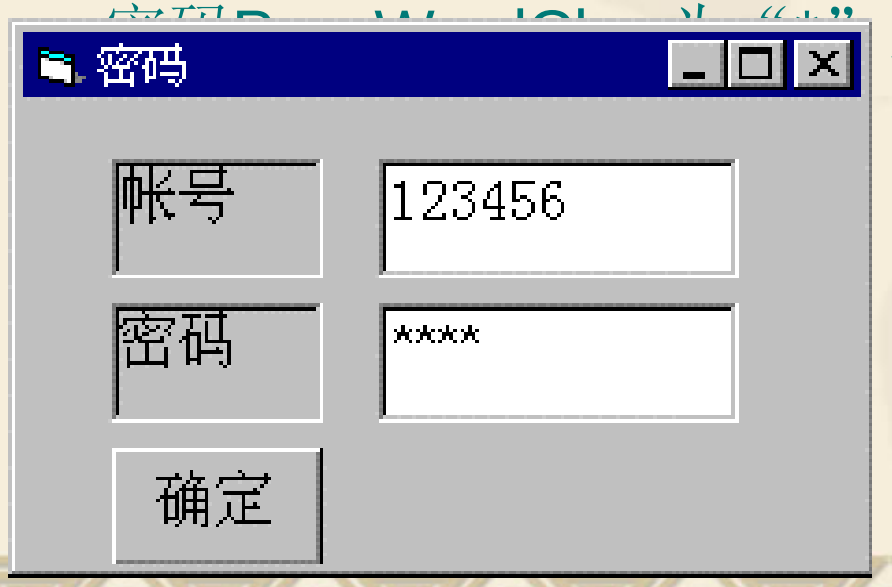
帐号不超过6位数字，有错，清除原内容再输入。

密码输入时在屏幕上以“*”代替； 若密码错，显示有关信息，选择“重试”按钮，清除原内容再输入，选择“取消”按钮，停止运行。

分析：

帐号6位，MaxLength为6，LostFocus判断数字IsNumeric函数

，MsgBox函数设置密码错对话

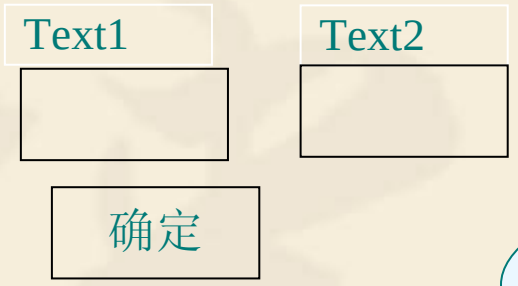


常 见 错 误

数据合法性检查中引起程序的死循环

```
Private Sub Text1_LostFocus()
 If Not IsNumeric(Text1) Then
    Text1 = ""
    Text1.SetFocus
  End If
End Sub
```

```
Private Sub Text2_LostFocus()
 If Not IsNumeric(Text2) Then
    Text2 = ""
    Text1.SetFocus
  End If
End Sub
```



If Text2 <> "" And Not IsNumeric(Text2)

4.2 选 择 结 构

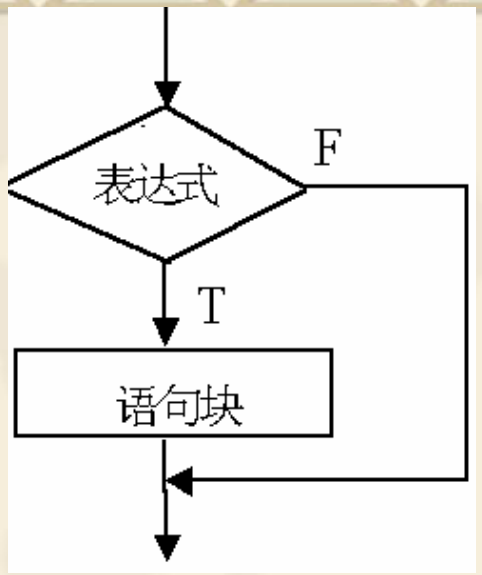
1. If…Then语句(单分支结构)

If <表达式> Then

语句块

End If

或 **If <表达式> Then <语句>**

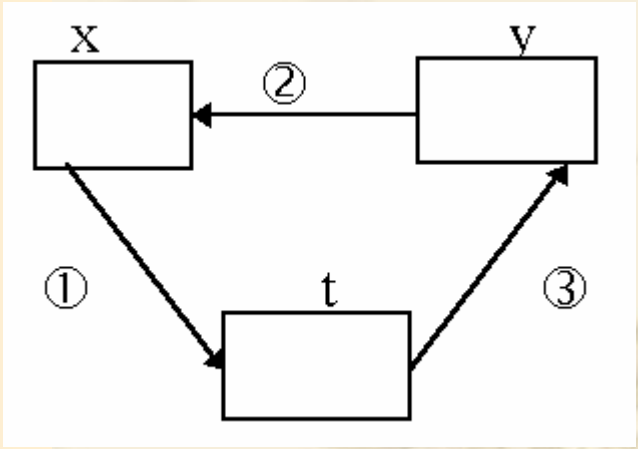


例：已知两个数*x*和*y*，比较它们的大小，

使得x大于y.

```
If  x<y Then
   t=x
   x=y
   y=t
End If
```

或 If x<y Then x=x+y:y=x-y:x=x-y

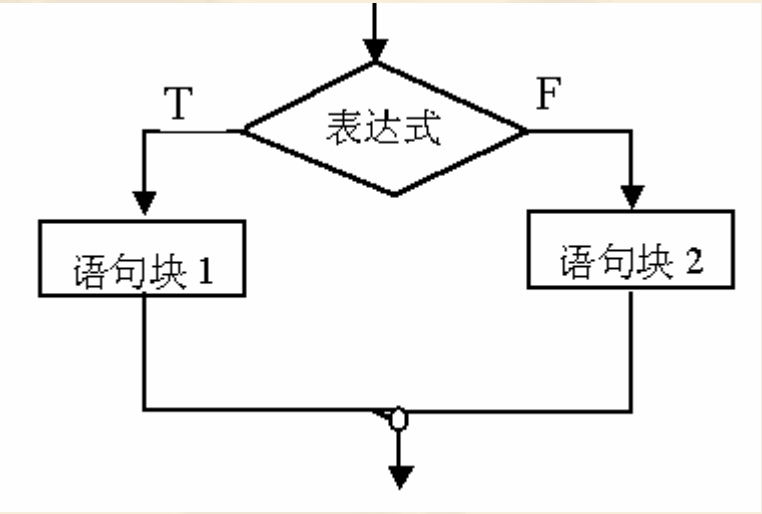


重要的是学会两个数的交换，若上述语句次序变一下，结果如何？

2. If…Then…Else语句(双分支结构)

```
If <表达式> Then
   <语句块1>
Else
   <语句块2>
End If
```

If <表达式> Then <语句1> Else <语句2>



计算分段函数：

$$y=\begin{cases}\sin x+\sqrt{x^2+1} & x\neq 0\\ \cos x-x^3+3x & x=0\end{cases}$$

单分支结构实现：

```
y=cos(x)－x^3+3*x
If x<>0 Then  y=sin(x)+sqr (x*x+1)
```

双分支结构实现：

```
If  x<>0 Then
    y=sin(x)+sqr (x*x+1)
Else
    y=cos(x)－x^3+3*x
End If
```

3. If...Then...ElseIf语句(多分支结构)

形式：

If <表达式1> Then

<语句块1>

ElseIf <表达式2>Then

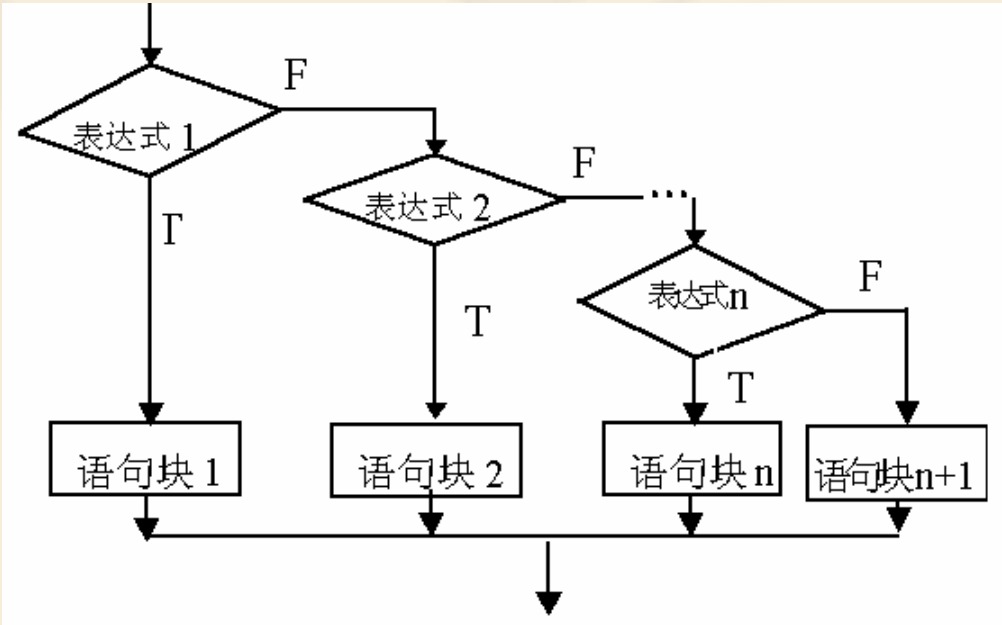
<语句块2>

...

[Else

语句块 n+1]

End If



例4.2 已知变量strC中存放了一个字符，判断该字符是字母字符、数字字符还是其他字符。

用多分支结构实现:

```
If  Ucase(strC) >="A"  And Ucase (strC) <="Z" Then
      Print  strC + "是字母字符"
ElseIf strC >="0"  And strC <="9"  Then
      Print  strC + "是数字字符"
Else
      Print  strC + "其他字符"
End If
```

不管有几个分支，依次判断，当某条件满足，执行相应的语句，其余分支不再执行；若条件都不满足，且有**Else**子句，则执行该语句块，否则什么也不执行。.

ElseIf不能写成 **Else If**。

例4.2a 根据边长判断三角形类型

例4.3 已知百分制成绩mark，显示对应的五级制成绩

哪些正确，哪些错误？

方法一

```
If mark >= 90 Then
        Print "优"
   ElseIf mark >= 80 Then
           Print "良"
   ElseIf mark >= 70 Then
        Print "中"
   ElseIf mark >= 60 Then
        Print "及格"
   Else
     Print "不及格"
 End If
```

方法二

```
If mark < 60 Then
        Print "不及格"
   ElseIf mark < 70 Then
           Print "及格"
   ElseIf mark < 80 Then
        Print "中"
   ElseIf mark < 90 Then
        Print "良"
   Else
     Print "优"
 End If
```

方法三

```
If mark >= 60 Then
    Print "及格"
ElseIf mark >= 70 Then
        Print "中"
ElseIf mark >= 80 Then
    Print "良"
ElseIf mark >= 90 Then
    Print "优"
Else
  Print "不及格"
End If
```

返回72

4. If语句的嵌套

If语句的嵌套是指If或Else后面的语句块中又包含If语句。

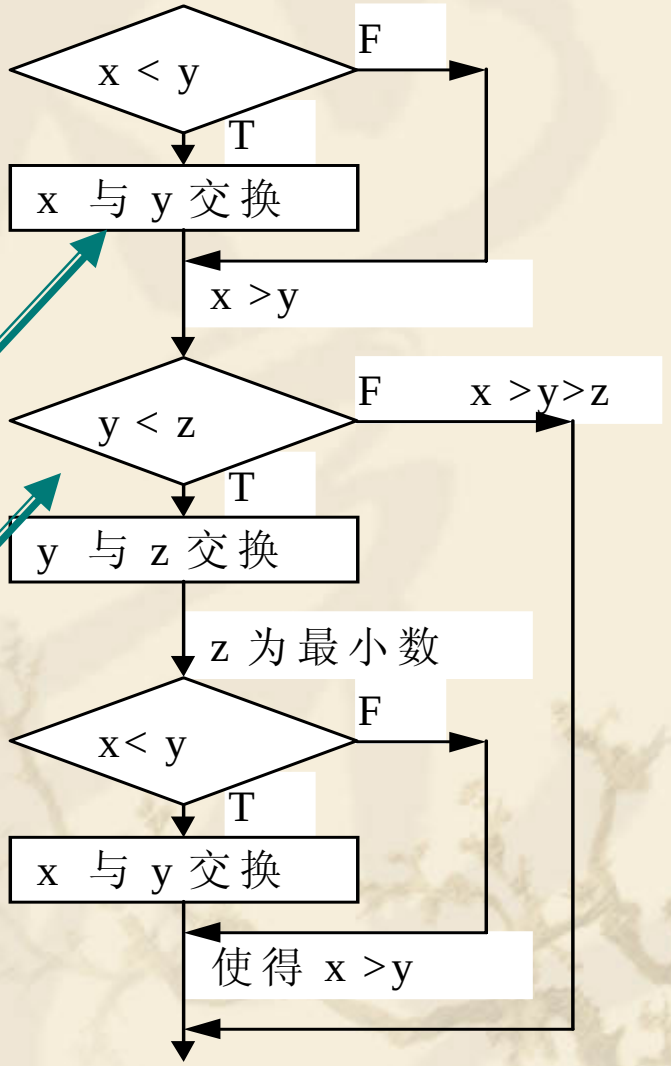
形式如下：

```
If <表达式1> Then
    If  <表达式11> Then
    …
    End If
   …
End If
```

例4.4 已知x，y，z三个数，使得x>y>z

用一个IF语句和一个嵌套的IF语句实现

```
If  x<y Then t=x: x=y: y=t
If  y<z Then
     t=y: y=z: z=t
     If  x<y  Then
       t=x: x=y: y=t
     End If
End If
```



If语句的嵌套注意事项：书写锯齿型； If 与End If配对。

5. Select Case语句(情况语句)

形式：

数值型或字符串表达式

```
Select  Case 变量或表达式
    Case 表达式列表1
        语句块1
    Case 表达式列表2
        语句块2
        …
    [Case Else
        语句块n+1]
End Select
```

<表达式列表>：与<变量或表达式>同类型的下面四种形式之一：

表达式	例："A"
一组枚举表达式(用逗号分隔)	2,4,6,8
表达式1 To 表达式2	60 To 100
Is 关系运算符表达式	Is < 60

例4.5 变量strC中存放了一个字符，判断该字符类型。

在例4.2中用多分支结构实现：

```
If  Ucase(strC) >="A"  And Ucase (strC) <="Z" Then
        Print  strC + "是字母字符"
ElseIf strC >="0"  And strC <="9"  Then
        Print  strC + "是数字字符"
Else
        Print  strC + "其他字符"
End If
```

用**Select Case**语句实现：

```
Select Case  strC
    Case "a" To "z"，  "A" To "Z"
            Print  strC + "是字母字符"
    Case "0" To "9"
            Print  strC + "是数字字符"
    Case Else
            Print  strC + "其他字符"
End Select
```

例4.6 已知坐标点(x，y)，判断其落在哪个象限。

方法一

```
If x > 0 And y >0  Then
   Print "在第一象限"
ElseIf x < 0 And y > 0 Then
   Print "在第二象限"
ElseIf x < 0 And y < 0 Then
   Print "在第三象限"
ElseIf x > 0 And y <0 Then
   Print "在第四象限"
End If
```

方法二

```
Select Case x,y
  Case x > 0 And y > 0
       Print "在第一象限"
  Case x < 0 And y > 0
       Print "在第二象限"
  Case x < 0 And y < 0
       Print "在第三象限"
  Case x > 0 And y <0
       Print "在第四象限"
End Select
```

哪个能实现，哪个不能实现？

方法二代码错误：

1. Select Case 后不能出现多个变量；

2. Case后不能出现变量及有关运算符。

返回72

例4.7由计算机来当一年级的算术老师，要求给出一系列的1～10的操作数和运算符，学生输入该题的答案，计算机根据学生的答案判断正确与否，当结束时给出成绩。

分析:产生1～10操作数，可通过Int(10 * Rnd + 1) 实现

设置的控件名

Label1

Text1

Picture1

Command1

6. 条件函数

（1）IIf函数形式是：

IIf（表达式，当表达式为True时的值，当表达式为False时的值）

例如，求x,y中大的数，放入Tmax变量中，语句如下：

Tmax=IIf（x > y，x，y）

（2）Choose函数形式是：

Choose（数字类型变量，值为1的返回值，值为2的返回值……）

例如，Nop是1-4的值，转换成＋、－、×、÷运算符的语句如下：

0p= Choose（Nop，"+"，"–"，"×"，"÷"）

当值为1，返回字符串“+”，然后放入0p变量中，值为2，返回字符串“–”，依次类推；当Nop是1-4的非整数，系统自动取Nop的整数办法在判断；若Nop不在1~4之间，函数返回Null值。

（3） Switch函数形式是：

Switch（条件表达式1，条件表达式1为True时的值
[，条件表达式2，条件表达式2为True时的值……]）

常 见 错 误

1. 在选择结构中缺少配对的结束语句

对多行式的If块语句中，应有配对的 End If语句结束。

2. 多边选择ElseIf关键字的书写和条件表达式的表示

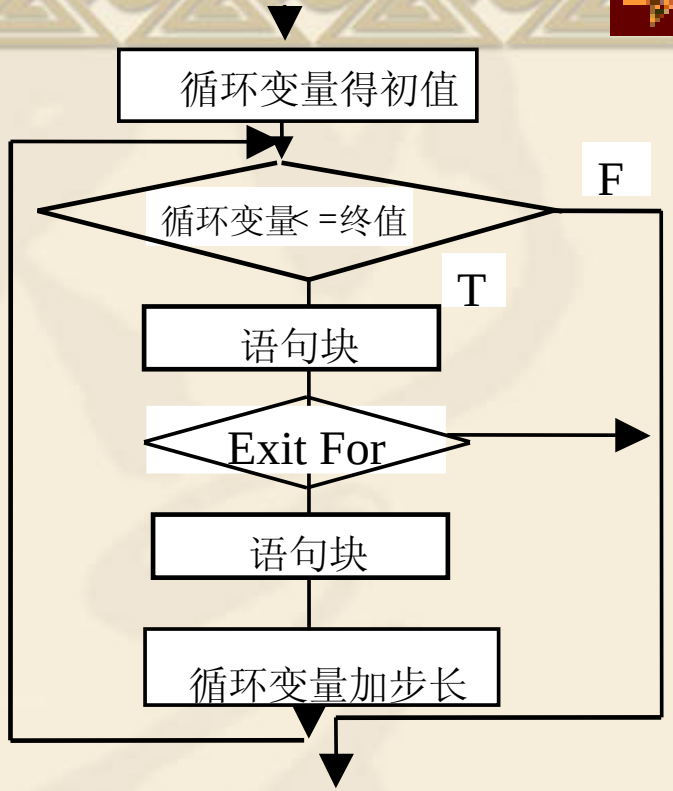
ElseIf 不要写成Else If;

多个条件表达式次序问题,见例4.3。

3. Select Case语句的使用

Select Case 后不能出现多个变量；Case子句后不能出现变量，见例4.6。

4.3 循 环 结 构



1. For循环语句 (一般用于循环次数已知)

形式

For 循环变量＝初值 to 终值 [Step 步长]

语句块

[Exit For] } 循环体

语句块

Next 循环变量

例4.8 计算1~100的奇数和，程序段如下：

```
Dim i %, s%
s=0
For i = 1 To 100 step 2
  s = s + i
Next i
```

等价于：

```
Dim i %, s%
s=0 ： i = 1
re:  If i <= 100 Then
       s = s + i
       i = i +2
       GoTo re
     End If
```

步长 { >0 初值<终值 =1 时,可省略
<0 初值>终值
=0 死循环

循环次数 $= Int(\frac{终值-初值}{步长}+1)$

要注意：

❖ 出了循环，循环控制变量值的问题。

例程序段：

```
For  i=2  To  13  Step   3
    Print  i ,
 Next i
Print     :   Print "I=", i
```

循环执行次数 $= Int(\frac{13-2}{3}+1)=4$

输出 i的值分别为:

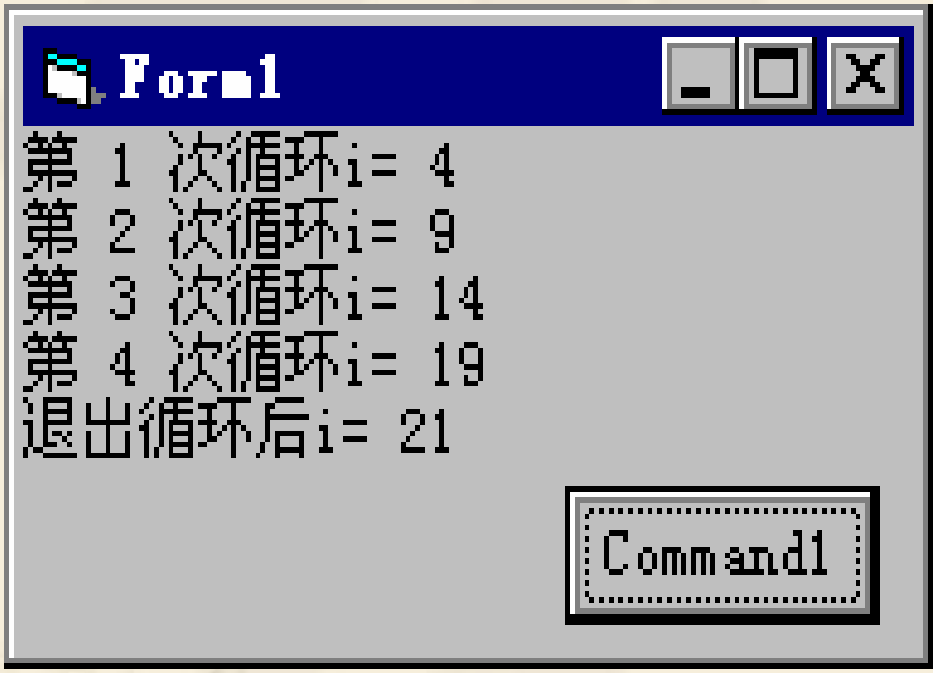
2　5　8　11

出了循环输出为: I=14

❖ 在循环体内对循环控制变量可多次引用；但最好不要对其赋值，否则影响原来的循环控制规律。

例4.9 改变循环控制变量对循环的影响。

```
Private Sub Command1_Click()
   j = 0
   For i = 1 To 20 Step 2
      i = i + 3
      j = j + 1
      Print "第"; j; "次循环i="; i
   Next i
   Print "退出循环后i="; i
End Sub
```



正常情况：i=1,3,5,7,9,11,13,15,17,19

现在：i=4,9,14,19

例4.10 输出可打印的ASCII码字符与它的编码值。

2. Do…Loop循环语句(用于控制循环次数未知)

形式1：

Do { While|Until }<条件>
　　语句块
　　[Exit Do
　　语句块]
Loop

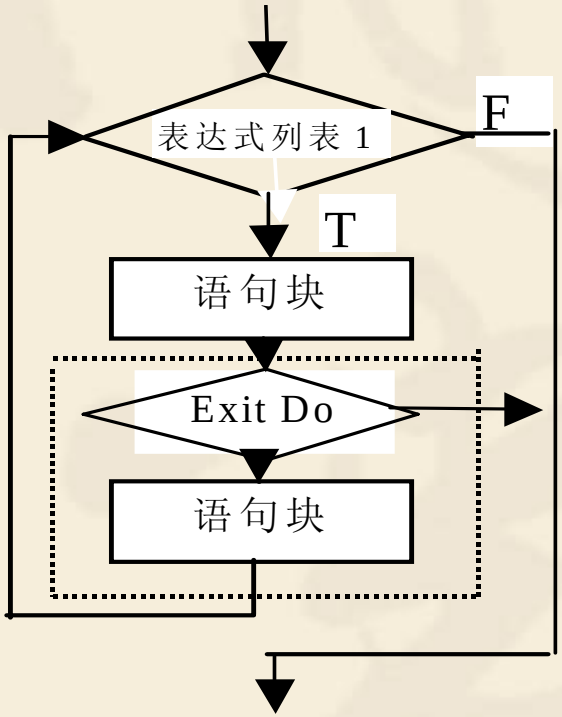


图 1-3-15　Do　While…Loop

形式2：

Do
　　语句块
　　[Exit Do
　　语句块]
Loop { While|Until} <条件>

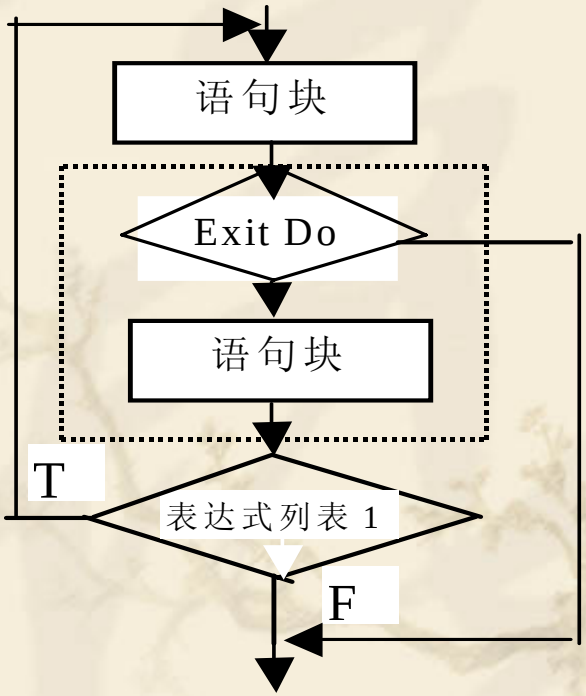


图 1-3-16　Do　…Loop　While

例4.11　我国有13亿人口，按人口年增长0. 8%计算，多少年后我国人口超过26亿。

分析：解此问题两种方法，可根据公式：$26=13*(1+0.008)^n$

直接利用标准对数函数求得；也可利用循环求得，程序如下：

```
Private Sub Command1_Click()
    x = 13
    n = 0
    Do While x < 26
     x = x * 1.008
     n = n + 1
    Loop
    Print n, x
End Sub
```

例4.12用辗转相除法求两自然数*m*，*n*的最大公约数和最小公倍数。

分析：求最大公约数的算法思想：

(1)对于已知两数*m*，*n*，使得*m*>*n*；

(2) *m*除以*n*得余数*r*；

(3)若*r*=0，则*n*为最大公约数结束；否则执行(4)；

(4)*m*←*n*，*n*←*r*，再重复执行(2)。

例 求 *m=14* ,*n=6* 的最大公约数.

```
If m < n Then t = m: m = n: n = t
 r=m mod n
 Do While (r <> 0)
     m=n
     n=r
     r= m mod n
 Loop
 Print "最大公约数=", n
```

m	n	r
14	6	2
6	2	0

辗转相减法

- *m=m-n* *m>n*
- *n=n-m* *n>m*
- *m*、*n*为公约数 *m=n*

```
Do While m <> n
   If m > n Then
       m = m – n
   Else
       n = n – m
   End If
Loop
```

m	n
14	6
8	6
2	6
2	4
2	2

3. 循环的嵌套

一个循环体内又包含了一个完整的循环结构称为循环的嵌套.

例4.13打印九九乘法表.

```
For i = 1 To 9
   For j = 1 To 9
     se = i & "×" & j & "=" & i * j
     Picture1.Print Tab((j - 1) * 9 + 1); se;
   Next j
   Picture1.Print
 Next i
```

For j = 1 To i

```
1×1=1   1×2=2   1×3=3   1×4=4   1×5=5   1×6=6   1×7=7   1×8=8   1×9=9
2×1=2   2×2=4   2×3=6   2×4=8   2×5=10  2×6=12  2×7=14  2×8=16  2×9=18
3×1=3   3×2=6   3×3=9   3×4=12  3×5=15  3×6=18  3×7=21  3×8=24  3×9=27
4×1=4   4×2=8   4×3=12  4×4=16  4×5=20  4×6=24  4×7=28  4×8=32  4×9=36
5×1=5   5×2=10  5×3=15  5×4=20  5×5=25  5×6=30  5×7=35  5×8=40  5×9=45
6×1=6   6×2=12  6×3=18  6×4=24  6×5=30  6×6=36  6×7=42  6×8=48  6×9=54
7×1=7   7×2=14  7×3=21  7×4=28  7×5=35  7×6=42  7×7=49  7×8=56  7×9=63
8×1=8   8×2=16  8×3=24  8×4=32  8×5=40  8×6=48  8×7=56  8×8=64  8×9=72
9×1=9   9×2=18  9×3=27  9×4=36  9×5=45  9×6=54  9×7=63  9×8=72  9×9=81
```

思考：打印上三角或下三角程序如何改动？要打印下三角？

对于循环的嵌套，要注意以下事项：

- 内循环变量与外循环变量不能同名；
- 外循环必须完全包含内循环，不能交叉；
- 不能从循环体外转向循环体内，反之则可以。

正确

```
For  ii =1  To  10
    For jj=1 To  20
      …
    Next  jj
Next  ii
```

错误

```
For  ii =1  To  10
    For  jj=1 To  20
      …
    Next  ii
Next  jj
```

```
For  ii =1  To  10
      …
Next  ii
For  ii =1  To  10
      …
Next  ii
```

```
For  ii =1  To  10
    For  ii=1 To  20
      …
    Next  ii
Next  ii
```

4.4 其他辅助控制语句

1. Go To 语句

形式： **Go To {标号|行号}**

作用：无条件地转移到标号或行号指定的那行语句.

标号是一个字符序列,行号是一个数字序列.

2.Exit语句

多种形式： **Exit For、Exit Do、Exit Sub、Exit Function**等。

作用：退出某种控制结构的执行。

3. End语句

多种形式： End、 End If、 End Select、 End With、 End Type、 End Sub、 End Function、

作用：End结束一个程序的运行；其余表示某个结构的结束，与对应的结构语句配对出现。

4.With 语句

形式如下：

With 对象
语句块
End With

作用：对某个对象执行一系列的操作，而不用重复指出对象的名称。

```
With  Label1
    .Height = 2000
    .Width = 2000
    .FontSize=22
   .Caption = "MyLabel"
End With
```

等价

```
Label1.Height = 2000
Label1.Width = 2000
Label1.FontSize=22
Label1.Caption = "MyLabel"
```

4.5 常用算法（一）

算法是对某个问题求解过程的描述

1. 累加、连乘

1～100的5或7的倍数的和

```
Sum = 0
For i = 1 To 100
  If i Mod 5 = 0 Or i Mod 7 = 0 Then
     Sum = Sum + i
   End If
Next  i
Print Sum
```

3～10的乘积

```
t = 1
For i =3 To 10
  t = t * i
Next i
Print t
```

思考：若把循环体前面置各变量初值的语句放在循环体内，程序运行时会产生什么情况？

例4.14 求自然对数e的近似值,要求其误差小于0.00001，近似公式为：

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{i!} + \ldots = \sum_{i=0}^{\infty} \frac{1}{i!} \approx 1 + \sum_{i=1}^{m} \frac{1}{i!}$$

该例题涉及两个问题：

（1）用循环结构求级数和的问题。本例根据某项值的精度来控制循环的结束与否。

（2）累加：*e=e+t*　　循环体外对累加和的变量清零　*e=0*

连乘：*n=n*i*　　循环体外对连乘积变量置1　　*n=1*

```
Private Sub Form_Click()
   Dim i%,n&, t!, e!
    e = 0   :  n = 1                ‘ e存放累加和、n存放阶乘
    i = 0   : t = 1                 ‘ i计数器、t第i项的值
    Do While t > 0.00001
      e = e + t :  i = i + 1  ‘ 累加、连乘
      n = n * i :  t = 1 / n
    Loop
    Print ″计算了 ″; i; ″ 项的和是 ″; e
End Sub
```

2. 求素数

素数是一个大于2，且不能被1和本身以外的整数整除的整数。
判别某数*m*是否为素数最简单的方法是:
对于m 从 i=2，3，…，m-1判别m能否被i整除，只要有一个能整除，m不是素数，否则m是素数。

例4.15 求100以内素数的代码:

求100以内的素数

```
For m = 2 To 100
    For i = 2 To m - 1
        If (m Mod i) = 0  Then   GoTo  NotNextM
    Next i
    Print m
NotNextM:
Next m
```

m是否为素数

思考： 此例用Go To语句对非素数不作判断，若不用GoTo语句，如何修改程序？

3. 穷举法

“穷举法”也称为“枚举法”或“试凑法”，即将可能出现的各种情况一一测试，判断是否满足条件，一般采用循环来实现。

例4.16 百元买百鸡问题。假定小鸡每只5角，公鸡每只2元，母鸡每只3元。现在有100元钱要求买100只鸡，编程列出所有可能的购鸡方案。

设母鸡、公鸡、小鸡各为x、y、z只，根据题目要求，列出方程为：

x+y+y=100

3x+2y+0.5z=100

三个未知数，两个方程，此题有若干个解。

解决此类问题采用“试凑法”，把每一种情况都考虑到。

方法一：最简单三个未知数利用三重循环来实现。

方法二：从三个未知数的关系，利用两重循环来实现。

4. 递推法

“递推法”又称为“迭代法”，其基本思想是把一个复杂的计算过程转化为简单过程的多次重复。每次重复都从旧值的基础上递推出新值，并由新值代替旧值。

例4.17 猴子吃桃子。小猴在某天摘桃若干个，当天吃掉一半多一个；第二天吃了剩下的桃子的一半多一个；以后每天都吃尚存桃子的一半多一个，到第7天要吃时只剩下一个，问小猴共摘下了多少个桃子？

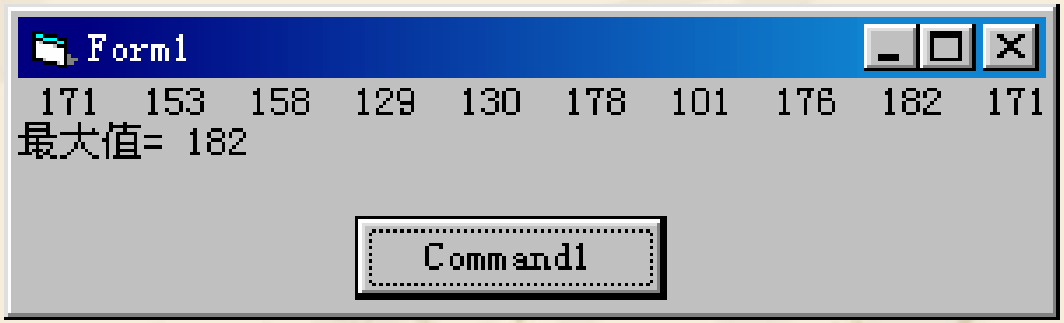
分析：这是一个“递推”问题，先从最后一天推出倒数第二天的桃子，再从倒数第二天的桃子推出倒数第三天的桃子……。设第n天的桃子为x_n，那么它是前一天的桃子数的x_{n-1}的一半减1，即$x_n=(x_{n-1}+1)\times 2$

5. 最小、最大值

在若干个数中求最大值，一般先假设一个较小的数为最大值的初值，若无法估计较小的值，则取第一个数为最大值的初值；然后将每一个数与最大值比较，若该数大于最大值，将该数替换为最大值；依次逐一比较。

例 随机产生10个100～200之间的数，求最大值。

```
Private Sub Command1_Click()
    Max = 100
    For i = 1 To 10
      x = Int(Rnd * 101 + 100)
      Print x;
      If x > Max Then Max = x
    Next i
    Print
    Print "最大值="; Max
End Sub
```



例4.18 实际应用，求最短残料。

有一根长度为321米的钢材料，要将它截取成两种规格a、b的长度分别为17米和27米的短料，每种至少1段，问分隔成a，b各多少段后，剩余的残料r最少？

分析，该题利用“试凑法”通过二重循环求残料r的最小值正数，残料不可能是负数。程序如下：

```
Private Sub Command1_Click()
    Dim a%, b%, r!, ia%, ib%
    r = 321                                  ‘ 最小值初值取钢材料的长度
    For b = 1 To 321 \ 27                    ‘ b最多的段数
       For a = 1 To 321 \ 17 - b             ‘ a最多的段数
         t = 321 - b * 27 - a * 17           ‘ 当前的残料
         If t > 0 And t < r Then
            r = t                            ‘ 求最短的残料
            ia = a                           ‘ 最短残料时a的段数
            ib = b                           ‘ 最短残料时b的段数
         End If
       Next a
    Next b
    Print ia, ib, r
End Sub
```

4.6 常 见 错 误

1. 不循环或死循环的问题

主要是循环条件、循环初值、循环终值、循环步长的设置有问题。

2. 循环结构中缺少配对的结束语句

For 少 配对的Next

3. 循环嵌套时,内外循环交叉

4. 累加、连乘时，存放累加、连乘结果的变量赋初值问题

(1)一重循环

在一重循环中, 存放累加、连乘结果的变量初值设置应在循环语句前。

（2）多重循环

这要视具体问题分别对待。

第五章　数　组

（4学时）

5.1 数组的概念

1. 引例

例5.1若我们要求一个班100个学生的平均成绩，然后统计高于平均分的人数。

按以前简单变量的使用和循环结构相结合，求平均成绩程序段如下：

```
aver = 0
For i = 1 To 100
  mark = InputBox("输入" + i + "位学生的成绩")
  aver = aver + mark
Next i
aver = aver / 100
```

但若要统计高于平均分的人数，则无法实现。mark是一个简单变量，存放的是最后一个学生的成绩。

已有知识解决方法：再重复输入成绩，带来两个问题：

（1）输入数据的工作量成倍增加；

（2）若本次输入的成绩与上次不同，则统计的结果不正确。

解决此问题的根本方法，引入数组，始终保持输入的数据，一次输入，多次使用。

5.2 静态数组及声明

数组不是一种数据类型，而是一组相同类型的变量的集合，数组必须先声明后使用。

两类数组： 静态(定长)数组、动态(可变长)数组

1. 静态数组及声明

形式： **Dim 数组名(下标1[,下标2...]) [As 类型]**

声明了数组的名、维数、大小、类型

维数：几个下标为几维数组，最多60维。

下标：[下界 To] 上界 省略下界为0 ,必须为常数。

每一维大小：上界－下界+1

数组大小： 每一维大小的乘积

例：**Dim mark(1 to 100) As Integer**

mark(1)	mark(2)	mark(3)	……	mark(99)	mark(100)

Dim lArray(0 To 3， 0 To 4) As Long 共有 4×5个元素

等价于: **Dim lArray(3， 4) As Long**

lArray(0,0)	lArray(0,1)	lArray(0,2)	lArray(0,3)	lArray(0,4)
lArray(1,0)	lArray(1,1)	lArray(1,2)	lArray(1,3)	lArray(1,4)
lArray(2,0)	lArray(2,1)	lArray(2,2)	lArray(2,3)	lArray(2,4)
lArray(3,0)	lArray(3,1)	lArray(3,2)	lArray(3,3)	lArray(3,4)

注意：(1)下界缺省为0，也可在重新定义数组的下界。例如：

Option Base 1

(2)错误的声明，下标是变量

n =Inputbox("输入n "）： Dim x(n) As Single

(3)在数组声明中的下标说明了数组的整体，即每维的大小；而在程序其他地方出现的下标表示数组中的一个元素。两者写法形式相同，但意义不同。

例如： **Dim x(10) As Integer** ' 声明了x数组有11个元素

x(10)=100 ' 对x(10)这个数组元素赋值

5.3 动态数组及声明

动态数组指在声明数组时未给出数组的大小(省略括号中的下标)，当要使用它时，随时用ReDim语句重新指出数组大小。形式如下：

ReDim 数组名(下标[，下标2…]) [As 类型]

例 Sub Form_Load( )

Dim x( ) As Single

…

n =Inputbox(“输入n”）

ReDim x(n)

…

End Sub

例5.2 求若干个学生的平均分。

说明：

- Dim、Private、Public变量声明语句是说明性语句，可出现在过程内或通用声明段；ReDim语句是执行语句，只能出现在过程内。
- 在过程中可多次使用ReDim来改变数组的大小和维数。
- 使用ReDim语句会使原来数组中的值丢失，可以在ReDim语句后加Preserve参数来保留数组中的数据。使用Preserve只能改变最后一维的大小，前面几维大小不能改变。 ReDim中的下标可以是常量，也可以是有了确定值的变量。
- 静态数组在程序编译时分配存储单元，动态数组在运行时分配存储单元。

5.4 数组的基本操作

1. 数组元素的赋初值

(1)用循环

```
For  i  = 1 To 10
 iA(i)=0
Next i
```

(2) Array函数

```
Dim ib As Variant
ib = Array("abc", "def", "67")
For i = 0 To UBound(ib)
    Picture1.Print ib(i); " ";
Next i
```

注意：

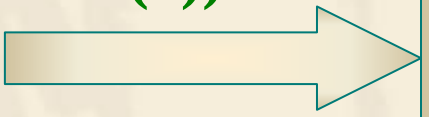
• 利用Array对数组各元素赋值，声明的数组是可调数组或连圆括号都可省，并且其类型只能是Variant。

• 数组的下阶为零，上界由Array函数括号内的参数个数可决定，也可通过函数Ubound获得。

2. 数组的赋值

在VB6.0中，提供了数组直接对数组的赋值。例如：

```
Dim a() As Variant, b()  As Variant, i%
a = Array(1, 2, 3, 4, 5)
ReDim b(UBound(a))
b = a
```



```
For i = 0 To UBound(a)
    b(i) = a(i)
Next i
```

注意：赋值号左边的数组只能声明为Variant的可调数组或简单变量。

3. 数组的输出

输出方阵sC中的下三角元素

```
For i = 0 To 4
    For j = 0 To i
      sc(i, j) = i * 5 + j
      Print sc(i, j); " ";
    Next j
    Print     ' 换行
 Next I
```

4. 求数组中最大元素及所在下标

```
Dim Max As Integer,iMax As Integer
Max=iA(1)：iMax=1
For  i = 2  To  10
    If   iA(i)>Max  Then
       Max=iA(i)
       iMax=i
    End If
Next I
```

5. 将数组中各元素交换

```
For  i =1 To  10\2
  t=iA(i)
  iA(i)=iA(10-i+1)
  iA(10-i+1)=t
Next I
```

交换前：

2	4	6	8	10	1	3	5	7	9

交换后：

9	7	5	3	1	10	8	6	4	2

图 1-3-22　数组中个元素交换

5.5 控 件 数 组

一组相同类型的控件组成。 它们共用一个控件名，具有相同的属性，建立时系统给每个元素赋一个唯一的索引号(Index)。

控件数组共享同样的事件过程，通过返回的下标值区分控件数组中的各个元素。

例： **Private Sub cmdName _Click(Index As Integer)**

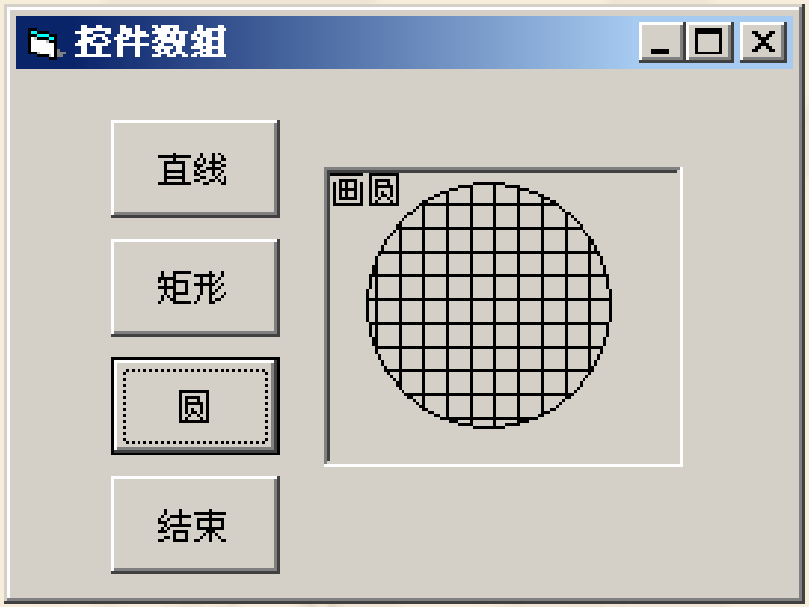
```
    …
    If Index = 3 then
    ‘ 处理第四个命令按钮的操作
    End If
    …
End Sub
```

1. 在设计时建立控件数组

- 在窗体上画出控件，进行属性设置，这是建立的第一个元素
- 选中该控件，进行“Copy”进行若干次和“Paste”操作建立了所需 个数的控件数组元素。
- 进行事件过程的编程。

例5.3 建立含有四个命令按钮的控件数组，当单击某个命令按钮，分别显示不同的图形或结束操作。

```
Private Sub Command1_Click(Index As Integer)
   Select Case Index
        Case 0
          …… "画直线"
        Case 1
          …… "画矩形"
       Case 2
          …… "画圆"
       Case Else
         End
        End Select
End Sub
```



控件名	Index	Caption
Command1	0	直线
Command1	1	矩形
Command1	2	圆
Command1	3	结束
Picture1	空白	——

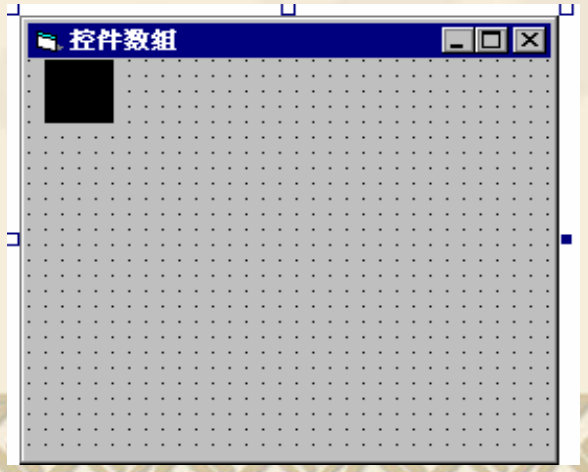
2运行时添加控件数组

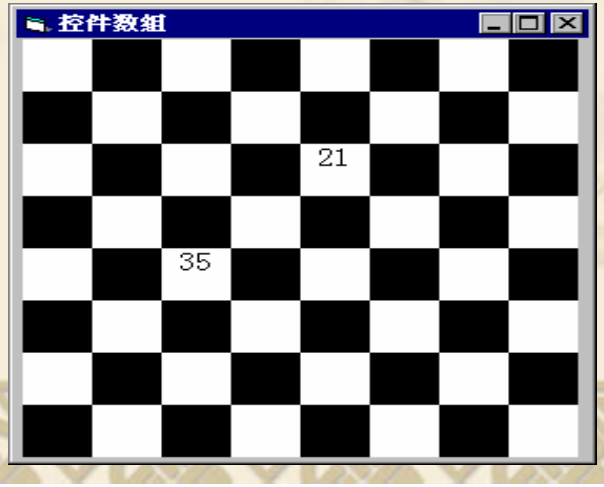
建立的步骤如下：

- 在窗体上画出某控件，设置该控件的Index值为0，表示该控件为数组，这是建立的第一个元素。
- 在编程时通过Load方法添加其余的若干个元素，也可以通过Unload方法删除某个添加的元素
- 每个新添加的控件数组通过Left和Top属性确定其在窗体的位置，并将Visible 属性设置为True。

例5.4 利用在运行时产生控件数组，构成一个国际象棋棋盘。

当单击棋格，显示对应的序号,并且将所有棋格颜色变反。





5.6 自定义数据类型

一组不同类型变量的集合。相当于C语言中的结构类型；Pascal中的记录类型。

1. 自定义类型的定义

形式如下：

Type 自定义类型名
　　元素名[(下标)] As 类型名
　　…
　　[元素名[(下标)] As 类型名]
End Type

元素名：表示自定义类型中的一个成员
下标：表示是数组
类型名：为标准类型

例如，以下定义了一个有关学生信息的自定义类型：

```
Type StudType
　　No As Integer                  ' 学号
　　Name As String * 20            ' 姓名
　　Sex As String * 1              ' 性别
　　Mark(1 To 4) As Single         ' 4门课程成绩
　　Total  As Single               ' 总分
End Type
```

注意：

(1)自定义类型一般在标准模块(.BAS)中定义，默认是Public；在窗体必须是Private。

(2)自定义类型中的元素类型可以是字符串，但应是定长字符串。

(3)不要将自定义类型名和该类型的变量名混淆，前者表示了如同Integer、Single等的类型名，后者VB根据变量的类型分配所需的内存空间，存储数据。

(4)自定义类型一般和数组结合使用，简化程序的编写。

2. 自定义类型变量的声明和使用

(1) 声明形式：

Dim 变量名 As 自定义类型名

例 Dim Student As StudType

(2) 引用

形式：变量名.元素名

例 表示Student变量中的姓名，第4门课程的成绩，则表示如下：

Student.Name，Student.Mark(4)

3. 自定义类型数组的应用

例5.5 利用自定义类型数组，编写一个输入、显示、查询程序。

5.7 常用算法（二）

1. 统计

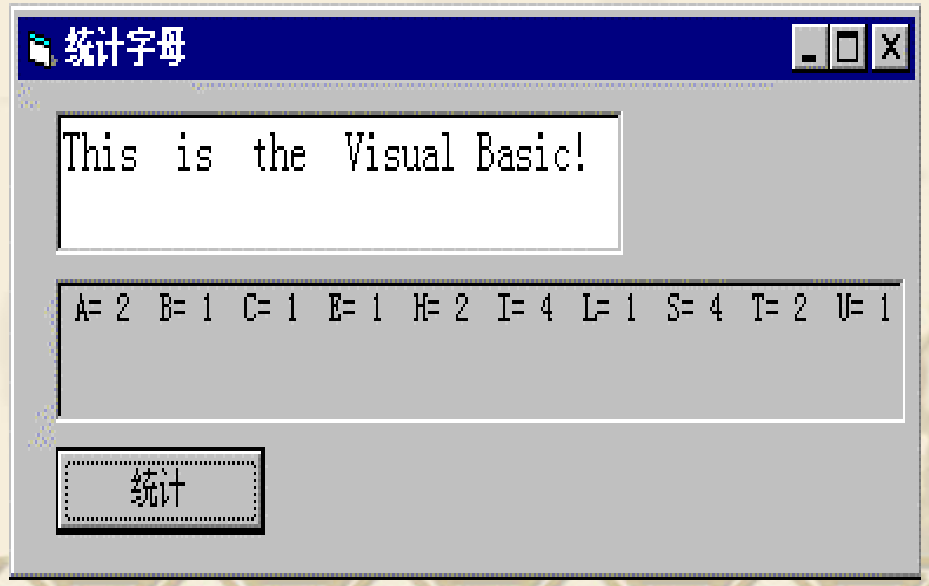
例5.6 输入一串字符，统计各字母出现的次数，不区分字母大小写。

分析：

- 统计26个字母出现的个数，先声明一个具有26个元素的数组，每个元素的下标表示对应的字母，元素的值表示对应字母出现的次数。
- 从输入的字符串中逐一取出字符，转换成大写字符（不区分大小写），进行判断。

运行界面：

```
For I = 1 To le
  c = UCase(Mid(Text1, I, 1))
  If c >= "A" And c <= "Z" Then
     j = Asc(c) - 65 + 1
     a(j) = a(j) + 1
  End If
Next I
```



2. 大量数据的输入

例5.7 输入一系列的数据，并将它们分离后存放在数组中。对输入的数据允许修改和自动识别非数字数据。

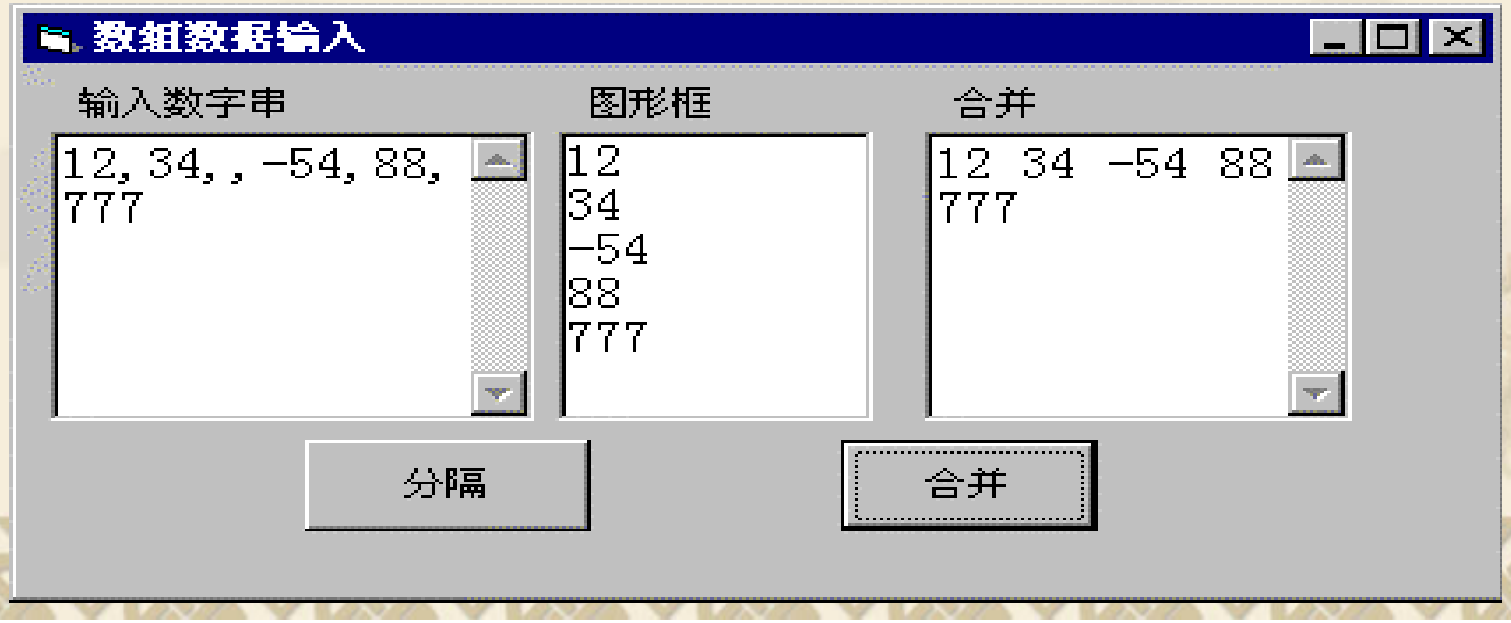
分析：

（1）利用文本框输入和编辑数据，输入时去除非法数字。

（2）输入结束利用Rplace函数去除重复输入的分隔符；

（3）对利用Split函数按分隔符分离，放到数组中；

（4）还可利用Join函数将数组中各元素合并成一个字符串。



3. 数组排序（选择法）

例5.8 对已知存放在数组中的*n*个数，用选择法按递增顺序排序。

(1) 从*n*个数的序列中选出最小的数(递增)，与第1个数交换位置；

(2) 除第1个数外，其余*n*−1个数再按(1)的方法选出次小的数，与第2个数交换位置；

(3) 重复(1)*n*−1遍，最后构成递增序列。

```
For i = 1 To n - 1
   iMin= i
    For j = i+1 To n
       If iA(j) < iA(iMin) Then  iMin = j
    Next j
    t= iA(i):   iA(i) = iA(iMin) :  iA(iMin) = t
Next I
```

						原始数据	8	6	9	3	2	7
a(1)	a(2)	a(3)	a(4)	a(5)	a(6)	第 1 趟交换后	2	6	9	3	8	7
	a(2)	a(3)	a(4)	a(5)	a(6)	第 2 趟交换后	2	3	9	6	8	7
		a(3)	a(4)	a(5)	a(6)	第 3 趟交换后	2	3	6	9	8	7
			a(4)	a(5)	a(6)	第 4 趟交换后	2	3	6	7	8	9
				a(5)	a(6)	第 5 趟无交换	2	3	6	7	8	9

图 1.3.25　排序过程示意图

排序（冒泡法）

例5.9 选择法排序在每一轮排序时找最小(递增次序)数的下标，出了内循环(一轮排序结束)，再交换最小数的位置；而冒泡法排序在每一轮排序时只要将第一个与其他几个比较，只要次序不对，就交换，出了内循环，最小数已冒出。排序进行的过程见下表。

```
For i = 1 To n-1                    ' 进行n-1轮比较
    For j = i+1  To n               ' 从n~i个元素进行两两比较
       If iA(j) < iA(i) Then        ' 若次序不对，则马上进行交换位置
         t = iA(j)  :    iA(j) = iA(i)  :      iA(i) = t
       End If
    Next j          ' 出了内循环，一轮排序结束，最小数已冒到最上面
  Next i
```

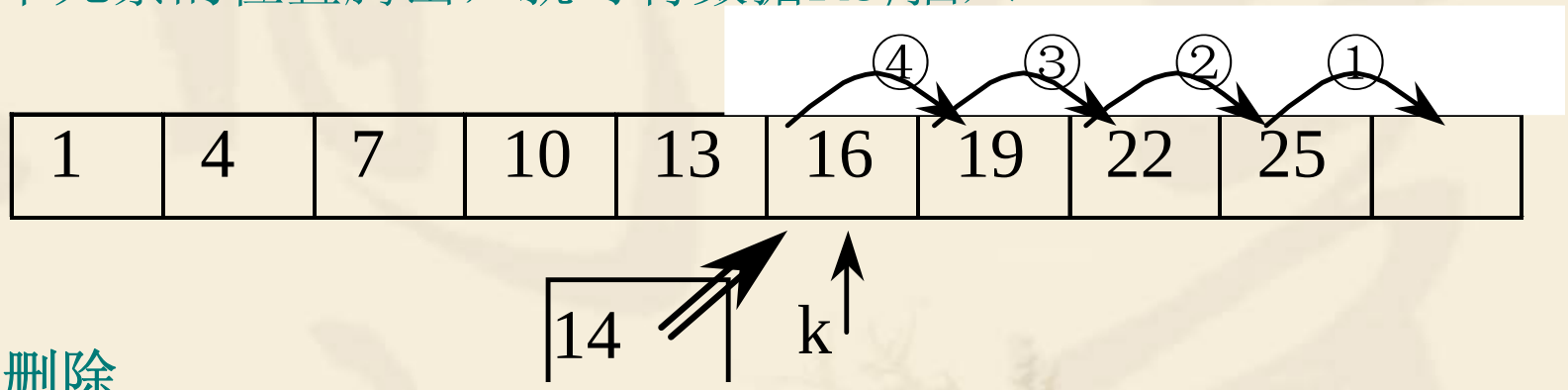
						原始数据	8	6	9	3	2	7
a(1)	a(2)	a(3)	a(4)	a(5)	a(6)	第 1 趟排序	2	8	9	6	3	7
	a(2)	a(3)	a(4)	a(5)	a(6)	第 2 趟排序	2	3	9	8	6	7
		a(3)	a(4)	a(5)	a(6)	第 3 趟排序	2	3	6	9	8	7
			a(4)	a(5)	a(6)	第 4 趟排序	2	3	6	7	9	8
				a(5)	a(6)	第 5 趟排序	2	3	6	7	8	9

4. 数组元素的插入与删除

(1) 插入 例5.10

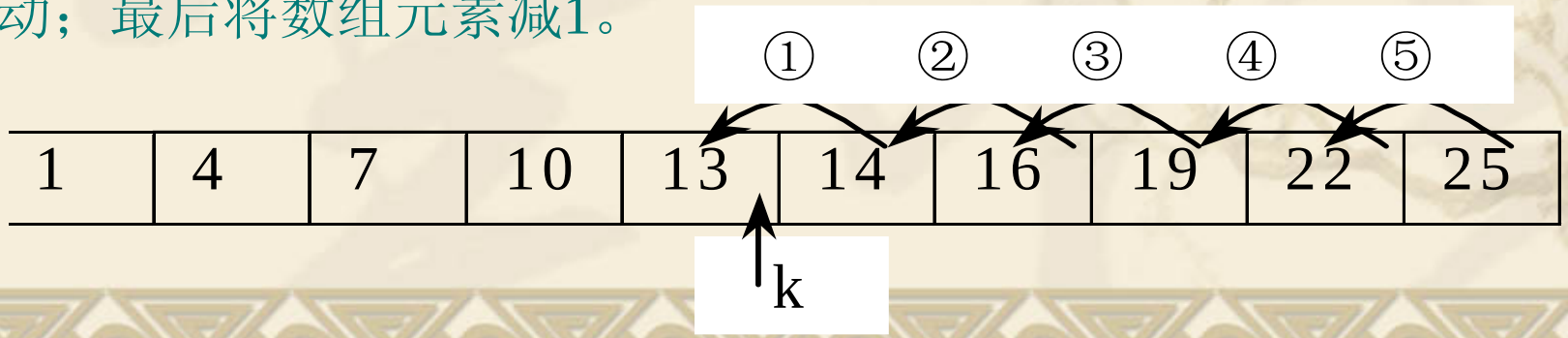
在有序数组a(1 to n)(原有$n-1$个元素)插入一个值Key元素，算法：

- 查找要插入的位置k（$1<=k<=n-1$）
- 腾出位置，把最后一个元素开始到第k个元素往后移动一个位置
- 第k个元素的位置腾出，就可将数据Key插入



(2) 删除

要找到欲删除的元素的位置k；然后从$k+1$到n个位置开始向前移动；最后将数组元素减1。



5.8 数组中常见错误和注意事项

1. 静态数组声明下标出现变量

```
n = InputBox("输入数组的上界")
Dim a(1 To n) As Integer
```

2. 数组下标越界

引用的下标比数组声明时的下标范围大或小。

```
Dim a(1 To 30) As Long, i%
a(1) = 1: a(2) = 1
For i = 2 To 30
    a(i) = a(i - 2) + a(i - 1)
Next i
```

a(0)不存在

3. 数组维数错

数组声明时的维数与引用数组元素时的维数不一致。

```
Dim a(3, 5) As Long
    a(i)=10
```

4. Aarry函数与Split函数使用问题

只能对Variant 的变量或动态数组赋值。

5. 获得数组的上界、下界

Ubound 、Lbound函数

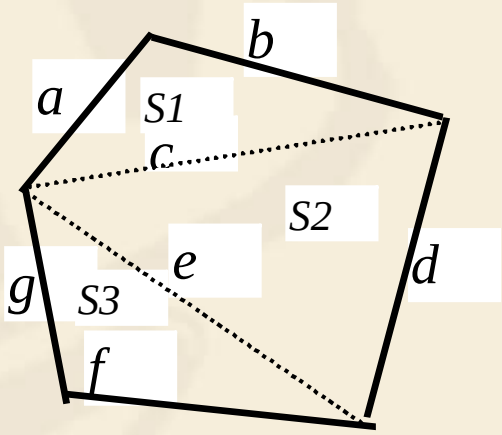
第六章　过　程

（5学时）

6.1 函数过程的定义

除了系统提供的内部函数过程和事件过程外，用户可自定义过程：

- 以Sub保留字开始的为子过程；
- 以Function保留字开始的为函数过程。



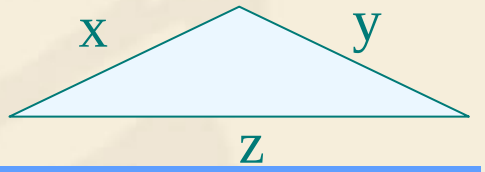
1. 引例6.1

已知多边形的各条边的长度，要计算多边形的面积。

计算多边形面积，可将多边形分解成若干个三角形。

计算三角形面积的公式如下：

$$area = \sqrt{c(c-x)(c-y)(c-z)} \qquad c = \frac{1}{2}(x+y+z)$$



定义函数过程**area**：

```
Public Function area(x!, y!, z!) As Single
  Dim c!
  c = 1 / 2 * (x + y + z)
  area = Sqr(c * (c - x) * (c - y) * (c - z))
End Function
```

调用函数过程：

```
Sub  command1_click()
  …… 输入若干个三角形边长
  S=area(a,b,c)+area(c,d,e)
  S=S+area(e,f,g)
  Print S
End Sub
```

2. 函数过程的定义

自定义函数过程有两种方法：

(1)利用“工具”菜单下的“添加过程”命令定义，生成一个函数的框架。

(2)利用代码窗口直接定义。

函数过程形式：

Function 函数过程名([参数列表]**)** [As 类型]

局部变量或常数定义
语句块
函数名 = 返回值
[Exit Function]
语句块
函数名 = 返回值

（函数过程体）

End Function

函数过程名：命名规则同变量名

参数列表形式：**[ByVal]变量名[()][As 类型]**

称为形参或哑元，仅表示参数的个数、类型，无值。

函数名 = 返回值　　在函数体内至少对函数名赋值一次。

[Exit Function] ：　表示退出函数过程。

例6.2 定义MyReplace（S,OldS,NewS）函数过程，具有Replace函数功能。

```
Function MyReplace(s$, OldS$, NewS$) As String
    Dim i%, lenOldS%
    lenOldS = Len(OldS)              ' 取OldS字符子串长度
    i = InStr(s, OldS)               ' 在字符串中找有否OldS字符子串
    Do While i > 0                   ' 找到用NewS 字符子串替换OldS字符子串
      s = Left(s, i - 1) + NewS + Mid(s, i + lenOldS)
      i = InStr(s, OldS)             ' 找下一个OldS字符子串
    Loop
    MyReplace = s                    ' 替换后的字符串赋值给函数过程名
End Function
```

假定S为“abcdefgabcd”，Oolds为“cd”，News为“3”，调用MyReplace函数的返回值为“ab3efgab3”。上述程序循环执行2次：

(“abcdefgabcd”,“cd”,“3”) 第1 次 i=3 结果 “ab3efgabcd”

(“ab3efgabcd”,“cd”,“3”) 第2 次 i=9 结果 “ab3efgab3”

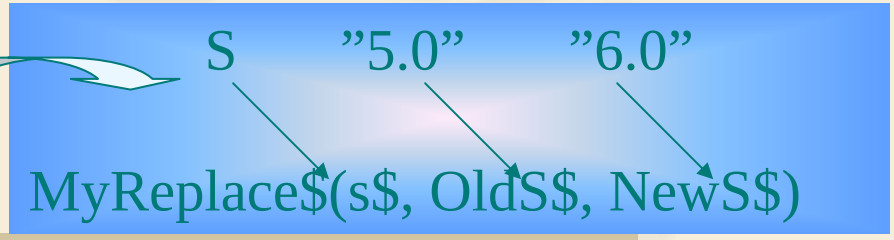
3. 函数过程的调用

函数过程调用同标准函数调用，形式：**函数过程名([参数列表])**

参数列表：称为实参或实元，它必须与形参个数相同，位置与类型一一对应。可以是同类型的常量、变量、表达式。

程序运行的流程：

```
Private Sub Command1_Click()
   s= "VB程序设计教程5.0版"
   Print MyReplace(s, "5.0", "6.0")
End Sub
```




```
Function MyReplace$(s$, OldS$, NewS$)
    Dim i%, lenOldS%
    lenOldS = Len(OldS)
    i = InStr(s, OldS)
    Do While i > 0
      s= Left(s, i - 1) + NewS + Mid(s, i + lenOldS)
      i = InStr(s, OldS)
    Loop
    MyReplace = s
End Function
```

6.2 子 过 程

函数过程的不足：

（1）不是为了获得某个函数值，而是为了某种功能的处理，如例1.1。

（2） 要获得多个结果。

1. 引例6.3

编写一个两个数交换的过程供多次调用。

Swap (x,y)子过程的定义

```
Public Sub Swap(x, y)
   Dim t
   t = x
   x = y
   y = t
End Sub
```

主调程序调用Swap子过程

```
Private Sub Form_Click()
    Dim a, b
    a = 10
    b = 20
    Call Swap (a, b)
    Print "a=";a,"，  b="; b
End Sub
```

2. 子过程定义

Sub 子过程名[(参数列表)]
局部变量或常数定义
语句
[Exit Sub]
语句
End Sub

3. 子过程的调用

子过程名 [参数列表]
或 **Call** 子过程名(参数列表)

4. 子过程与函数过程区别:

(1)函数过程名有值，有类型，在函数体内至少赋值一次；
子过程名无值，无类型，在子过程体内不能对子过程名赋值；
(2)调用时，子过程调用是一句独立的语句。
函数过程不能作为单独的语句加以调用，必须参与表达式运算。
(3)一般当过程有一个函数值，使用函数过程较直观；
反之若过程无返回值，或有多个返回值，使用子过程较直观。

例6.4分别编一计算某级数部分和的子过程和函数过程，并调用。

级数为：$1+x+\frac{x^2}{2!}+\ldots+\frac{x^n}{n!}+\ldots$　　精度为：$\left|\frac{x^n}{n!}\right|<\text{eps}$

函数过程实现求部分级数和

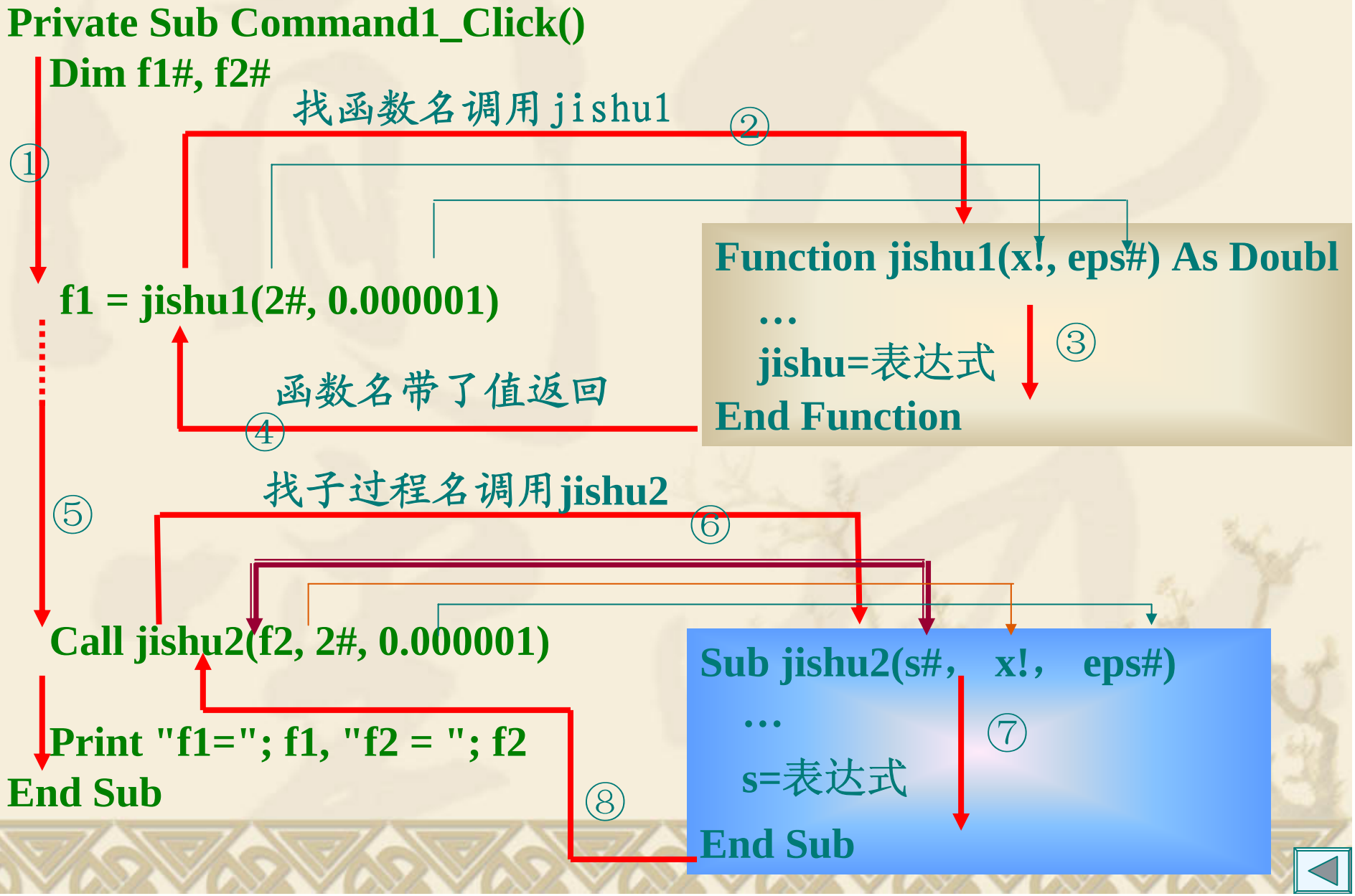
```
Function jishu1(x!, eps#) As Double
  Dim n%, s#, t#
  n = 1: s = 0: t = 1
  Do While (Abs(t) >= eps)
     s = s + t
     t = t * x / n
     n = n + 1
  Loop
  jishu1 = s
End Function
```

子过程实现求部分级数和

```
Sub jishu2(s#, x!, eps#)
   Dim n%, t#
n = 1: s = 0: t = 1
Do While (Abs(t) >= eps)
    s = s + t
    t = t * x / n
    n = n + 1
Loop
End Sub
```

```
f1 = jishu1(2#, 0.000001)     ’ 调用函数过程
Call jishu2(f2, 2#, 0.000001)‘ 调用子过程
或 jishu2 f2,2#,0.00001
```

程序运行流程：


Private Sub Command1_Click()
Dim f1#, f2#
找函数名调用jishu1
①
②
f1 = jishu1(2#, 0.000001)
Function jishu1(x!, eps#) As Doubl
…
jishu=表达式
③
End Function
函数名带了值返回
④
找子过程名调用jishu2
⑤
⑥
Call jishu2(f2, 2#, 0.000001)
Sub jishu2(s#, x!, eps#)
…
⑦
s=表达式
End Sub
Print "f1="; f1, "f2 = "; f2
End Sub
⑧

6.3 参数传递

指主调过程的实参传递给被调过程的形参。

1. 传址与传值

传址：

形参得到的是实参的**地址**，当形参值的改变同时**也改变实参的值**。

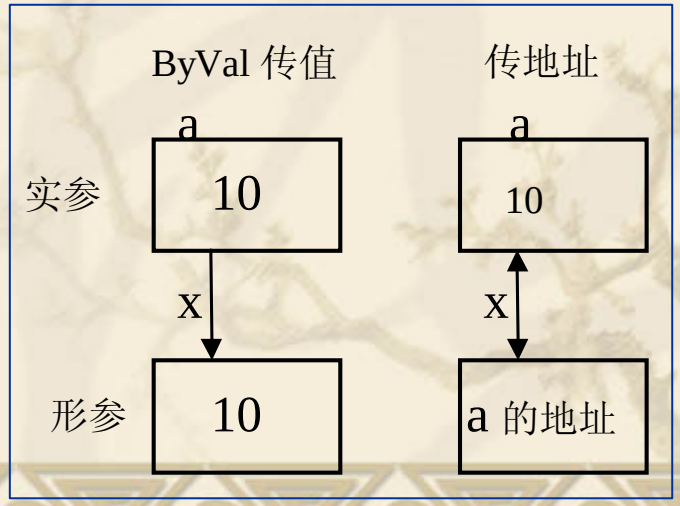
传值：

形参得到的是实参的**值**，形参值的改变**不会影响实参的值**。

例6.5 两个变量的交换。

```
Sub Swap1(ByVal  x%, ByVal  y%)
      t% = x: x = y: y = t
End Sub
Private Sub Command1_Click()
      a% = 10: b% = 20: Swap1 a, b  '传值
      Print "A1="; a, "B1="; b
      a = 10: b = 20:  Swap2 a, b       '传址
      Print "A2="; a, "B2="; b
End Sub
```

```
Sub Swap2(x%, y%)
      t% = x: x = y: y = t
End Sub
```



例6.6 求若干个数的最大公约数。

2. 数组参数的传递

当参数是数组通过传址方式进行传递。注意：

- 在实参和形参中写数组名，忽略维数的定义，但圆括号不能省。
- 被调过程可通过Lbound和Ubound函数确定实参数组的下、上界。

Lbound和Ubound函数的形式如下：

{L|U}bound(数组名[，维数])

其中：维数指明要测试的是第几维的下标值，缺省是一维数组。

例 6.7 编一函数tim，求任意一维数组中各元素之积。

调用tim,求 $t1=\prod_{i=1}^{5} a_i$ 和 $t2=\prod_{i=3}^{8} b_i$

```
Function tim(a()  As Integer)
    Dim t#，  i%
    t = 1
    For i = Lbound(a) To Ubound(a)
        t = t * a(i)
    Next i
    tim = t
End Function
```

调用：

```
Sub Command1_Click()
    Dim a%(1 To 5)，b%(3 To 8)
    …
    t1# = tim(a())
    t2 #= tim(b())
    Print  t1，  t2
End Sub
```

使用过程注意事项：

1. 确定自定义的过程是子过程还是函数过程

函数过程名有值，子过程名无值。

2. 过程中形参的个数和传递方式的确定

过程中参数的作用是实现过程与调用者的数据通信。

(1)从主调程序获得初值，值传递。

(2)将结果返回给主调程序，地址传递。

3. 实参与形参结合时对应问题

个数、类型、位置、次序一一对应。

形参是值传递，对应实参可以是表达式、常量、数组元素。

形参是地址传递，对应实参只能是简单变量。

数组、记录类型、对象只能是地址传递。

实验6.2子过程DeleStr(s1，ByVal s2 As String)形参的确定

s1，要处理的字符串，从主调程序得初值，删除子串后结果在s1中，所以用地址传递。

s2删除的子串，值传递。

实验6.3函数过程 MaxLength(s)形参的确定

s 要处理的字符串，值传递。

MaxLength函数名，最长的单词长度。

实验6.4回文数的判断中形参的确定

1. 函数过程，用一个形参，值传递对所判断的数字；函数名是否为回文数。

```
Function   IsH(ByVal  ss  As String) As Boolean
```

2. 子过程，用两个形参，值传递对所判断的数字，地址传递是否为回文数。

```
Sub hui(ByVal ss As String, Tag As Boolean)
```

6.4 变量、过程的作用域

作用域：变量、过程随所处的位置不同，可被访问的范围。

1. 过程的作用域

窗体/模块级：加**Private**关键字的过程，只能被定义的窗体或模块中的过程调用。

全局级：加Public关键字（缺省）的过程，可供该应用程序的所有窗体和所有标准模块中的过程调用。

作用范围	模块级		全局级	
	窗体	标准模块	窗体	标准模块
定义方式	过程名前加 Private 例: Private Sub Mysub1(形参表)		过程名前加 Public 或缺省 例 [Public] Sub My2(形参表)	
能否被本模块其它过程调用	能	能	能	能
能否被本应用程序其它模块调用	不能	不能	能,但必须在过程名前加窗体名，例: Call 窗体名.My2(实参表)	能，但过程名必须唯一,否则要加标准模块名,例: Call 标准模块名.My2(实参表)

2. 变量的作用域

局部变量：在过程内用声明的变量，只能在本过程中使用。

窗体/模块级变量：在“通用声明”段中用Dim语句或用Private语句 声明的变量,可被本窗体/模块的任何过程访问。

全局变量：在“通用声明”段中用Public语句声明的变量，可被本应用程序的任何过程或函数访问。

作用范围	局部变量	窗体/模块级变量	全局变量	
			窗体	标准模块
声明方式	Dim ,Static	Dim , Private	Public	
声明位置	在过程中	窗体/模块的“通用声明”段	窗体/模块的“通用声明”段	
被本模块的其它过程存取	不能	能	能	
被其它模块存取	不能	不能	能,但在变量名前加窗体名	能,

例如在下面一个标准模块文件中不同级的变量声明：

```
Public Pa As integer                    ' 全局变量
Private  Mb  As string *10              ' 窗体/模块级变量

Sub  F1( )
    Dim  Fa  As integer                 ' 局部变量
        …
End Sub

Sub  F2( )
    Dim  Fb  As Single                  ' 局部变量
        …
End Sub
```

若在不同级声明相同的变量名,系统按局部、窗体/模块、全局次序访问如:

```
Public Temp As integer                  '全局变量
Sub Form_Load ()
    Dim Temp As Integer                 '局部变量
    Temp=10                             '访问局部变量
    Form1.Temp=20                       '访问全局变量必须加窗体名
    Print Form1.Temp, Temp              '显示 20   10
End Sub
```

3. 静态变量

局部变量声明：

Dim 声明，随过程的调用而分配存贮单元，每次调用都对变量初始化；过程体结束，变量的内容自动消失，存储单元释放。

Static 声明，每次调用过程，变量保持原来的值。

声明形式：Static 变量名 [AS 类型]

Static Function 函数过程名([参数列表]) [As 类型]

Static Sub 子过程名[(参数列表)]

过程名前加Static，表示该过程内的局部变量都是静态变量。

例6.9

```
Private Sub Form_Click()
  Dim  i% ，isum%
  For i = 1 To 5
     isum = sum(i)
     Print    isum
  Next i
End Sub
```

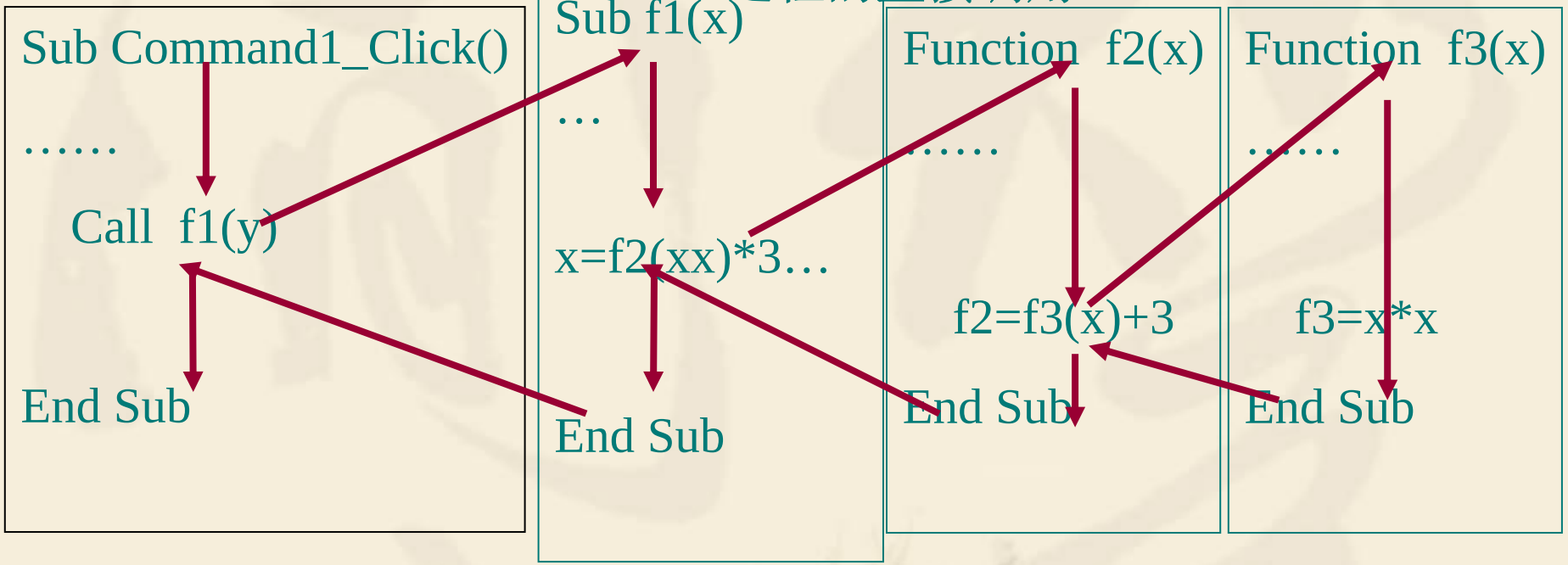
```
Private  Function  sum(n  As  Integer)
     Dim  j  As Integer
     j = j + n
     sum = j
End Function
```

结果为：1，2，3，4，5

Static j As Integer，结果？

6.5 递归

过程的直接调用



过程的递归调用

```
Sub Command1_Click()
……
   Call f1(y)

End Sub
```

```
Sub f1(x)
…
   Call f1(y)

End Sub
```

1. 递归的概念

用自身的结构来描述自身就称为“递归”。例对阶乘的定义：

$$n! = n*(n-1)!$$

$$(n-1)! = (n-1)*(n-2)!$$

2. 递归过程

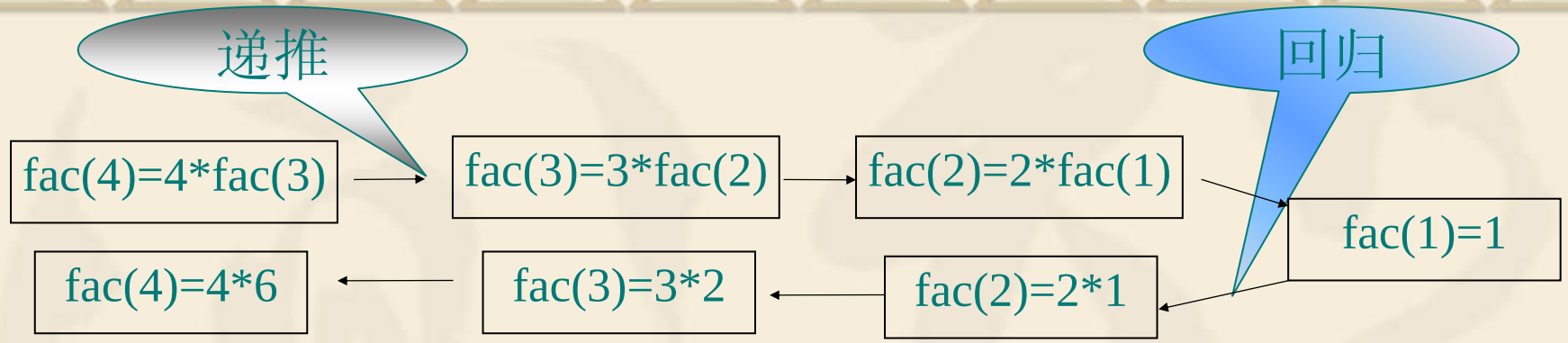
过程在自身定义的内部调用自己。

例6.10 编fac(*n*)=*n*! 的递归函数 $\mathrm{fac}(n)=\begin{cases}1 & n=1\\ n*\mathrm{fac}(n-1) & n>1\end{cases}$

```
Function fac(n As Integer) As Integer
    If n = 1 Then
        fac = 1
    Else
        fac = n * fac(n - 1)
    End If
End Function
```

```
Sub Command1_Click()
    Print “fac(4)=”; fac(4)
End Sub
```

结果：　　fac(4)=24



在递归处理中，用栈来实现。栈中存放形参、局部变量、返回地址。

递推过程：每调用自身，当前参数压栈，直到达到递归结束条件。
回归过程：不断从栈中弹出当前的参数，直到栈空。
递归算法设计简单，但消耗的机时和占据的内存空间比非递归大。
思考：

若上述fac函数中少了： If n = 1 Then fac = 1
即仅有语句： fac = n * fac(n - 1)
或 n<=0

程序运行将造成何结果？由此可见构成递归的结构如下：

递归结束条件及结束时的值；
能用递归形式表示，并且递归向终止条件发展。

例6.11利用递归求最大公约数

$$\gcd(m,n)=\begin{cases} n & m \text{ Mod } n = 0 \\ \gcd(n, m \text{ Mod } n) & m \text{ Mod } n \neq 0 \end{cases}$$

```
Public Function gcd(m As Integer, n As Integer) As Integer
    If (m Mod n) = 0 Then
        gcd = n
    Else
        gcd = gcd(n, m Mod n)
    End If
End Function

Private Sub Form_Click()
    Print gcd(10, 4)
End Sub
```

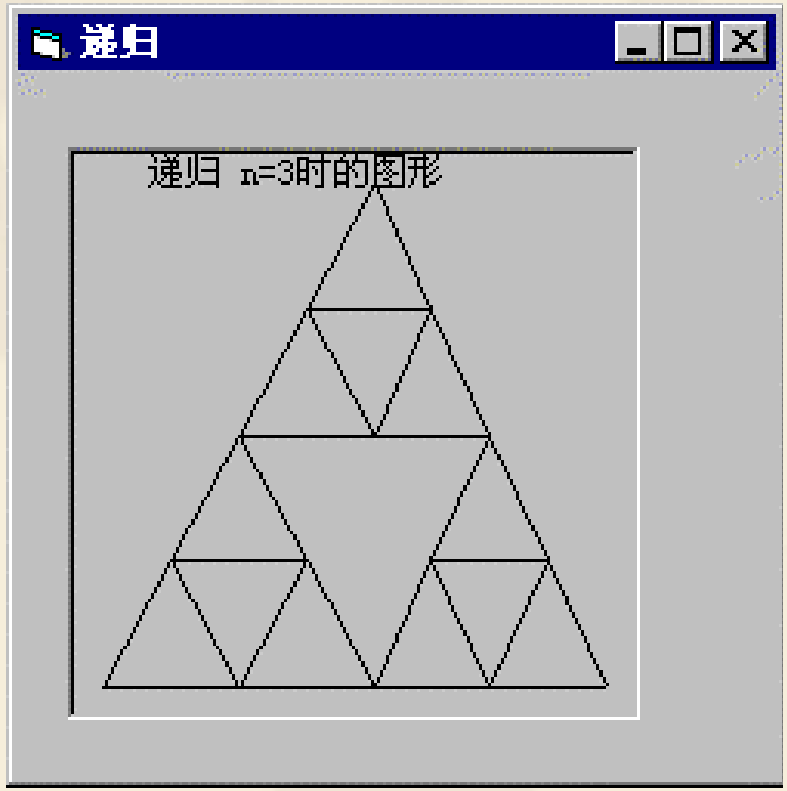
分析以下子过程的功能，当n=100，r=8，结果是多少？

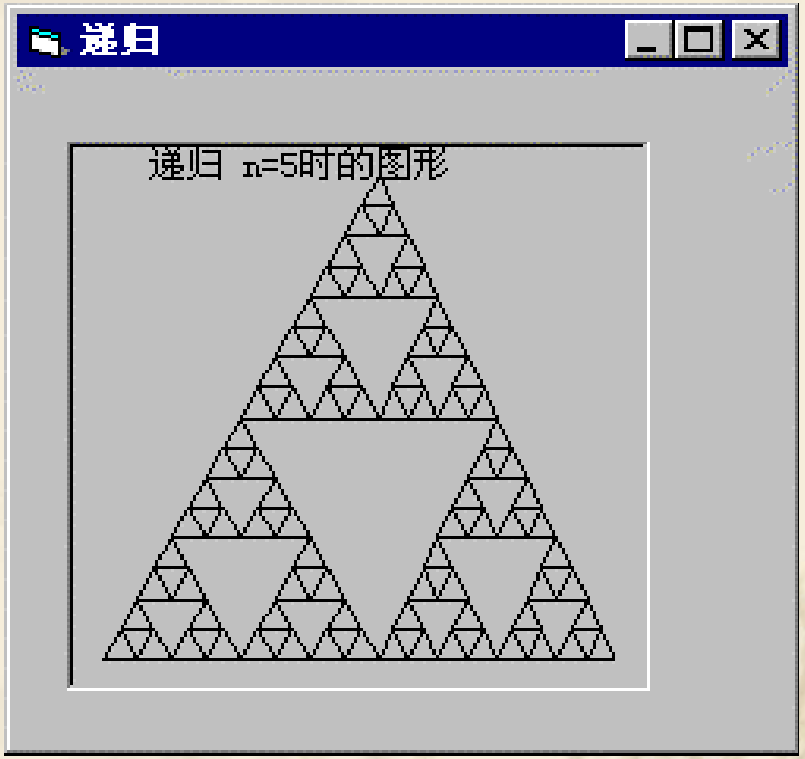
```
Public Sub  f(ByVal n %, ByVal r %)
    If n >r Then Call  f(n \ r, r)        ' ①
    Print n Mod r;                        '②
End Sub
Private Sub Command1_Click()
    Call f(100, 8)
End Sub
```

n	r	返回地址
12	8	2
100	8	2

显示结果 1 4 4

例6.12 打印分形图





递归常见错误：

1. 递归调用出现“栈溢出”

在递归调用时，其中的参数要向终止方向收敛。如下求阶乘的递归函数过程：

```
Public Function fac(n As Integer) As Integer
    If n = 1 Then
       fac = 1
    Else
        fac = n * fac(n - 1)
    End If
End Function
Private Sub Command1_Click()
    Print " fac(5)= " ;  fac(5)
    Print " fac(5)= " ;  fac(-5)      ' 栈溢出
End Sub
```

6.6 常用算法(三)

1. 数制转换

例6.13 将一个十进制整数*m*转换成 *r* (2~16)进制字符串。

方法：将*m*不断除 *r* 取余数，直到商为零，以反序得到结果。

2. 例6.14加密和解密

简单加密的思想是：

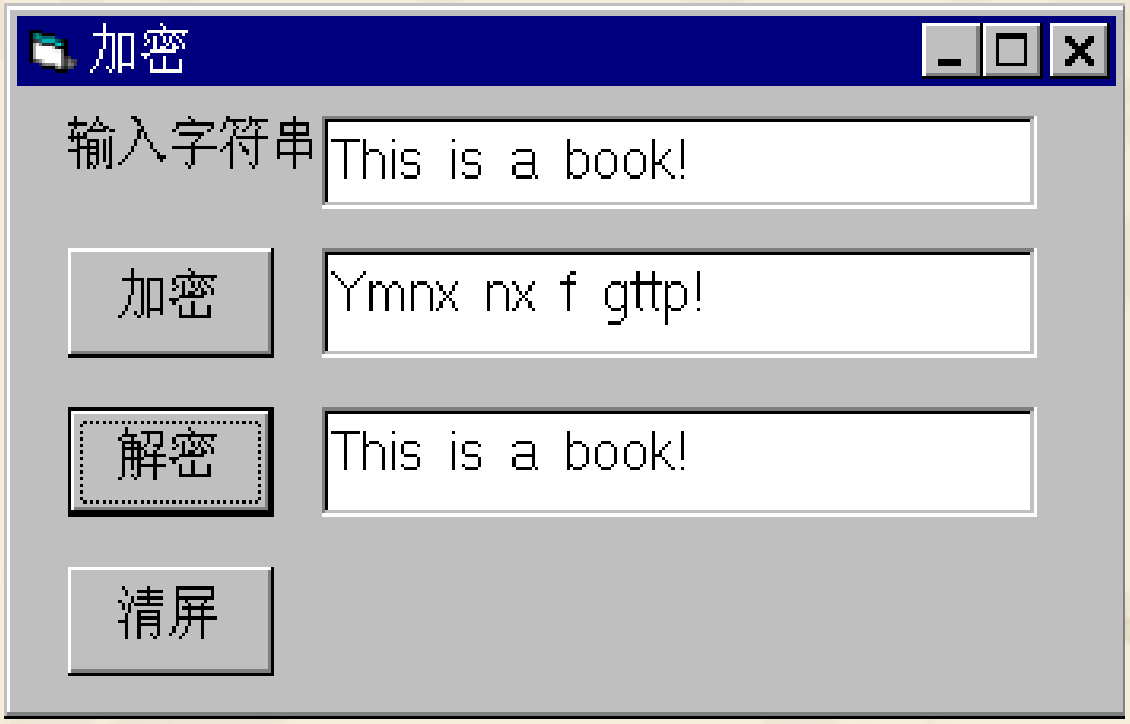
将每个字母C加一序数K，式子 c=chr(Asc(c)+k),

例如序数k为5，这时

“A”➔“F”，“a”➔“f”，“B”➔“G”…

当加序数后的字母超过“Z”或“z”则 c=hr(Asc(c)+k -26)。

解密为加密的逆过程。



返回224

3. 查找

(1)顺序查找 例6.15。

顺序查找根据查找的关键值与数组中的元素逐一比较(数组可无序)

```
Public Sub Search(a() As Variant, ByVal  key As Variant, index%)
    Dim i%
    For i = LBound(a) To UBound(a)
        If key = a(i) Then        ' 找到，元素的下标在index中，结束查找
            index = i
            Exit Sub
        End If
    Next i
    index = -1                    ' 找不到， index形参的值为-1
End Sub
```

平均查找次数 n/2

(2)二分法查找

要查找的数组**必须有序**。

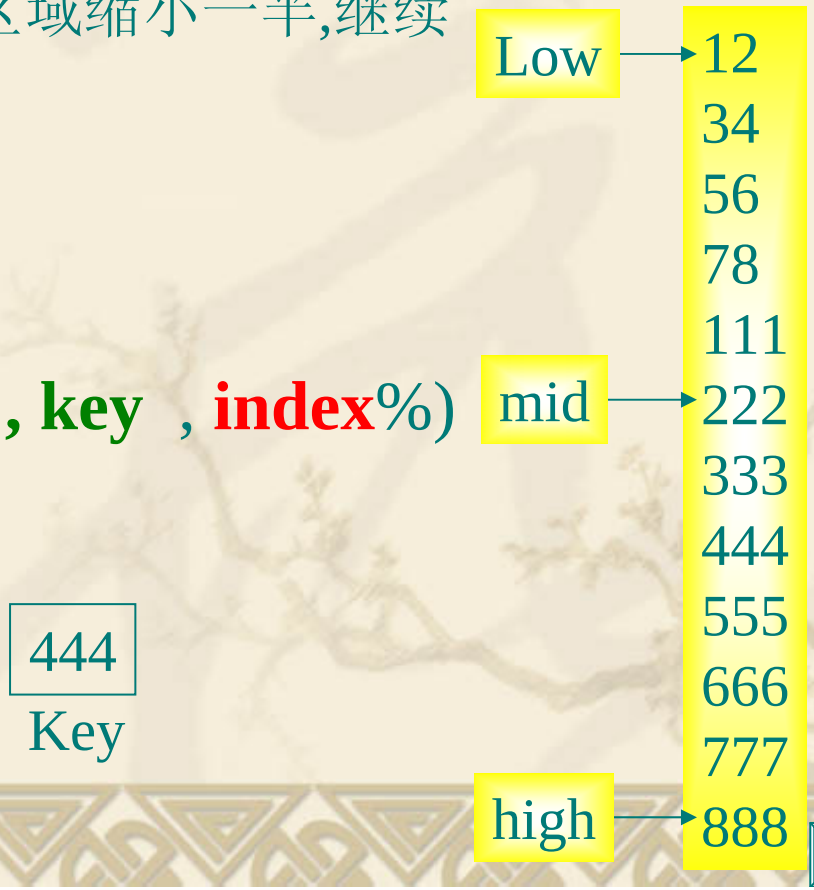
思想：要查找的关键值Key同数组的中间mid项元素比较:

- Key<a(mid) high=mid-1　查找区域缩小一半,继续
- Key=a(mid)　找到 结束
- Key>a(mid) low=mid+1　查找区域缩小一半,继续

直到找到或查找区域中无元素.

本例用递归实现**6.16**

Sub birsearch(a(), **low%, high%, key** , **index**%)



4. 排序

选择、冒泡、插入法排序等。

前两种排序欲排序的数据全部输入后，再进行排序；

插入法排序每输入一项，马上插入到数组应在的位置，数组始终有序。

例6.17　实现的步骤：

(1)输入欲排序的数据项x；在数组a中找x应所处的位置j；

(2)从数组的最后一个元素开始到下标j依次往后移，使j位置空出；

(3)将x放入位置j处，一个数据插入完成；

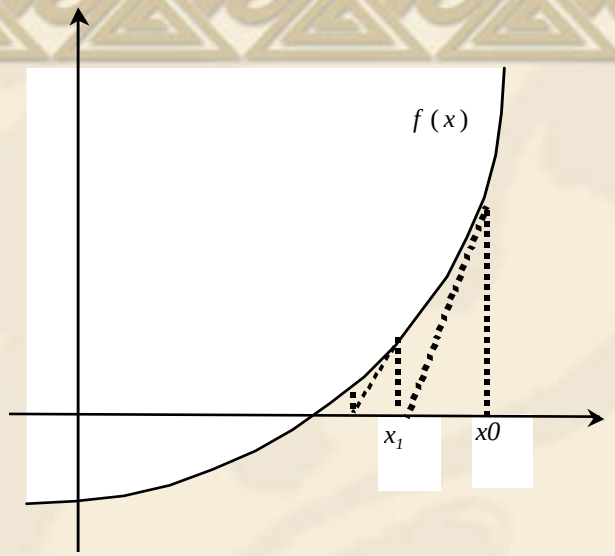
(4)有若干个数重复(1)～(3)。

5. 例6.18 高次方程求根

有牛顿迭代法、二分法、弦截法等

（1）牛顿迭代法

迭代公式：$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

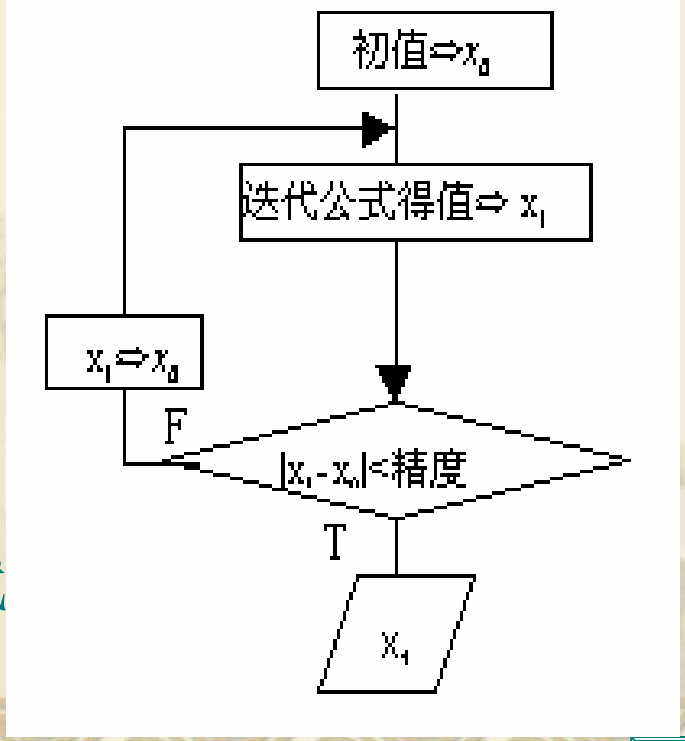


思想：

对方程给定一个初值x_0作为方程的近似根，利用迭代公式，求得x_1，

当 $|x_1 - x_0| \le \varepsilon$

x_1为求得的近似根，否则x_1作为x_0再迭代



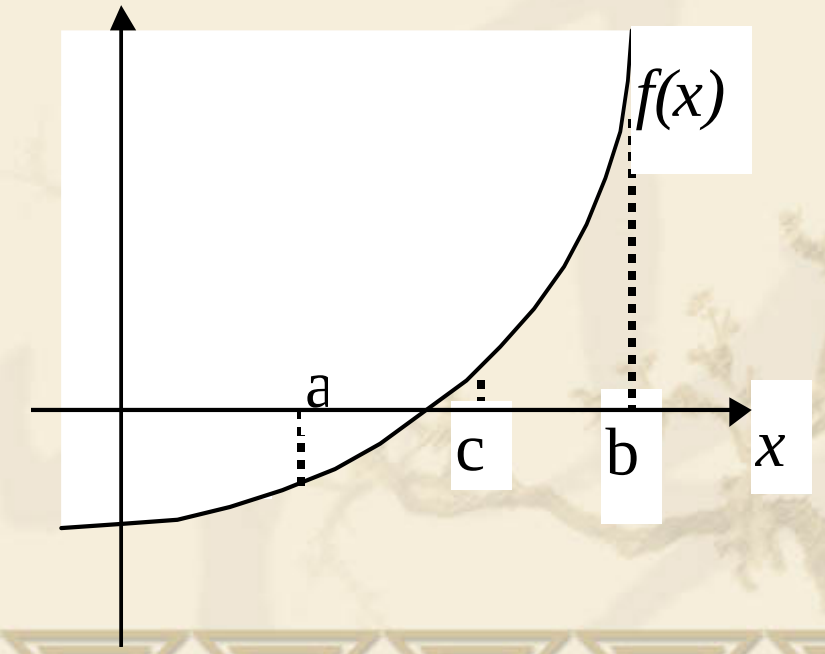
(2) 二分法求根

思想：　已知求根区间[a,b]有一根，每次把求根区间缩小一半，直到找到解或求根区间足够小。

方法：求[a,b]的中点c，判断：

- f(c)=0,，c为求得的根，结束；
- f(a)与f(c)同号，则[a,c]无根，代替a；
- 否则[c,b]无根，c代替b；
- 使求根区间缩小一半，重复上述步骤，直到区间小于精度。

```
Public Function halfRoot(ByVal a!, ByVal b!)
  Dim c!
  Do While Abs(b - a) > 0.00001
    c = (a + b) / 2
    If f(c) = 0 Then
       Exit  Do
    ElseIf f(a) * f(c) > 0 Then
       a = c
    Else
       b = c
    End If
  Loop
  halfRoot = c
End Function
```



6. 例6.19数值积分

有矩形法、梯形法、抛物线法（又称辛卜生法）等。

梯形法积分的思想是：

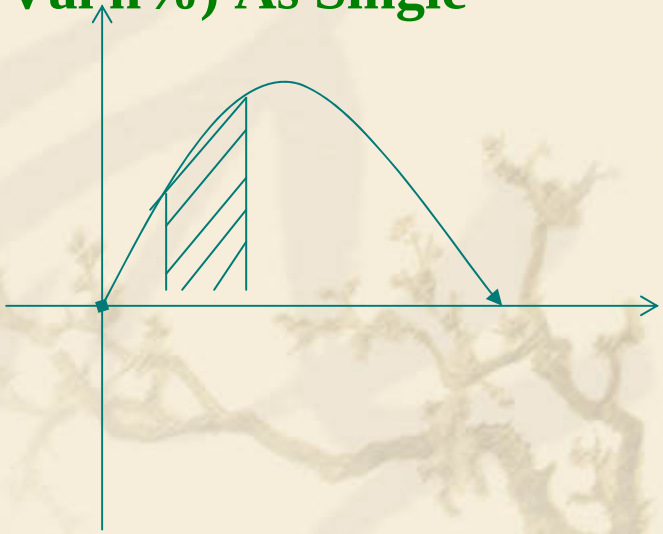
将积分区间[a，b]n等分，小区间的长度为，$h = \dfrac{b-a}{n}$

第i块小矩形的近似面积为：$s_i = \dfrac{f(x_i)+f(x_{i+1})}{2}\times h$

整个积分的结果为这n块小面积的累加，即：

$$S \approx \sum_{i=1}^{n} \frac{f(x_i)+f(x_{i+1})}{2}\times h \approx \left\{\frac{1}{2}(f(a)+f(b))+\sum_{i=1}^{n-1} f(x_i)\right\}\times h$$

```
Public Function trapez(ByVal a!, ByVal b!, ByVal n%) As Single
   Dim sum!, h!, x!
   h = (b - a) / n
   sum = (f(a) + f(b)) / 2
   For i = 1 To n - 1
      x = a + i * h
      sum = sum + f(x)
   Next i
  trapez = sum * h
End Function
```



7. 字符串处理

例6.20 编写一个英文打字训练的程序.要求如下：

(1)在标签框内随机产生30个字母的范文；

(2)当焦点进入文本框时开始计时，并显示当时时间；

(3)在键入文本框按产生的范文输入相应的字母；

(4)当键入满了30个字母后结束计时，禁止向文本框输入内容，与范文逐一比较，显示打字的速度和正确率。

6.7 重点和难点

1. **确定自定义的过程是子过程还是函数过程**

函数过程名有值，子过程名无值。

过程有一个返回值，则使用函数过程；

若返回多个值或无返回值，一般使用子过程。

2. **过程中形参的个数和传递方式的确定**

过程中参数的作用是实现过程与调用者的数据通信。

(1)从主调程序获得初值，值传递。

(2)将结果返回给主调程序，地址传递。

3. **实参与形参结合时对应问题**

个数、类型、位置、次序一一对应。

形参是值传递，对应实参可以是表达式、常量、数组元素。

形参是地址传递，对应实参只能是简单变量。

数组、记录类型、对象只能是地址传递。

4. **变量的作用域问题**

局部变量、静态变量、全局变量特点、作用

5. **递归调用出现“栈溢出”**

递归过程中有终止的条件和终止时的值或某种操作；

每递归调用一次，其中的参数要向终止方向收敛。

第七章 常 用 控 件

1. 标准控件

内部控件

出现在工具箱上的控件20个。

2. ActiveX控件

ActiveX部件：是可以重复使用的编程代码和数据。

是由用ActiveX技术创建的一个或多个对象所组成 。

ActiveX部件文件：扩展名OCX，在Windows的SYSTEM目录中。

ActiveX部件
- ActiveX控件
- ActiveX文档
- ActiveX DLL
- ActiveX EXE

ActiveX控件添加到工具箱：

工程/部件 ⟶ 选定控件

常用ActiveX控件所在的文件：

ActiveX控件	ActiveX部件	文件名
通用对话框(CommonDialog)	Microsoft Common Dialog Control 6.0	COMDLG32.OCX
ToolBar	Microsoft Windows Common Control 6.0	MSCOMCTL.OCX
StatusBar		
ProgressBar		
Slider		
Animation	Microsoft Windows Common Control-2 6.0	MSCOMCT2.OCX
UpDown		

ActiveX控件与ActiveX DLL/EXE的区别：

ActiveX控件：

有界面；

用“工程/部件”命令加载；

工具箱上有图标。

ActiveXDLL/EXE：

没有界面；

用“工程/引用”设置引用；

工具箱上没有图标。

3. 可插入对象

Windows应用程序的对象；

可插入对象可添加到工具箱上；

同标准控件一样使用。

例如：Microsoft Excel工作表。

7.1 单选钮和复选框

Caption属性： 文本标题。

Alignment属性

0：控件钮在左边，标题显示在右边。

1：控件钮在右边，标题显示在左边。

Value属性

单选钮(逻辑型)　检查框（数值型）

True：选定　　0--Unchecked：未被选定

False：未选定　1--Checked：选定

2--Grayed：灰色，禁止选择

Style属性

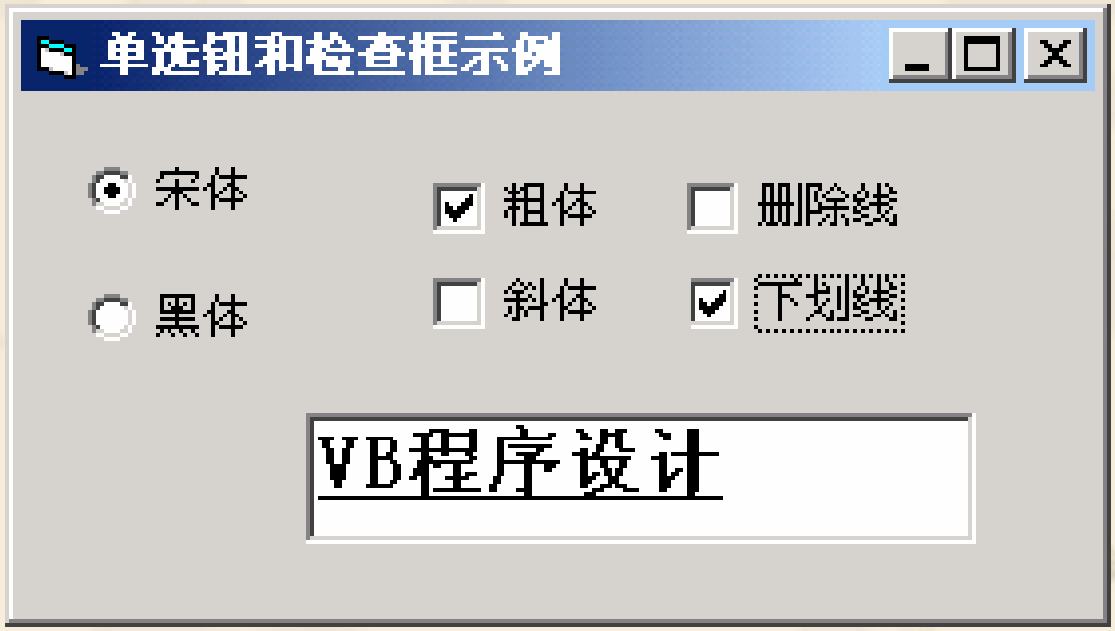
0--Standard：标准方式

1--Graphical：图形方式

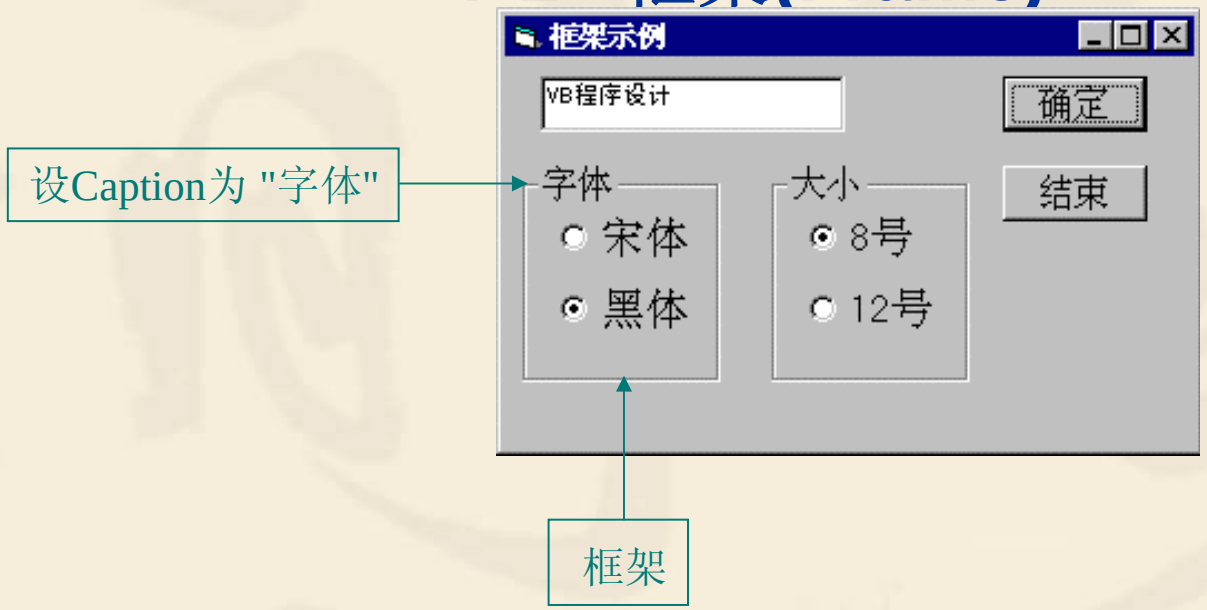
事件：Click

因为单击时自动改变状态，故不需要编写过程。

例7.1 用单选钮和检查框设置文本框的字体。



7.2 框架(Frame)



框架内控件的创建方法：

方法1：单击工具箱上的工具，然后用出现的“+”指针，在框架中适当位置拖拉出适当大小的控件。

不能使用双击工具箱上图标的自动方式。

方法2：将控件“剪切”到剪贴板，然后粘贴(Ctrl+V)到框架。

Caption属性：框架标题

Enabled属性

False：标题呈灰色，不允许对框架内的对象进行操作。

Visible属性

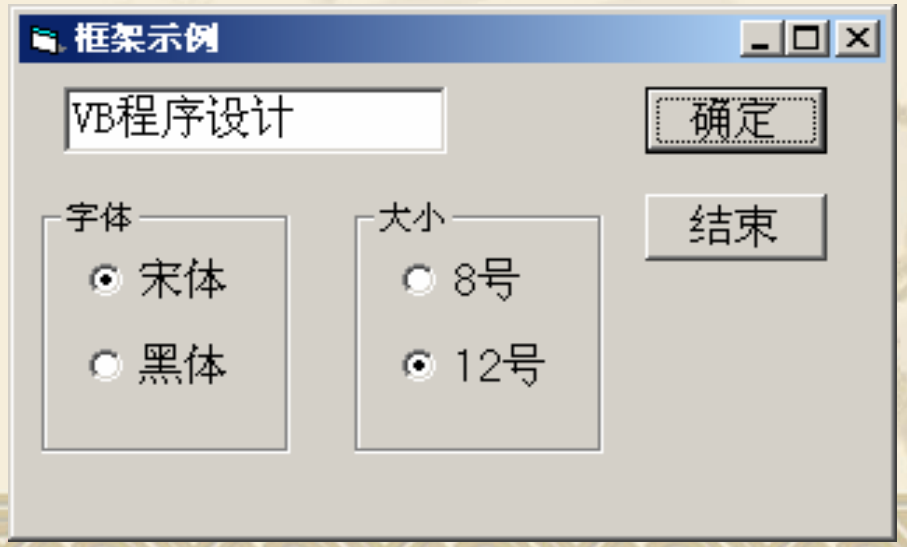
True：框架及其控件可见。

False：框架及其控件被隐含起来。

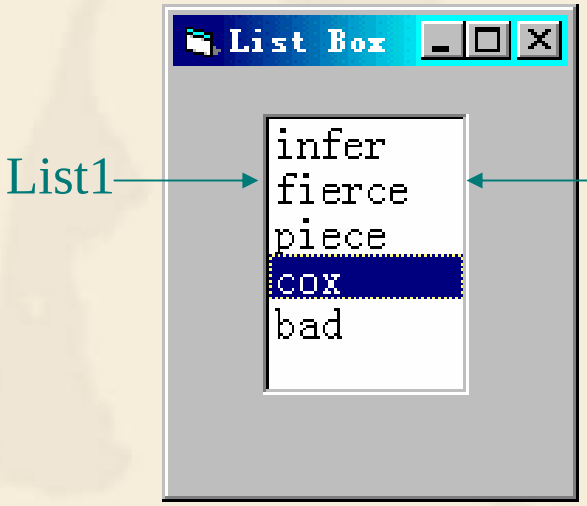
Click、DblClick事件：

一般不需要编写框架的事件过程。

例7.2 框架用法示例。



7.3 列表框和组合框



列表框各主要属性的值：

List1.ListIndex = 3 (下标从0开始的)

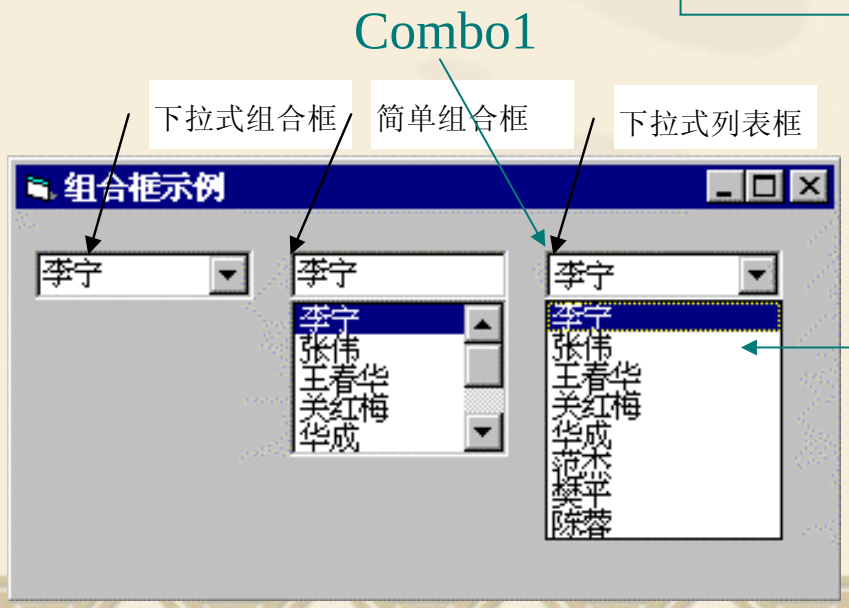
List1. ListCount = 5

List1. Selected(3) = True，其余为False。

List1. Sorted = False，没有排序。

List1. Text 为 “cox”，

与List1. List(List1. ListIndex)相等



列表框各主要属性的值：

List1.ListIndex = 0

List1. ListCount = 8

List1. Selected(3) = True

其余为False。

List1. Sorted = False

List1. Text 为 “李宁”

1. 共有的重要属性

(*P：可在程序中设置或引用，D：可在设计状态设置。*)

List ：PD

字符型数组，存放列表框的项目，下标是从0开始。

ListIndex：P

选中的项目的序号，没有项目被选定时为-1

ListCount：P

项目的数量，ListCount-1是最后一项的下标。

Sorted：D

True：按字母顺序排列。

False：按加入先后顺序排列。

Text：P

列表项中被选定的内容，

List 1.List(List 1.ListIndex) = List1.Text。

2. 列表框的特有属性

Selected：P

逻辑数组。

Selected(i)的值为True表示第i+1项被选中。

MultiSelect

0-None：禁止多项选择。

1-Simple：简单多项选择。

2-Extended：扩展多项选择。

3. 列表框的特有属性

Style

类型	Style	输入
下拉式组合框	0	能
简单组合框	1	能
下拉式列表框	2	不能

4. 方法

AddItem

对象.AddItem item [, index]

RemoveItem

对象.RemoveItem index

Clear

对象.Clear

5. 事件

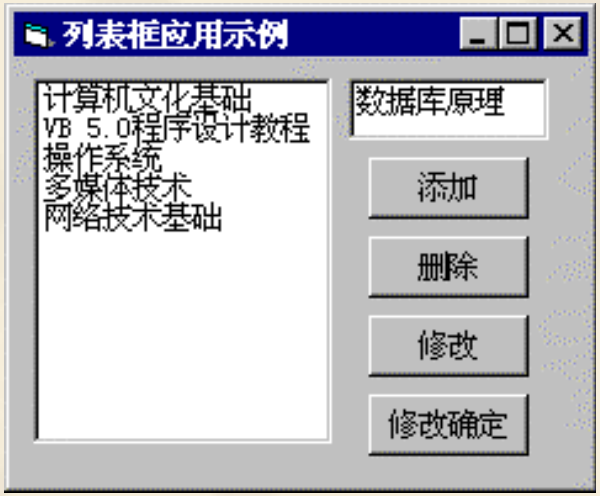
列表框：Click、DblClick。

组合框：Click，只有简单组合框才有DblClick事件。

一般不需要编写Click事件过程。

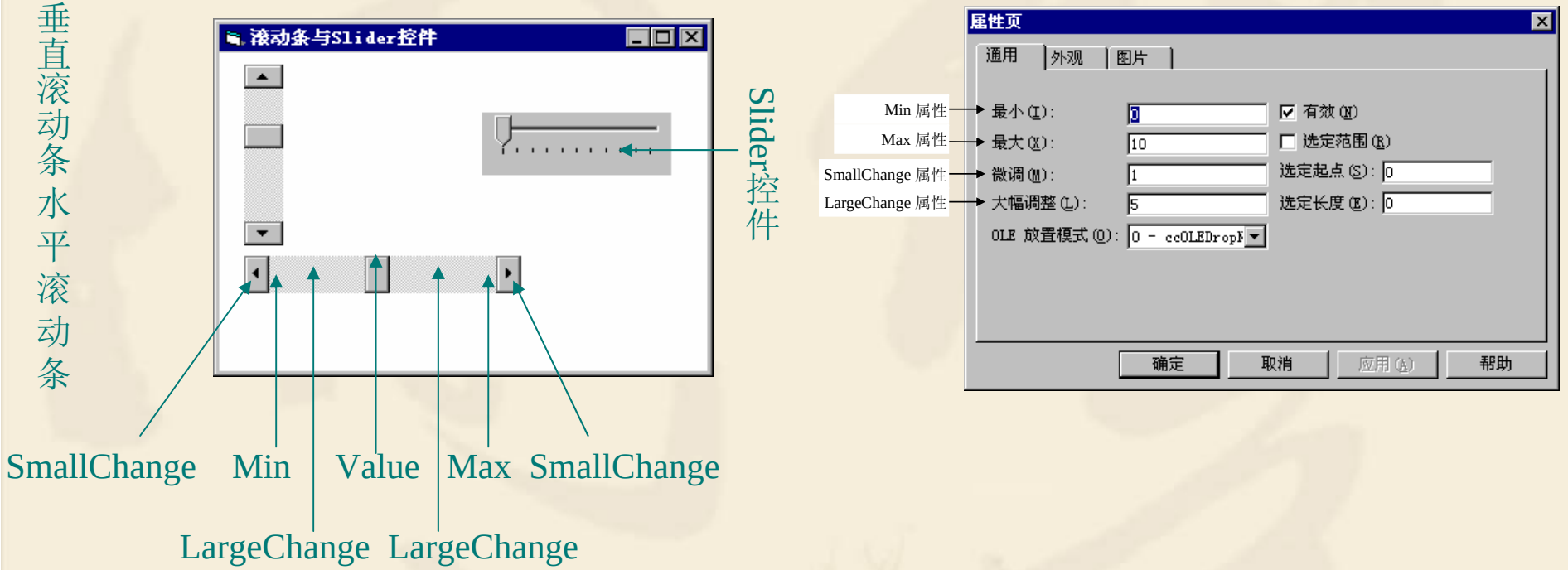
通常在单击命令按钮或发生DblClick事件时才读取Text属性。

例7.3 对列表框进行项目添加、修改和删除操作。



例7.4 对列表框进行项目添加、修改和删除操作。

7.4 滚动条和Slider控件



共同具有的重要属性

Max：最大值　　-32 768~32 767

Min：最小值　　-32 768~32 767

SmallChange　最小变动值，单击箭头时移动的增量值。

LargeChange　最大变动值，单击空白处时移动的增量值。

Value　　滑块所处位置所代表的值。

事件：

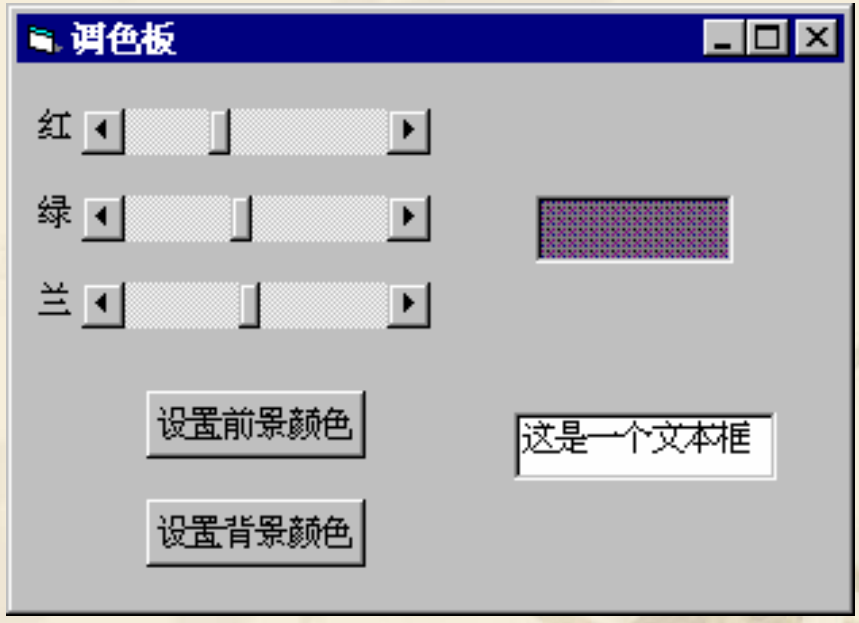
Scroll：拖动滑块时会触发Scroll事件。

Change：Value属性改变时触发Change事件。

例7.5 用一个文本框(txtSpeed)显示滚动条(hsbSpeed)滑块当前位置所代表的值。

例7.6 调色板程序。


```
Sub hsbSpeed_Change()
    Text1.Text= HScroll1.Value
End Sub
```



例7.7 用Slider控件设置文本框中的字体大小。

7.5 时钟(Timer)

时钟控件以Interval为时间间隔产生 Timer事件。

属性

Interval属性

单位：ms(0.001s)，0.5秒是500。

Interval＝0：屏蔽计时器。

Enabled属性

True：有效计时

False：停止时钟工作

事件

Timer

例7.8 定时的闹钟

例7.9 蝴蝶飞舞

7.6 ProgressBar控件

位于Microsoft Windows Common Control 6.0部件。

重要属性：

Max、Min：该控件的界限。

Value：决定控件被填充多少。

例7.10 用进度条指示一个大数组的计算进度。

7.7 UpDown控件

位于Microsoft Windows Common Control-2 6.0。

通常与伙伴控件“捆绑”在一起使用。

UpDown应用示例。

7.8 Animation控件

位于Microsoft Windows Common Control-2 6.0。

属性：

Center：决定动画是否在控件的中央播放。

AutoPlay属性：决定在用Open方法打开文件时是否自动播放。

方法：

Open：打开文件；

Play：播放动画；

Stop：停止播放；

Close：关闭文件。

例7.11 为例7.10配上动画。

7.9 SSTab控件

位于Microsoft Tabbed Dialog Control 6.0。

重要属性：

Style：选项卡样式；

Tabs： 选项卡总数；

TabsPerRow：每一行选项卡的数目；

Rows：选项卡总行数；

TabOrientation：选项卡的位置；

ShowFocusRect：决定选项卡上的焦点矩形是否可视；

Tab：当前选项卡的序号。序号从0开始，如果Tab为1，则第 二个选项卡为当前活动的选项卡。

例7.12 选项卡制作示例。

7.10 鼠标器和键盘

1. 鼠标器事件

注意：鼠标事件发生在什么对象上，是窗体上还是控件上。

MouseDown事件

```
Sub Form_MouseDown(Button As Integer, Shift As Integer,
        X As Single, Y As Single)（发生在窗体上的事件过程）
```

MouseUp事件

```
Sub Form_MouseUp(Button As Integer, Shift As Integer,
        X As Single, Y As Single) （发生在窗体上的事件过程）
```

MouseMove事件

```
Sub Form_MouseMove(Button As Integer, Shift As Integer,
        X As Single, Y As Single) （发生在窗体上的事件过程）
```

(1) Button参数

	b_2	b_1	b_0

B_0为1：按下了左键；

B_1为1：按下了右键；

B_2为1：按下了中键。

例如：Button为2(010B)，即B_1 为1，表示按下了右键；

如果按了左键，则**B_0**为1， **B_1**和**B_2**为0， Button为1(001B)

```
If  Shift = 1  Then    '或者  If  Shift = vbLeftButton  Then
        …
        ' 这是按了左健后执行的代码
        …
Endif
```

使用符号常数：

1—vbLeftButton：用户按下左键触发了鼠标事件；

2—vbRightButton：用户按下右键触发了鼠标事件；

4—vbMiddleButton：用户按下中键触发了鼠标事件。

(2) Shift参数

	b_2	b_1	b_0

B_0为1：按下了Shift键；

B_1为1：按下了Ctrl键；

B_2为1：按下了Alt键。

例如：Button为2(010B)，即B_1 为1，表示仅按下了Shift键；

如果同时按了Ctrl和Shift键，则**B_0**和**B_1**为1，**B_2**为0，Button为3(011B)

注意： 可能同时按下两个或三个键。如果Button<>1成立，并不表示没有按下Shift，因为可能其他键也被按下了。如果要测试按下了某个键，则应用and进行位运算。例如Button and 1成立，表示肯定按下了Shift（可能其他键也被按下了）。

```
If  Shift = 1 and Button = 2 Then
    …
' 这是仅按住Ctrl了健单击鼠标后执行的代码
    …
End If
```

思考： Shift = 1 and Button and 2 表示什么意义？

符号常数：

1—vbShiftMask；

2—vbCtrlMask；

4—vbAltMask 。

Shfit And vbCtrlMask为真：

按下了Ctrl键；

CBool(Shift And vbCtrlMask) Or CBool(Shift And vbShiftMask)为真， 按下了Ctrl键和Shift键

(3) x,y：鼠标的当前位置

例7.13 显示鼠标器指针所指的位置。

例7.14 画圆程序。

按下鼠标右键画圆，按下鼠标左键移动时画线。

在画线时用note14.ico作为鼠标的指针。

2. 键盘事件

KeyPress事件过程

Sub Form_KeyPress(KeyAscii As Integer)

Sub object_KeyPress([index As Integer,]KeyAscii As Integer)

KeyDown事件过程

Sub Form_KeyDown(keycode As Integer, shift As Integer)

Sub object_KeyDown([index As Integer,]keycode As Integer, shift As Integer)

KeyUp和**KeyDown**事件过程

Sub Form_KeyUp(keycode As Integer, shift As Integer)

Sub object_KeyUp([index As Integer,]keycode As Integer, shift As Integer)

参数说明

Shift与鼠标事件过程Shift相同。

KeyCode：键盘扫描码；

区别

	KeyPress	KeyDown和KeyUp
事件发生的时间	输入一个ASCII字符	按任意一个键
参数值	KeyAscii接收到字符的ASCII值	KeyCode接收到键的扫描码
按Shift+A时事件发生的次数	事件发生一次	事件发生两次
按Shift+A时参数值(键盘处于大写状态)	97	第一次是17 第二次是65
按Shift+A时参数值(键盘处于小写状态)	65	第一次是17 第二次是65

假定窗体KeyPreview为True，并有下面事件过程，则当文本框中输入“1”时，实际上是得到的是“3”

```
Sub Form_KeyPress(KeyAscii AsInteger)
    KeyAscii = KeyAscii + 1
End Sub
```

```
Sub Text1_KeyPress(KeyAscii As Integer)
    KeyAscii = KeyAscii + 1
End Sub
```

？ 窗体KeyPreview为False时在文本框中输入“1”时，实际上得到的是什么。

对输入的数据进行验证、限制和修改

1. 修改输入数据

接收大写字符。

将KeyPreview设置为True时：

```
Sub Form_KeyPress(KeyAscii As Integer)
    If KeyAscii >= Asc("a") And KeyAscii <= Asc("z") Then
        KeyAscii = KeyAscii + Asc("A") - Asc("a")
    End If
End Sub
```

如果把它改为某个控件的事件过程，效果一样吗？

2. 限制数据输入

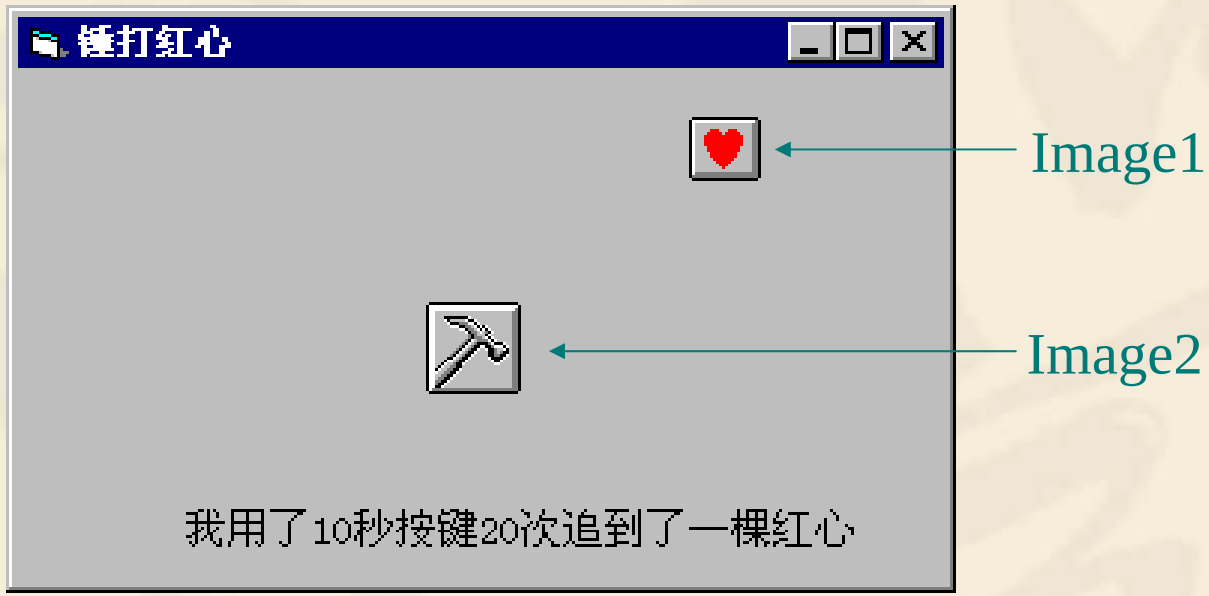
文本框只能接收“0”～“9”的数字字符。

```
Sub txtExample_KeyPress(KeyAscii As Integer)
  If KeyAscii < 48 Or KeyAscii > 57 Then
      KeyAscii = 0
  End If
End Sub
```

例7.15 编写一个程序，当按下Alt+F5时终止程序的运行。

```
'先把窗体的KeyPreview设置为True，再编写如下的程序：
Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    '按下Alt键时，Shift的值为4
    If (KeyCode = vbKeyF5) And (Shift = 4) Then
     End
  End If
End Sub
```

例7.16 “锤打红心”游戏



(1) 利用上、下、左、右四个箭头键控制“铁锤”：

“←”：37(&H25)　　“↑”：38(&H26)

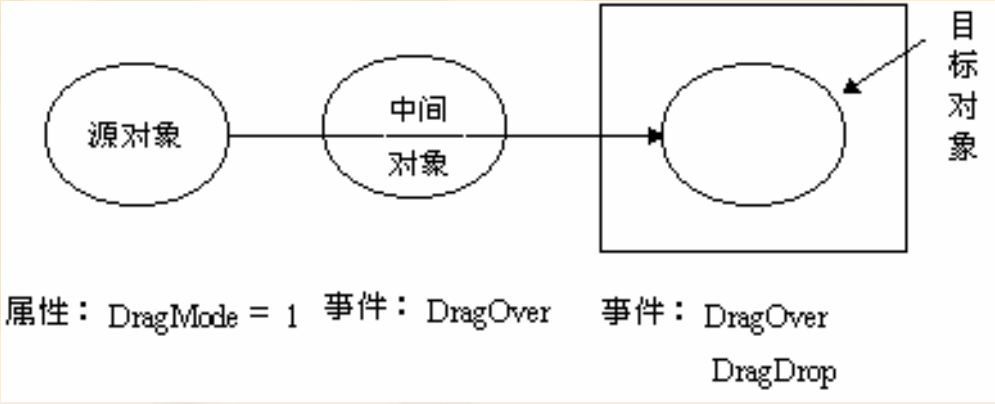
“→”：39(&H27)　　“↓”：40(&H28)

(2) 如果Abs(Image1.Left - Image2.left < 300) And Abs(Image1.Top - Image2.Top < 320)成立，则认为是重叠，“铁锤”锤打到“红心”。

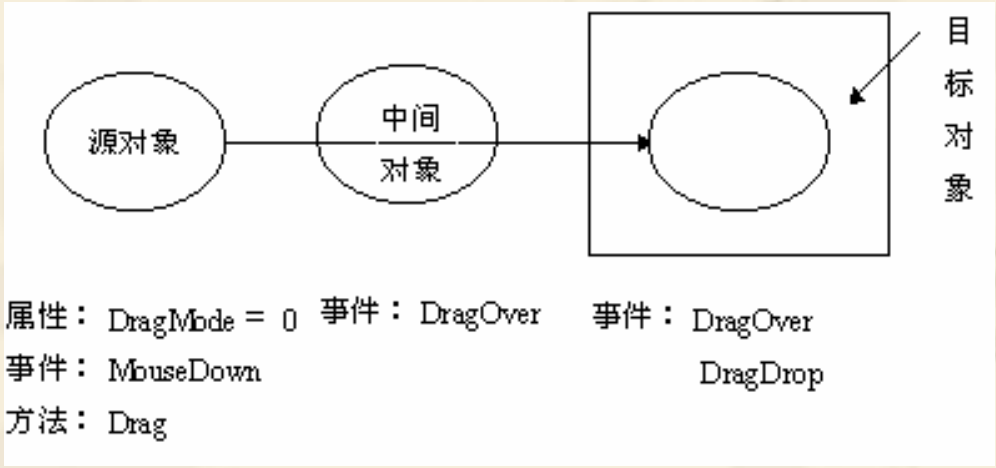
(3) 在窗体上的时钟控件(Timer1)过程中控制“红心”移动。Timer1的Interval属性为200，即每1秒产生5个Timer事件。

3. 拖放

自动拖放



手工拖放



（1DragMode属性

0：(缺省)，手工拖动模式；

1：自动拖动模式 。

（ 2 ）DragIcon属性

拖动过程中显示的图标(Ico或Cur文件)。

lblExample1.DragIcon=LoadPicture ("C:\Icons\Mail.ico")

lblExample2.DragIcon=picIcon.Picture

（ 3 ） Drag方法

当DragMode为0时，需用Drag方法启动拖放。

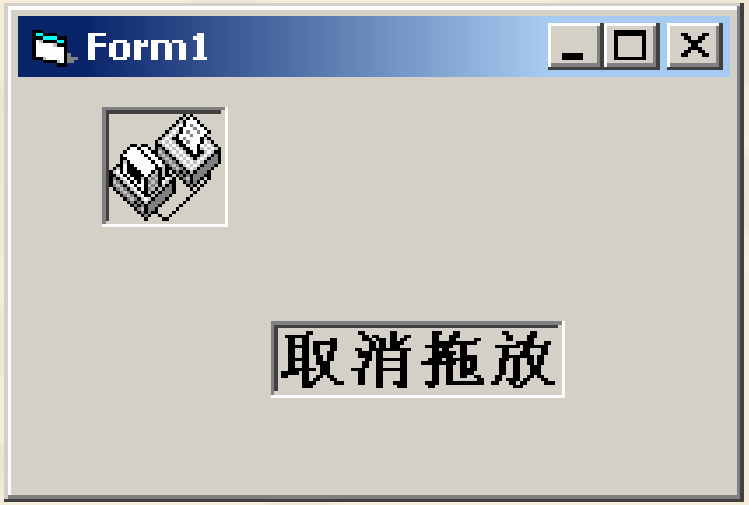
[控件名称.]Drag 参数

0：开始拖放操作；

1(省略)：结束拖放操作；

2：取消拖放操作。

例7.17 拖放应用示例。



例7.18 拖放应用示例。

4. OLE 拖放

OLEDragMode属性(源控件设置):

0----Manual：缺省，手工；

1----Automatic：自动。

OLEDropMode属性(目标控件设置):

0----None：缺省，目标控件不接受“放”；

1----Manual：手工实现“放”操作；

2----Automatic：自动实现“放”。

完全支持自动OLE拖放：

PictureBox　Image　TextBox。

示例

支持自动“拖”操作，不支持自动“放”：

ComboBox　FileListBox　DirListBox和ListBox。

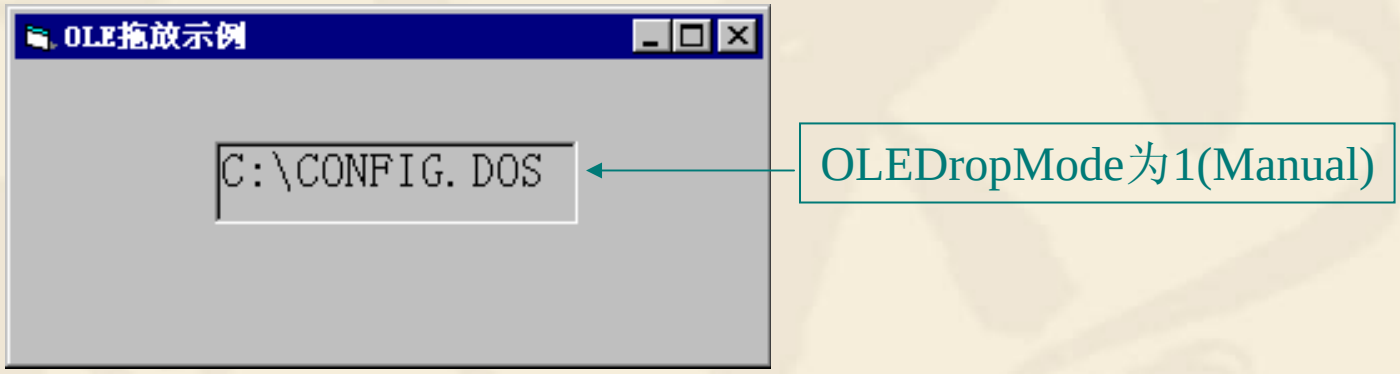
示例

只支持OLE拖放事件的控件有：

CheckBox　Frame　OptionButton、

Label　DriveListBox　CommandButton

例7.19从Windows资源管理器把文件的文件名拖到标签上。




```
Sub Label1_OLEDragDrop(Data As DataObject, Effect As Long, _
                       Button As Integer,    Shift As Integer,
                       X As Single, Y As Single)
    Label1.Caption = Data.Files(1)
    ' 在资源管理器中选定文件且拖出时，系统就把所选定的文件名
    ' 保存在Data对象的Files属性中，Files属性实质上是一个数组
    'Label1.Caption = Data.Files(1)将保存的第一个文件名显示'在标签中
End Sub
```

第八章 界面设计

（3学时）

8.1 通用对话框

8.2 菜单设计

8.3 多重窗体和多文档界面

8.4 工具栏和状态栏

8.5 RichTextBox控件

8.6 应用程序向导

8.1 通用对话框(CommonDialog)

通用对话框添加到工具箱：

工程/部件 → 选定部件标签 → 选定Microsoft CommDialog Control 6.0 → 确定

打开通用对话框：

通用对话框的类型	Action	方法
打开(Open)	1	ShowOpen
另存为(Save As)	2	ShowSave
颜色(Color)	3	ShowColor
字体(Font)	4	ShowFont
打印机(Printer)	5	ShowPrinter
和帮助(Help)	6	ShowHelp

属性

Action：打开通用对话框。

DialogTiltle：对通用话框标题。

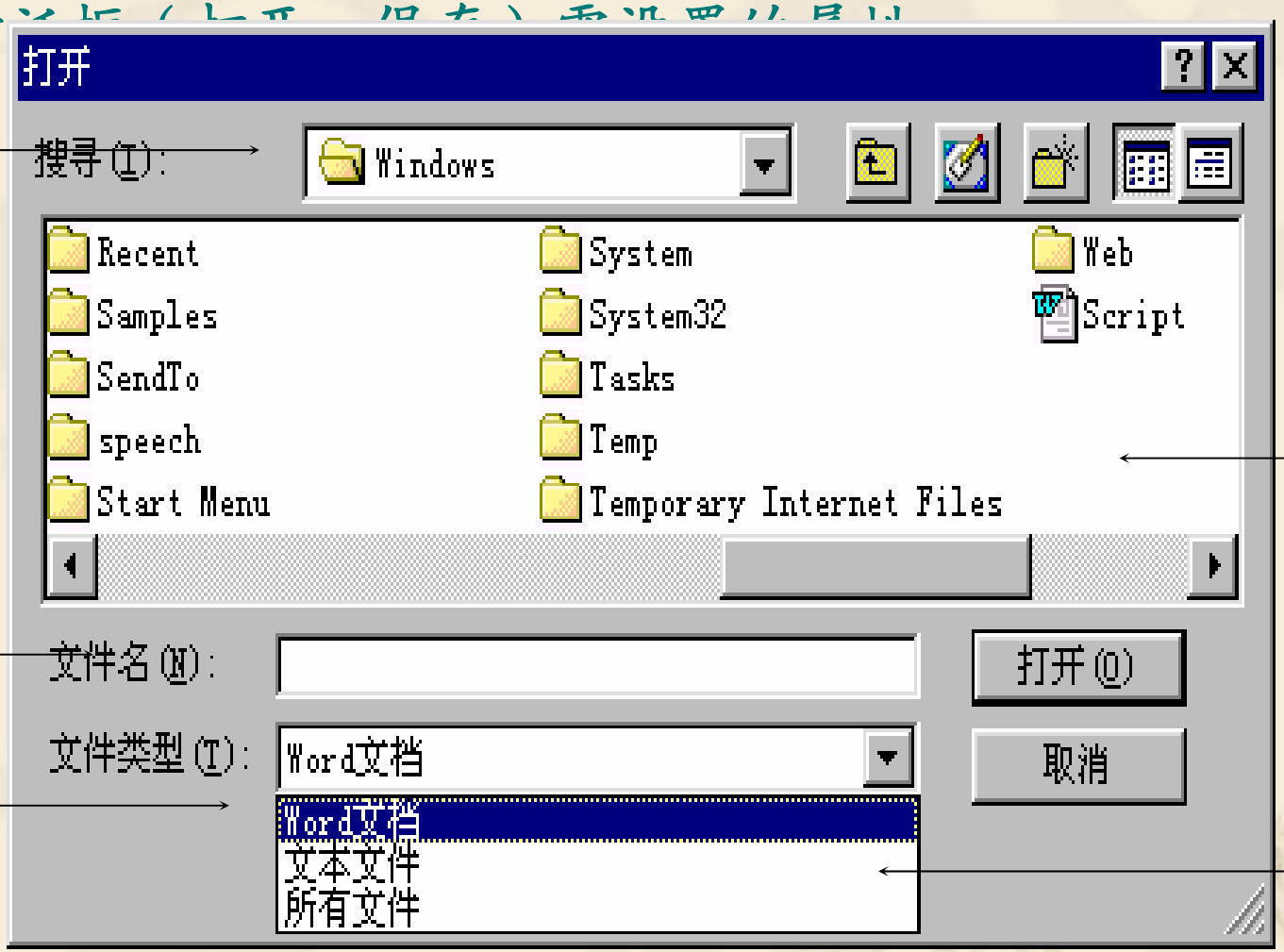
CancelError

True：选择“取消”按钮，出现错误警告；

Err.Number置为32755(cdCancel)。

False(缺省)：选择“取消”按钮，没有错误警告。

1. 文件对


InitDir
FileName
FilterIndex
Filter
打开
搜寻(I):
Windows
Recent
Samples
SendTo
speech
Start Menu
System
System32
Tasks
Temp
Temporary Internet Files
Web
Script
文件名(N):
打开(O)
文件类型(T):
Word文档
取消
Word文档
文本文件
所有文件

“打开”文件对话框属性页

属性

FileName：包含路径；

FileTitle：不包含路径。

Filter：例如，

Documents(*.DOC)|*.DOC|Text Files(*.TXT)|*.txt|All Files|*.*

FilterIndex：决定在文件类型列表框中显示第几组类型的文件。

InitDir：初始化路径。

例8.1 用命令按钮的Click事件显示文件打开对话框。

```
CommonDialog1.InitDir= "C:\Windows "        ' 设置初始目录
CommonDialog1.Filter = "文本文件 | *.Txt"   ' 过滤文件类型
CommonDialog1.CancelError = True            ' 控制取消按钮
CommonDialog1. ShowOpen                     ' 打开对话框
```

2. “另存为”对话框

没有提供真正的存储文件操作，仍需要编程完成储存操作。

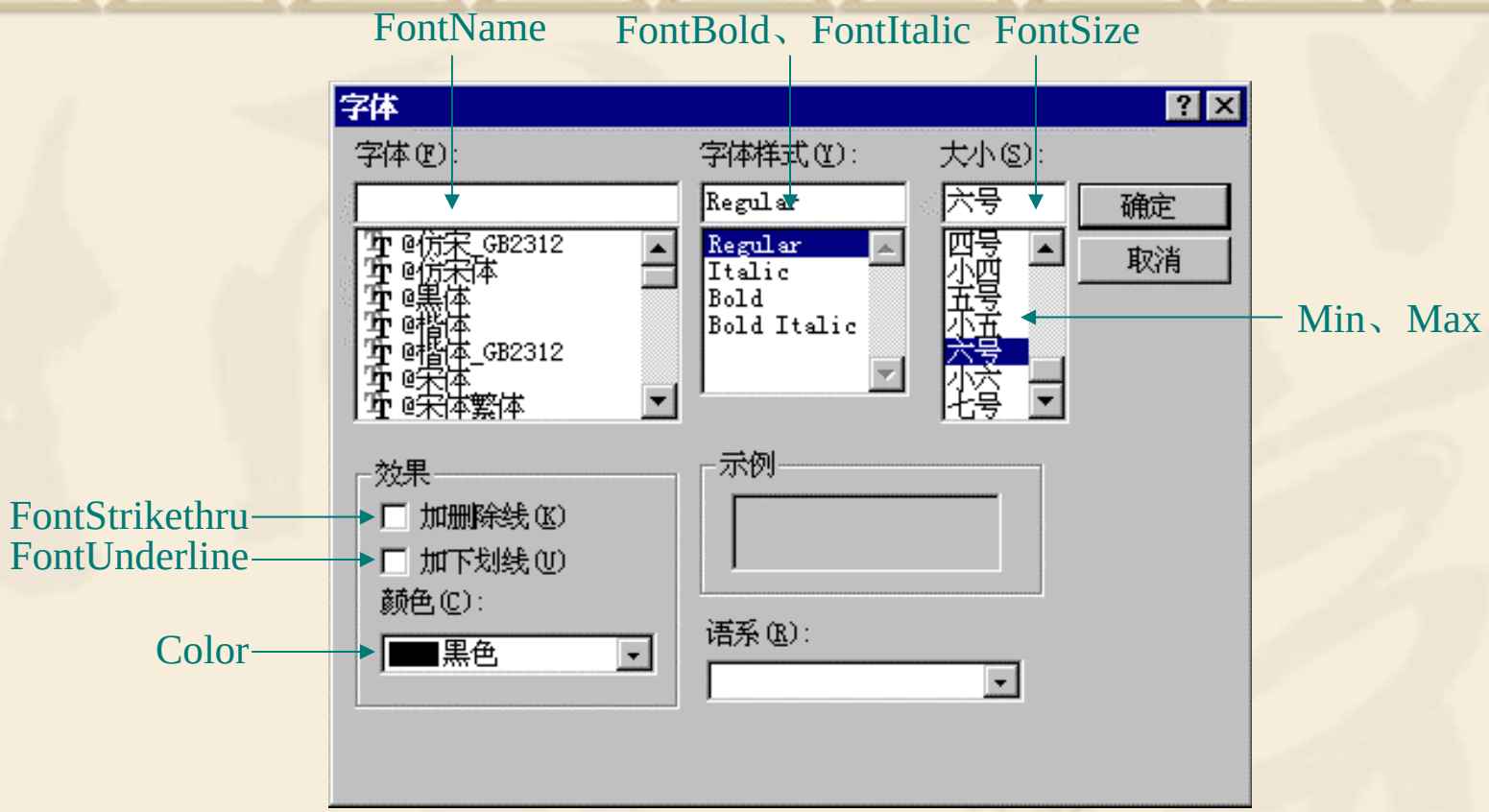
属性　DefaultExt：缺省扩展名。

3. “颜色”对话框

属性　Color：返回或设置选定的颜色。

例8.2　“颜色”对话框的使用。

4. “字体”对话框



Flags属性：指示所显示的字体类型，*必须设置*。

cdlCFScreenFonts	&H1	屏幕字体
cdlCFPrinterFonts	&H2	打印机字体
cdlCFBoth&H3	打印机字体和屏幕字体。	
cdlCFEffects	&H100	显示删除线和下划线检查框以及颜色组合框

例8.3 字体对话框的使用。

5. “打印”对话框

属性

FromPage：起始页号；

ToPage：终止页号；

Copies：打印份数。

如果打印驱动程序不支持多份打印，该属性有可能始终返回 1。

例8.4　打印对话框的使用。

6. “帮助”对话框

帮助文件需要用其他的工具制作，如Microsoft Windows Help Compiler。

例8.5

属性

HelpCommand：在线Help帮助类型；

HelpFile：Help文件的路径及其名称；

HelpKey：在帮助窗口显示由该帮助关键字指定的帮助信息。

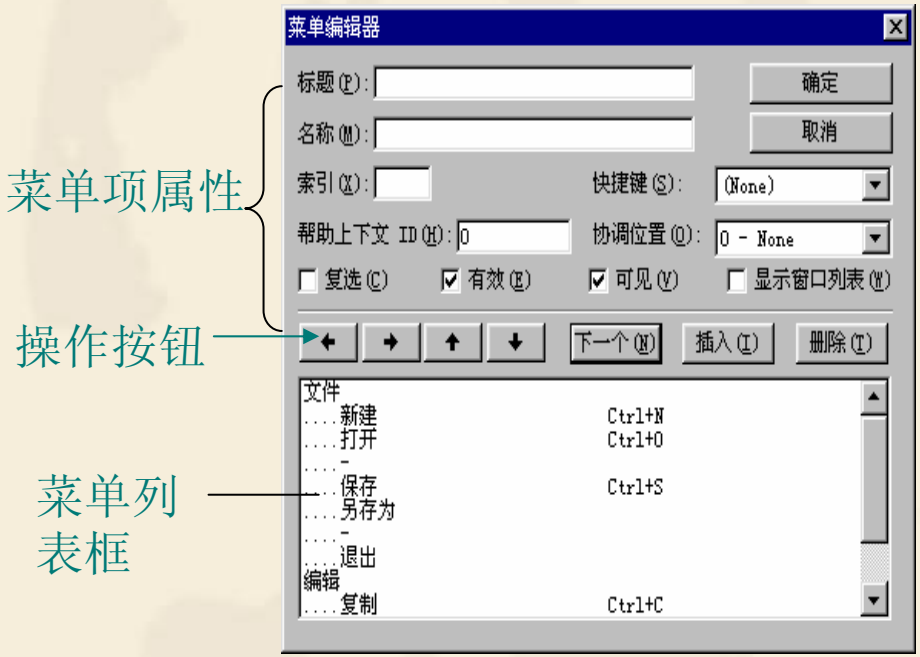
8.2 菜单的设计

下拉式菜单系统的组成结构：

一、菜单编辑器的使用

工具 / 菜单编辑器(Ctrl+E)；

窗体上快显菜单 / 菜单编辑器。



常用属性：

1. 标题(Caption) 热键　前面加&
2. 名称(Name)文本框　分隔符也应有名称
3. 快捷键(Shortcut)　菜单名没有快捷键
4. 复选(Checked)检查框　TRUE 有✓
5. 有效(Enabled)检查框
6. 可见(Visible)检查框

例 8.6 建立一个有菜单功能的文本编辑器。

二、菜单项增减

在程序运行时，菜单随时增减，如“文件”菜单能保留最近打开
的文件数。这同控件数组一样，使用菜单数组。

步骤：

1.在菜单设计时，加入一个菜单项，其Index为0（菜单数组），Visual为False。

2.在程序运行时，通过Load方法向菜单数组增加新的菜单项。

例8.7 在8.6中的文件菜单中保留最近打开的文件清单。

同样，要删除所建立的菜单项，使用UnLoad方法向菜单数组减少菜单项。

三、 弹出菜单(快捷菜单)

弹出菜单是独立于窗体菜单栏而显示在窗体内的浮动菜单。显示位置取决于单击鼠标键时指针的位置。设计与普通菜单相同（如果不希望菜单出现在窗口的顶部，该菜单名Visible属性设置为False）。菜单弹出的方法：

[对象.]PopupMenu 菜单名, 标志, x, y

标志，表示弹出的位置和触发的键

```
Sub Text1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 2 Then    PopupMenu EditMenu, vbPopupMenuCenterAlign
End Sub
```

例8.7 弹出菜单。

8.3 多重窗体和多文档窗体

一、多重窗体

1. 添加窗体

“工程|添加窗体”
- 新建
- 现存

添加“现存”窗体时要注意：

防止多个窗体的Name相同而不能添加；

添加的窗体实际是将其他工程中已有的窗体加入，多个工程共享窗体；

通过“另存为”命令以不同的窗体文件名保存，断开共享。

2. 保存窗体

一个工程中有多个窗体，应分别取不同文件名保存在磁盘上，VBP工程文件中记录了该工程的所有窗体文件名。

3. 设置启动窗体

“工程/属性”——“启动对象”

设置启动对象

4. 窗体语句

(1)Load语句：装入窗体到内存但没有显示窗体

形式： Load 窗体名称

(2)Unload语句：从内存删除窗体

形式： Unload 窗体名称

5. 窗体方法

(1)Show方法：显示一个窗体（当窗体没有Load，自动Load）

[窗体名称].Show [模式]

0 － Modal：关闭才能对其他窗体进行操作。

1 － Modeless，可以对其他窗体进行操作。

(2) Hide方法：隐藏窗体，没有Unload删除

[窗体名称.] Hide

6. 不同窗体间数据的存取

(1)存取控件的属性

另一窗体名.控件名.属性

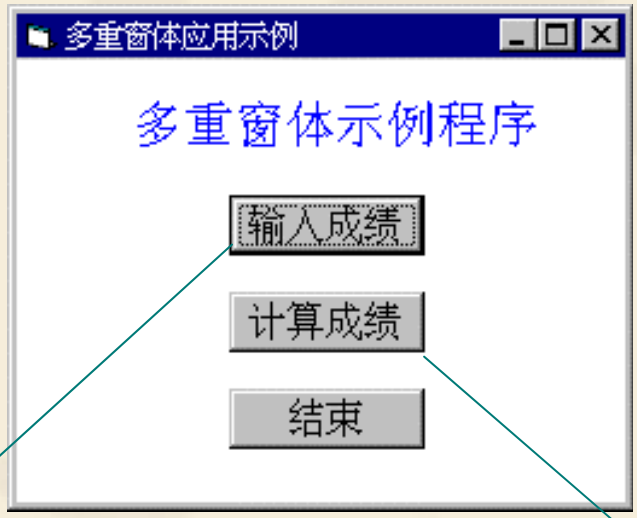
(2)存取变量的值

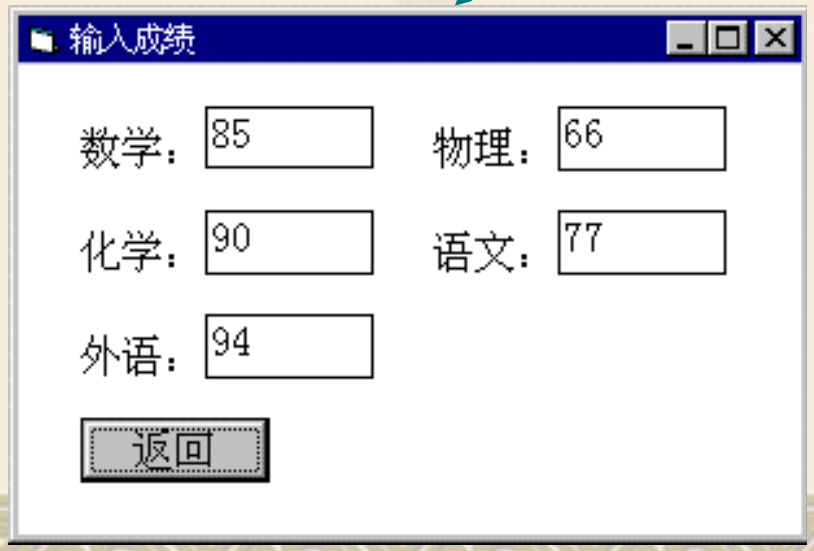
另一窗体名.全局变量名

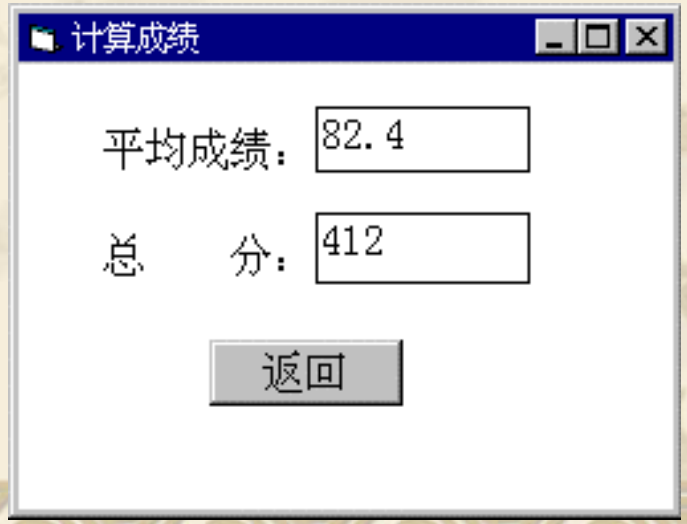
例8.8 输入和计算学习成绩。

模块：

```
Public MATH As Single
Public PHYSICS As Single
Public CHEMISTRY As Single
Public CHINESE As Single
Public ENGLISH As Single
```







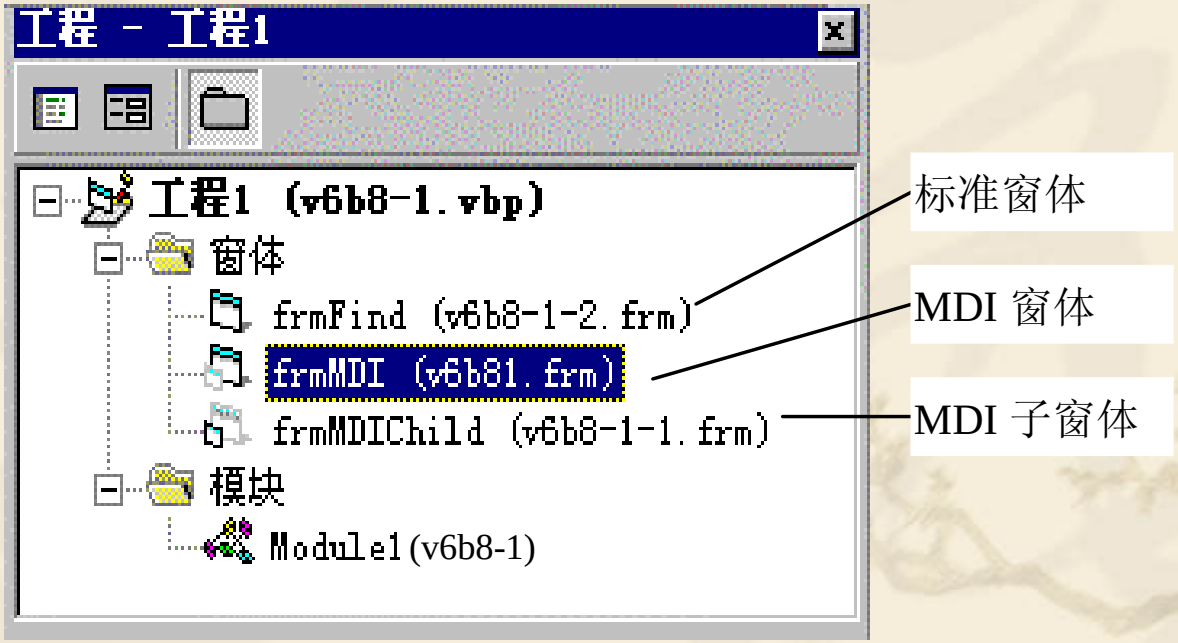
二、多文档界面(MDI)

例8.9

1. 创建和设计MDI窗体及其子窗体

(1) 创建和设计MDI窗体

创建：“工程/添加MDI窗体”命令



设计：一般有菜单栏、工具栏、状态栏

(2) 创建和设计MDI子窗体

创建： MDI子窗体是一个MDIChild属性为True的普通窗体。

要创建多个子窗体，通过窗体类来实现：

```
Public Sub FileNewProc()
    Dim NewDoc As New frmMDIChild
    No = No + 1
    NewDoc.Caption = "no" & No
    NewDoc.Show
End Sub
```

MDI子窗体的Name

设计：可有菜单栏，但必须有文本框。

2. MDI窗体与子窗体的交互

(1)活动子窗体和活动控件

MDI窗体的两个属性：ActiveForm 和ActiveContro。l

例：将子窗体的文本框中所选文本复制到剪贴板上。

```
ClipBoard.SetText  frmMDI.ActiveForm.ActiveControl.SelText
```

(2)显示MDI窗体及其子窗体

显示任何窗体的方法为show,还有有关规则：

- 加载子窗体时，其父窗体会自动加载并显示；反之则无。
- MDI窗体有AutoShowChildren属性，决定是否自动显示子窗体。

(3)维护子窗体的状态信息

(4)用QueryUnload卸载MDI窗体

3. 多文档界面应用程序中的“窗口”菜单

(1)显示打开的多个文档窗口

要在某个菜单上显示所有打开的子窗体标题，只需利用菜单编辑器将该菜单的WindowList属性设置为True。

(2)排列窗口

利用Arrange方法进行层叠、平铺和排列图标。

形式： MDI窗体对象.Arrange 的排列方式：

常数	值	描述
vbCascade	0	层叠所有非最小化
vbTileHorizontal	1	水平平铺所有非最小化
vbTileVertical	2	垂直平铺所有非最小化
vbArrangeIcons	3	重排最小化

8.4 工具栏和状态栏

“Microsoft Windows Common Controls 6.0”将控件添加到工具箱，通过ToolBar、ImageList组合使用建立工具栏，步骤：

（1）在ImageList控件中添加所需的图像。
（2） 在ToolBar控件中创建Button对象。
（3）在ButtonClick事件中用Select Case语句对各按钮进行相应的编程。

一、在ImageList控件中添加图像

ImageList控件不单独使用，专门为其他控件提供图像库。

索引（Index）、

关键字（Key）

在ToolBar中引用

图像文件的扩展名为： .ico、.bmp、.gif、.jpg等。

二、在ToolBar控件中添加按钮

1.为工具栏连接图像

ToolBar与
ImageList的连接

工具栏样式

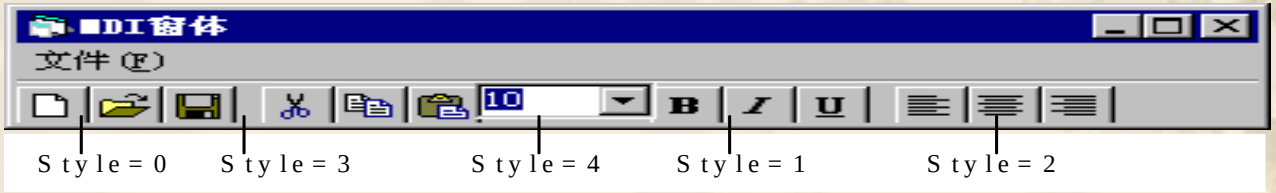
2.为工具栏增加按钮

索引（Index）、关键字（Key）每个按钮的编号、标识，
ButtonClick事件中引用；

样式(Style)，共6种

图像(Image) ImageList对象中的图像，值可以是Key或Index

除菜单按钮(5)外的5种样式



三、响应ToolBar控件事件

ToolBar控件常用的事件有两个：ButtonClick和ButtonMenuClick。前者对按钮样式为0~2，后者对样式为5的菜单按钮。

（1）用索引Index确定按钮

```
Private Sub Toolbar1_ButtonClick(Byval Button As
ComctlLib.Button)
  Select Case Button.Index
    Case 1
      FileNewProc
    Case 2
      FileOpenProc
      ……
    End Select
End Sub
```

(2)用关键字Key确定按钮

```
Select Case Button.Key
    Case  "TNew"
      FileNewProc
   Case   "TOpen"
     FileOpenProc
       . . . . .
```

注意：第1个按钮的**Index**值为1。用**Index**还是用**Key**比较，**Key** 可读性好，可维护性好。

四、状态栏

状态栏显示系统信息和对用户的提示，如：系统日期、软件版本、光标的当前位置、键盘的状态等。一般在窗口的底部。

1. 建立状态栏

8.5 RichTextBox控件

RichTextBox控件可以输入和编辑文本，还可以实现多种文字格式、段落等的设置，还可以插入图形的功能，可真正构成一个像Word一样的字处理软件。

选择“Microsoft Rich TextBox Controls 6.0”将控件添加到工具箱。

1.文件操作方法

（1）LoadFile方法

LoadFile方法能够将RTF文件或文本文件装入控件，其形式如下：

对象.LoadFile 文件标识符[，文件类型]

文件类型： 0或rtfRTF为RTF文件（缺省）；1或rtfTEXT为文本文件

（2）SaveFile方法

SaveFile方法将控件中的文档保存为RTF文件或文本文件，其形式：

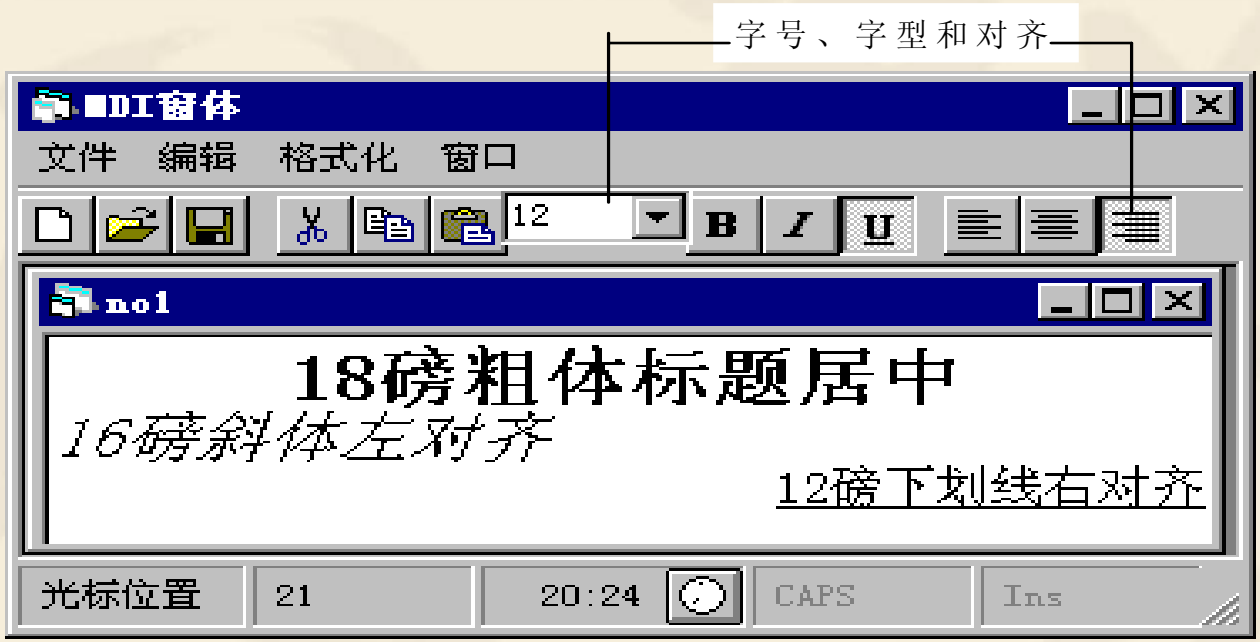
对象. SaveFile（文件标识符[，文件类型]）

2. 常用格式化属性

格式化属性，可对该控件中选中的任何部分的文本使用不同的格式。

分类	属性	值类型	说明
选中文本	SelText　SelStart　SelLength		意义同 Text 控件对应属性
字体、字号	SelFontName　SelFontSize		同上
字型	SelBold　SelItalic SelUnderline　SelStrikethru	逻辑量	粗体 斜体 下划线 删除线
上、下标	SelCharOffset	整型	>0 上标　<0 下标 以 Twip 为单位
颜色	SelColor	整型	
缩排	SelIndent　elRightIndent SelHangingIndent	数值型	缩排单位以 ScalMode 决定
对齐方式	SelAlignment	整型	0 左　1 右　2 中

3. 应用例8.9



4. 插入图像

在RichTextBox控件中可插入 (*.bmp)的图像文件，形式如下：

对象.OLEObjects.Add [索引]，[关键字]，文件标识符

其中：OLEObjects是集合，包含一组添加到RichTextBox控件的对象；

索引和关键字表示添加的元素编号和标识，可省，但逗号不能省。

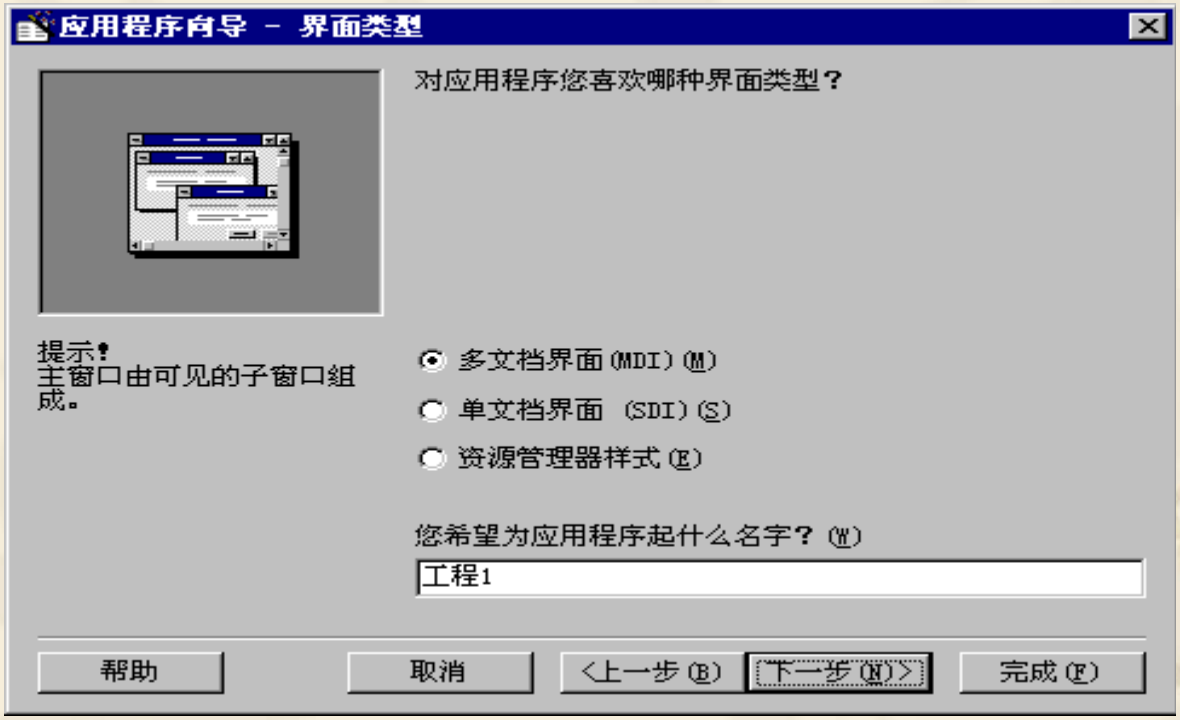
例如: RichTextBox1.OLEObjects.Add , ,"c:\windows\circles.bmp"

8.6 应用程序向导

是方便的程序生成器，用来生成一个应用程序的界面。

选择“文件/新建工程”命令，在其对话框选中“B应用程序向导”。

1. 选择操作界面, 有三种：



2. 选取菜单和菜单项

向导提供了文件、编辑、视图、工具、窗口、帮助六个菜单名，每个菜单名下有若干个菜单项。用户可自由地选取、取消菜单名或菜单项。

3. 选取工具栏按钮

提供的工具栏有13个按钮。用户也可根据需要增加（右移所选按钮）或删除（左移所选按钮）按钮。

第九章　文　件

9.1 驱动器、目录和文件列表框

```
Sub drvDrive_Change()
    dirDirectory.Path = drvDrive.Drive
End Sub
```

```
Sub dirDirectory_Change()
    filFile.Path = dirDirectory.Path
End Sub
```


```
Sub Form_Load()
    Item = “所有文件(*.*)”
     cboType.AddItem Item + Space(20 - Len(Item)) + "*.*"
    Item = "窗体文件(*.FRM)"
    cboType.AddItem Item + Space(20 - Len(Item)) + "*.FRM"
    Item = "位图文件(*.BMP)"
    cboType.AddItem Item + Space(20 - Len(Item)) + "*.BMP"
    cboType.ListIndex = 2
End Sub
Sub cboType_Click()
    filFile.Pattern = Mid(cboType.Text, 21)
End Sub
```

```
Sub filFile_Click( )
    ⋮
End Sub

Sub filFile_DblClick( )
    ⋮
End Sub
```

例9.1示例程序

1. 驱动器列表框

Drive属性：

[对象.]Drive [= drive]

Change事件：

重新设置Drive属性引发Change事件

2. 目录列表框

Path属性 ：

[对象.]Path [= pathname]

Change事件：

重新设置Path 属性引发Change事件

3. 文件列表框

Path属性：

显示该路径下的文件。

重新设置Path属性引发PathChange事件。

Pattern属性：

显示的文件类型。

[对象.]Pattern [= value]

重新设置Pattern属性引发PatternChange事件。

例如：filFile.Pattern = "*.frm"，显示*.frm文件。

多个文件类型用分号；分界。例如： "*.frm;*.frx"

FileName属性：

[对象.]FileName [= pathname]

引用时只返回文件名，相当于ilFile.List(filFile .ListIndex)，需用Path属性得到其路径；设置时可带路径。

Click、DblClick事件：

例如，单击输出文件名。

```
Sub filFile_Click( )
    MsgBox filFile.FileName
End Sub
```

例如，双击执行可执行程序：

```
Sub filFile_DblClick( )
    ChDir (dirDirectory.Path)      ' 改变当前目录

    RetVal = Shell(filFile.FileName, 1)    ' 执行程序

End Sub
```

9.2 文件的读写

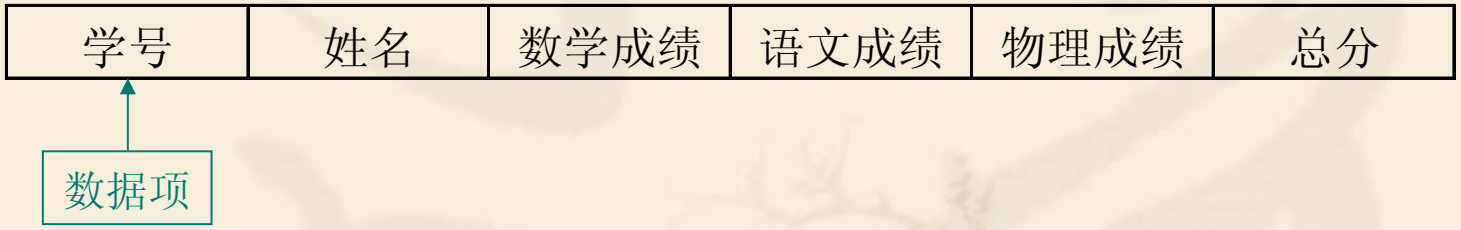
文件：存储在外部介质上数据的集合。

按名存取

1. 记录

由若干个相互关联的数据项组成。

例如，由学生的学习成绩信息组成的记录：



2. 文件及其种类

文件是记录的集合。

顺序访问模式：顺序文件，记录可长可短；

随机访问模式：随机文件，记录的长度相同；

二进制访问模式：二进制文件（可认为记录长度为1）。

(1) 顺序文件

按顺序依次把记录写入文件；

按顺序依次把记录读出来。

记录 1	记录 2	……	记录 N	文件结束标志

文本文件：一行一条记录，记录可长可短，以“换行”字符为分隔符号。

(2) 随机文件

随机文件可以直接访问文件中的任意一个记录。

记录长度相同；

根据记录号访问；

#1 记录 1	#2 记录 2	……	#N 记录 N

(3) 二进制文件

直接把二进制码存放在文件中。

一、 顺序访问模式

1. 打开文件

Open 文件名 [For 模式] As [#]文件号 [Len=记录长度]

(1) 模式

OUTPUT：写操作；

INPUT：读操作；

APPEND：追加到文件末尾。

(2) 文件号

1～511，可以用FreeFile函数获得下一个可利用的文件号。

(3) 记录长度

小于或等于32767的整数，它指定数据缓冲区的大小。

例如，打开C:\VB\SCORE，供写入数据，指定文件号为#1。

OPEN "C:\VB\SCORE" FOR OUTPUT AS #1

2. 写入命令

Print #文件号，[输出列表]

保存文本框

假定文本框的名称为Text1，文件名为TEST.DAT。

方法1：把整个文本框的内容一次性地写入文件。

```
Open "TEST.DAT" For Output As #1
Print #1, Text1
Close #1
```

方法2：把整个文本框的内容一个字符一个字符地写入文件。

```
Open "TEST.DAT" For Output As #1
For i=1 To len(Test1)
        Print #1,Mid(Text1, i, 1);
Next i
Close #1
```

Write #文件号，[输出列表]

紧凑格式。数据项之间插入“,”，并加上双引号。

例如，命令：Write #1,"One","Two",123

内容："One","Two",123

3. 关闭文件

Close [[#]文件号][, [#]文件号]...

例如，Close #1, #2, #3

4. 读顺序文件

INPUT #文件号,变量列表

把读出的每个数据项分别存放到所对应的变量。

LINE INPUT #文件号,字符串变量

读一行到变量中，主要用来读取文本文件。

INPUT$(读取字符数,#文件号)

随意读取字符

5. 函数

LOF(文件号)：返回文件的长度(总字节数)。

EOF(文件号)：返回读写位置。

指针在文件尾时，EOF函数为True，否则为False。

读文本文件到文本框

假定文本框名称为Text1，文件名为MYFILE.TXT。

方法1：一行一行读

```
Text1.Text ＝ ""
Open "MYFILE.TXT" For Input As #1
Do While Not EOF(1)
        Line Input #1, InputData
        Text1.Text ＝ Text1.Text + InputData+vbCrLf
Loop
Close #1
```

方法2：一次性读

```
Text1.Text ＝ ""
Open "MYFILE.TXT" For Input As #1
Text1.Text = Input$( LOF(1), 1)
Close #1
```

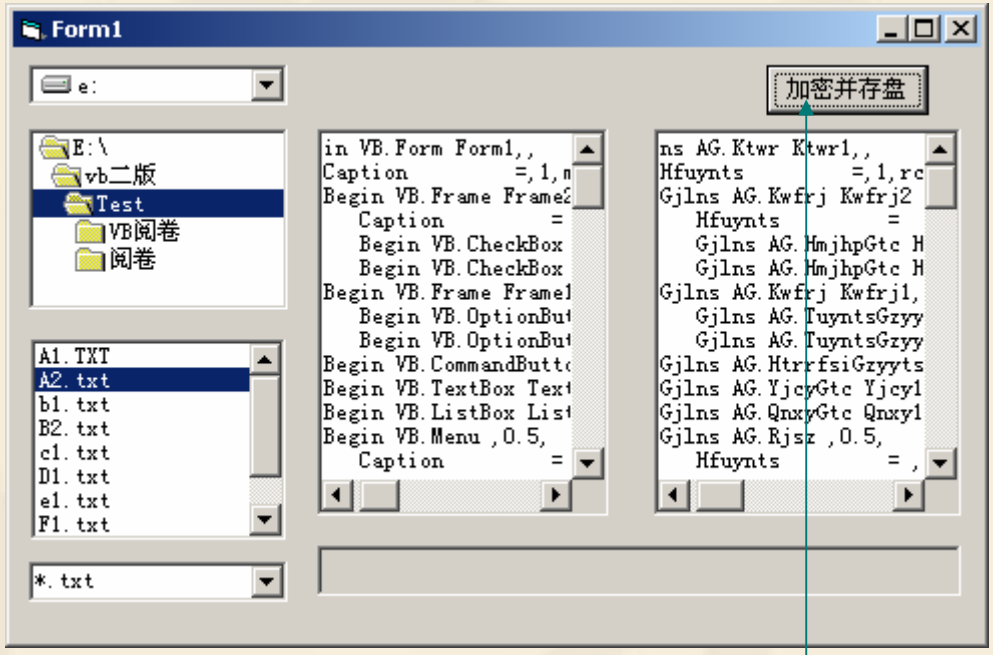
方法3：一个个字符读

```
Dim InputData as String*1
Text1.Text ＝ ""
Open "MYFILE.TXT" For Input As #1
Do While Not EOF(1)
        InputData= Input$(1,#1)
        Text1.Text ＝ Text1.Text + InputData
Loop
Close #1
```

例9.2读入文本文件。

例9.3 文件加密程序。

```
Private Sub File1_DblClick()
   Open  tfilename For Input As #1
   Text1.Text = ""
   Do While Not EOF(1)
        indata = Input(1, #1)
       Text1.Text = Text1.Text + indata
   Loop
End Sub
```




```
CommonDialog1.Action = 2
Open CommonDialog1.FileName For
Output As #1
For i = 1 To Len(Text1.Text)
        Print #1, Mid(Text1.Text, i, 1);
Next i
Close #1
```

参阅例6.14

二、随机文件

1. 打开

Open 文件名 **For Random** As #文件号 **[Len=记录长度]**

2. 写操作

Put [#]文件号，[记录号]，变量名

将一个记录变量的内容写到指定的记录位置处。

忽略记录号，则表示在当前记录后的位置插入一条记录。

3. 读操作

Get [#]文件号，[记录号]，变量名

忽略记录号，则读出当前记录后的那一条记录。

例9.4 学生信息管理程序。

追加记录(Command1)：将一个学生的信息作为一条记录添加到随机文件末尾。

显示记录(cmdDisplay)：显示在右边文本框(text4)中指定的记录。

三、二进制文件

1. 打开

Open 文件名 **For Binary** As #文件号

2. 写操作

Put [#]文件号，[位置]，变量名

写入长度等于变量长度的数据。

3. 读操作

GET [#]文件号，[位置]，变量名

从指定位置开始读出长度等于变量长度的数据存入变量中，数据读出后移动变量长度位置，如果忽略位置，则表示从文件指针所指的位置开始读出数据，数据读出后移动变量长度位置。

例9.5 文件复制

```
Dim char As Byte
Dim FileNum1,FileNum2 as Integer
FileNum1 = FreeFile
'打开源文件
Open "C:\STUDENT.DAT" For Binary As # FileNum1
FileNum2 = FreeFile
' 打开目标文件
Open "C:\STUDENT.BAK" For Binary As # FileNum2
Do While Not EOF(FileNum1)
    Get #1, , char          ' 从源文件读出一个字节
    Put #2, , char                  ' 将一个字节写入目标文
      件
Loop
Close #FileNum1
Close #FileNum2
```

9.3 常用的文件操作语句和函数

1. FileCopy语句

格式：FileCopy source , destination

功能：复制一个文件。

说明： FileCopy语句不能复制一个已打开的文件。

2. Kill语句

格式：Kill pathname

功能：删除文件。

说明：pathname中可以使用统配符“*”和“?”。

例如：Kill "*.TXT"

3. Name 语句

格式：Name oldpathname As newpathname

功能：重新命名一个文件或目录。

说明：(1) Name具有移动文件的功能。

(2) 不能使用统配符“*”和“?”，不能对一个已打开的文件上使用

Name语句。

4. ChDrive 语句

格式：ChDrive drive

功能：改变当前驱动器。

说明：如果drive为“ ”，则当前驱动器将不会改变；如果drive中有多个字 符，则ChDrive只会使用首字母。

5. MkDir 语句

格式：MkDir path

功能：创建一个新的目录。

6. ChDir 语句

格式：ChDir path

功能：改变当前目录。

例如：ChDir "D:\TMP"

7. RmDir 语句

格式：RmDir path

功能：删除一个存在的目录。

说明：只能删除空目录。

8. CurDir函数

格式：CurDir[(drive)]

功能：利用CurDir函数可以确定任何一个驱动器的当前目录。

说明： drive为" "，则CurDir返回当前驱动器的当前目录。

例9.6 利用ChDrive和ChDir语句改写例9.1中的File1_DblClick()事件过程。

```
Sub File1_DblClick()
        ChDrive Drive1.Drive        ' 设置缺省驱动器
        ChDir File1.Path            ' 设置缺省目录
        RetVal = Shell(File1.FileName, 1)
End Sub
```

例9.7 为例9.1添加事件过程File1_MouseDown()，使之支持Del键，即按下Del键删除选定的文件。

```
Sub File1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = vbKeyDelete Then
        ChDrive Drive1.Drive
        ChDir File1.Path
        Kill File1.FileName
        File1.Refresh      ' 文件删除后更新文件列表框
    End If
End Sub
```

第十章 图形操作

（3学时）

10.1 图形操作基础

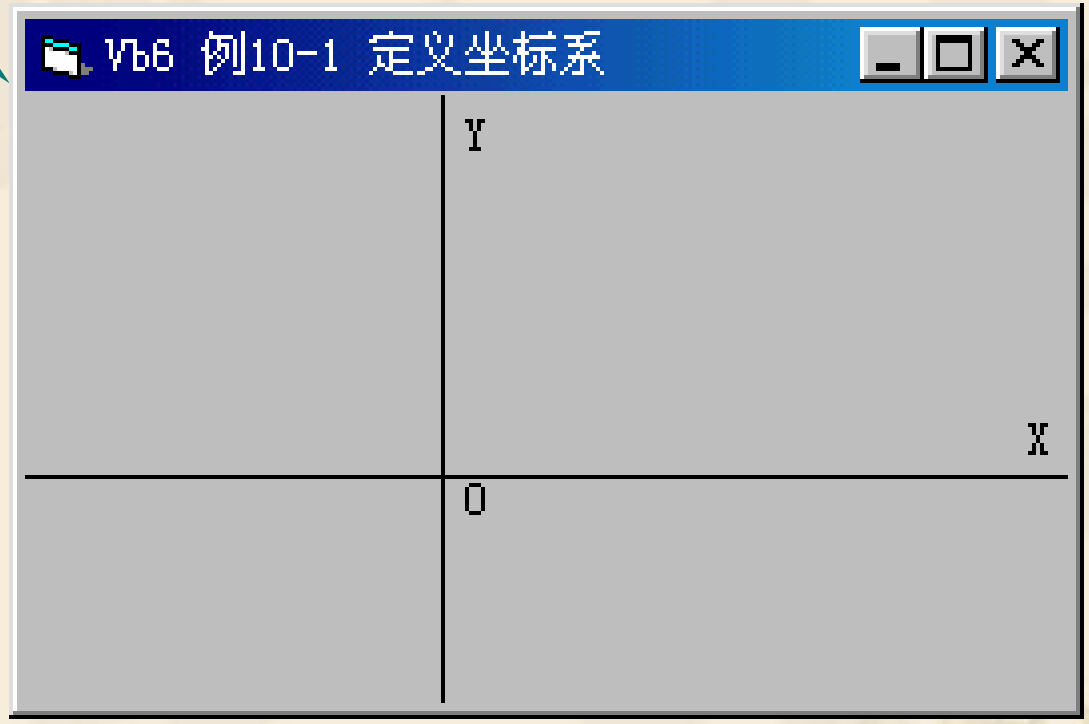
10.1.1 坐标系统

每个容器都有一个坐标系，构成一个坐标系，需要三个要素：坐标原点、坐标度量单位、坐标轴的长度与方向。坐标度量单位由容器对象的ScaleMode属性决定。缺省时为Twip。每英寸1440个Twip，20个Twip为一磅。

10.1.2 自行定义坐标系

方法一：通过ScaleTop，ScaleLeft，ScaleWidth和ScaleHeight属性实现。

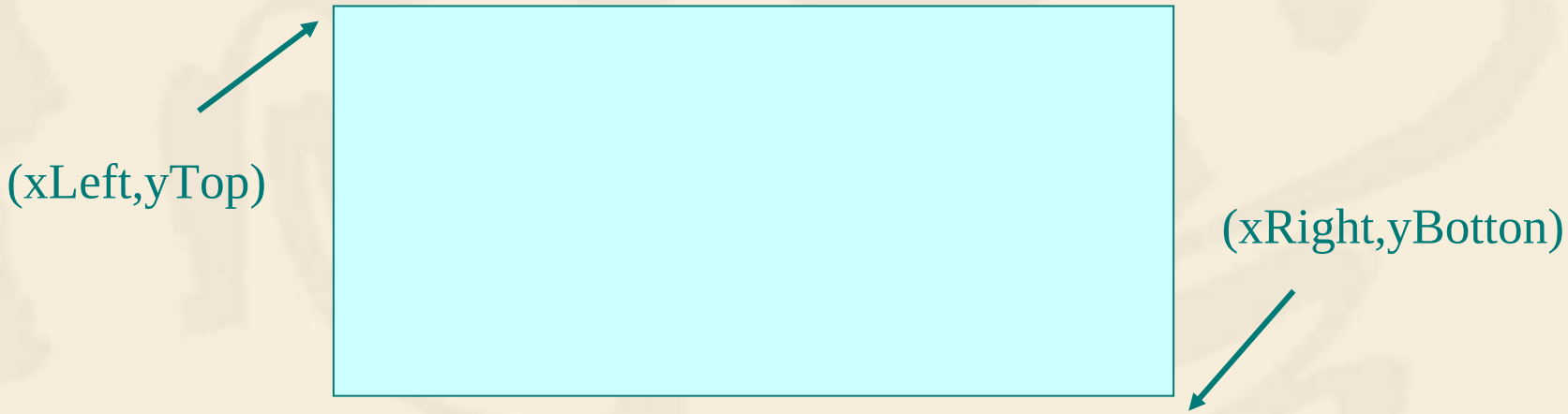
（ScaleTop，ScaleLeft）



例10.1 在Form_Paint 事件中定义窗体的坐标系。

方法二：采用Scale方法来设置坐标系：

[对象.]Scale [(xLeft,yTop)-(xRight,yBotton)]



例如，Form1.Scale (-200,250) - (300, -150)将建立和例10. 1一样的坐标系。

可在程序中使用Scale方法改变坐标系统。当Scale方法不带参数时，取消用户定义的坐标系，采用缺省坐标系。

改变坐标系后产生的影响：

例10. 1a　Line (0, 0) - (1000, 1000)在不同坐标系的效果。

例10. 1b　控件对象在坐标系内与X轴和Y轴的位置保持相对不变。

10.1.3 图形层

三个图形层放置的对象

层 次	对象类型
最上层	工具箱中除标签、线条、形状外的空件对象
中间层	工具箱中标签、线条、形状空件对象
最下层	由图形方法所绘制的图形

利用图形层的特点，实现悬浮效果。

在命令按钮后放置一个表面色彩为黑色的标签，如图所示。

同一图形层内控件对象排列顺序称为Z序列。
Zorder方法的语法为： 对象.Zorder [position]
position =0 表示该控件被定位于Z序列的前面；
position = 1表示该控件被定位于Z序列的后面。

例 Zorder方法的使用。

10.2 绘图属性

10.2.1 当前坐标

CurrentX，CurrentY属性给出窗体或图形框或打印机在绘图时的当前坐标。

这两个属性在设计阶段不能使用。

例10.2 利用CurrentX，CurrentY属性在窗体上随机打印100个“★”。

10.2.2 线宽与线型

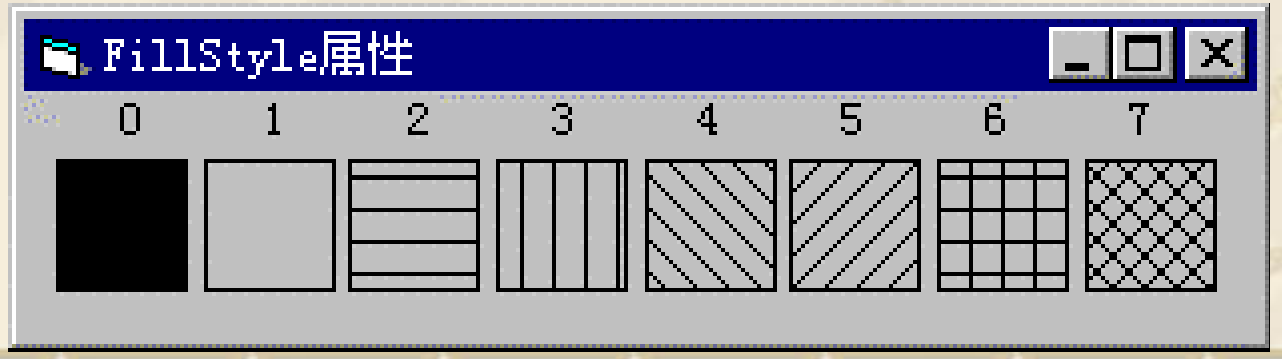
窗体、图形框或打印机的DrawWidth属性给出这些对象上所画线的宽度或点的大小。DrawWidth属性以像素为单位来度量，最小值为1。

例10.3 用DrawWidth属性改变直线宽度。

窗体或图形框或打印机的DrawStyle属性给出这些对象上所画线的形状。

设置值	线　型	图　示
0	实线(缺省)	
1	长划线	
2	点线	
3	点划线	
4	点点划线	
5	透明线	
6	内实线	

10.2.3　填充与色彩



Visual Basic 默认采用对象的前景色(ForeColor属性)绘图，也可以通过以下颜色函数指定色彩。

1. RGB(红,绿,蓝)函数　　红、绿、蓝三基色使用0~255之间的整数。

2. QBColor(颜色码)　　颜色码使用0~15之间的整数。

例10. 4　颜色的渐变过程。

10.3 图 形 控 件

10.3.1 Picture Box(图形框)

主要作用：显示图片、也可作为其他控件的容器。

装入图形：图形框对象.Picture = LoadPicture(“图形文件名”)

删除图形：图形框对象.Picture = LoadPicture( )

Autosize属性设置为True时，图形框能自动调整大小与显示的图片匹配。

10.3.2 Image(图像框)

图像框比图形框占用更少的内存块。图像框内不能保存其他控件。

Stretch属性=False，图像框可自动改变大小，以适应其中的图形。

Stretch属性=True， 图形可自动调整尺寸，以适应图像框的大小。

例10.5 图形框Autosize属性与图像框的Stretch属性对加载图形的影

SavePicture对象名.属性,文件名

把绘制或装在窗体、图形框和图像框内的图形保存到Bmp格式文件内。

10.3.3 Line(画线工具)

10.3.4 Shape(形状)

例10.6

例10.7

10.4.1 Line方法10.4 图 形 方 法

画直线或矩形： [对象.] Line [[Step] (x1,y1)]-(x2,y2)[,颜色][,B[F]]

其中：对象可以是窗体或图形框。

(x1,y1)，(x2,y2)为线段的起终点坐标或矩形的左上角右下坐标。

关键字B表示画矩形，关键字F表示用画矩形的颜色来填充矩形。

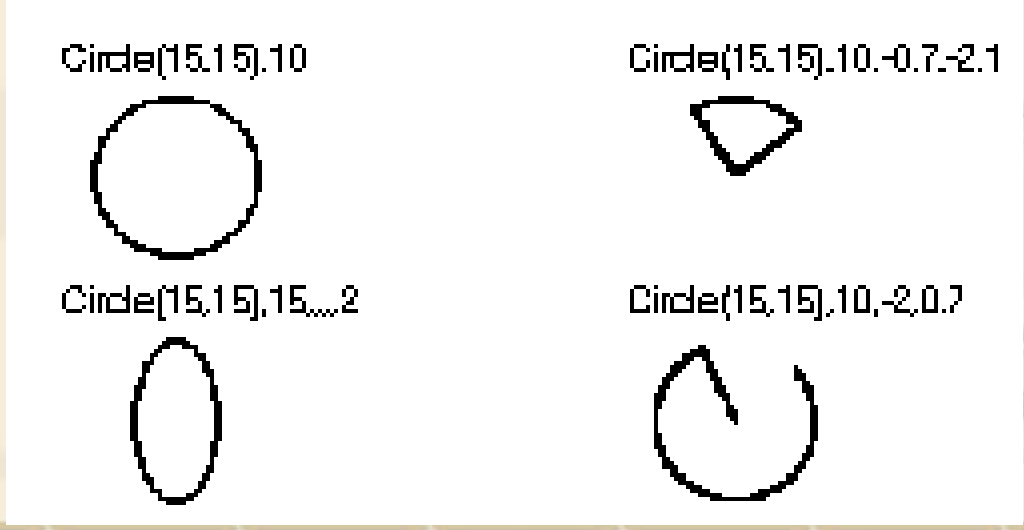
例10.8 用Line方法在一个窗体上画坐标轴与坐标刻度。

例10.9 用Line方法在一个窗体上画随机射线。

10.4.2 Circle方法

Circle方法

用于画圆、椭圆、圆弧和扇形。



10.4.3 Pset方法

Pset方法用于画点： [对象.] Pset [Step] (x,y) [,颜色]

参数(x,y)为所画点的坐标,关键字Step表示采用当前作图位置的相对值。

例10. 10 本例用Pset方法绘制阿基米德螺线

10.4.4 Point方法

Point方法用于返回指定点的RGB颜色，其语法格式如下：

[对象.] Point (x,y)

其中参数对象与(x,y)的意义与前述相同。

如果(x, y)点位于对象之外，Point 方法将返回True。

例10. 11 用Point方法获取一个区域的信息。

利用例10 .11的处理方法可使图片产生朦胧的效果。

10.5 应　用

10.5.1 几何图形绘制

利用Line方法和Circle方法及DrawWidth、DrawStyle和DrawMode属性。

例10.12 用Circle方法在窗体上绘制由圆环构成的艺术图案。

算法：等分半径为*r*的圆周为*n*份，以等分点为圆心，半径*r*1绘制*n*个圆。

例10.13 用Line方法绘制函数f(*x*)=x^2在区间[*a*, *b*]之间积分面积图。

为了能绘制任意区间[*a*,*b*]上函数*f*(*x*) 积分面积图，可在窗体上放置一个图形框和两个文本框。文本框用于指定积分上下限的值，图形框用于绘图。根据区间[*a*,*b*]的值设置图形框的左上角坐标为（*a*−1，*b*×*b*+1），右下角坐标为（*b*+1，−1）。将区间[*a*,*b*]等分为*n* 份，在每一等分点i上，用Line方法连线到（i,i*i）。

10.5.2 简单动画设计

动画：有计划地移动一个对象包括改变对象的形状和尺寸。
编程时可以采用帧动画原理，即通过一系列静态图辅之以连续快速变化产生动画效果，也可以通过Move方法改变图形对象的Top及Left属性来移动图形。动画的速度使用时钟控制。

例10.14 通过改变图形形状演示一个陀螺在图形框内转动。

例10.15 演示地球图标的转动，在转动时同时使地球图标在窗体内移动，并变化地球图标的尺寸。(用PictureClip控件来存放一组图片)

10.5.3 图形漫游

滚动图形通过滚动条的滑块移动，只要设置图形框的Left或Top为滑块当前值的负数，就可形成图形相对移动。

例10.16 本例在窗体内显示一幅地图，移动滚动条漫游地图。

10.5.4 图形处理技术

1. 操作象素

通过PaintPicture方法访问内置在操作系统中的BitBlt服务程序。

dpic.PaintPicture spic,dx,dy,dw,dh,sx,sy,sw,sh,rop

其中：

参数dpic为目标图形对象，spic为传送源，

dx,dy是传送目标矩形区域左上角坐标（目标控件内任一位置）。

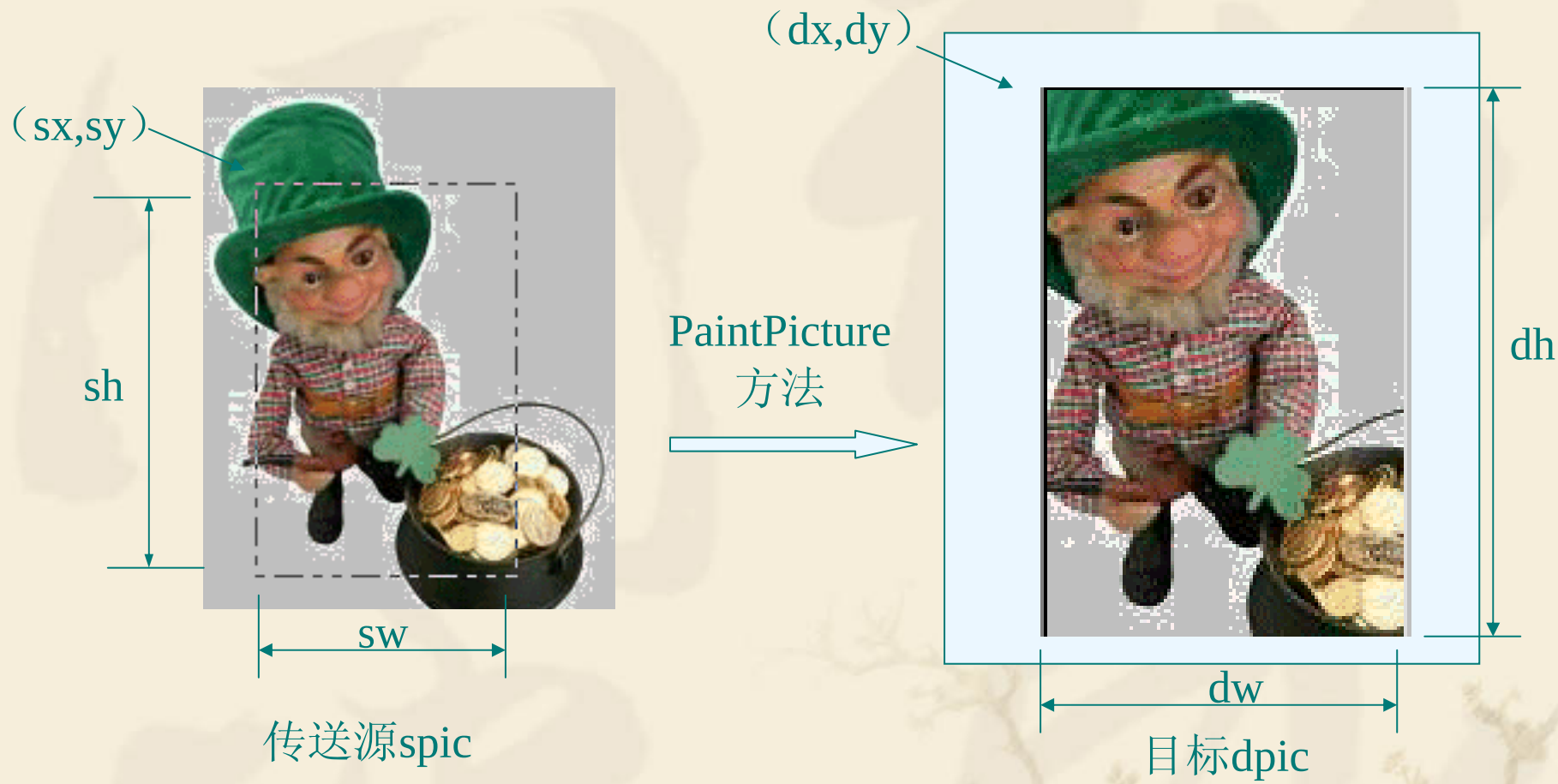
dw,dh是目标形矩形区域的宽和高。

sx,sy是要传送图形矩形区域左上角坐标。

sw,sh是要传送图形矩形区域的大小。

rop指定传送的象素与目标中现有的象素组合模式（如表所示）。

常　量	数　值	说　明
vbDstInvert	&H00055009	逆转目标位图
vbNotSrcCopy	&H00330008	复制源位图的逆到目标位图
vbSrcCopy	&H00cc0020	复制源位图的到目标位图
vbSrcInvert	&H00660046	用 XOR 组合源位图与目标位图



dpic.PaintPicture spic,dx,dy,dw,dh,sx,sy,sw,sh,rop

dw,dh sw,sh至少8个Twip

复制图形：设置目标区域左上角坐标和大小与源矩形区域相同。

翻转图形：只需改变坐标系，设置图形宽为负数，则水平翻转图形；图形高度为负数，则上下翻转图形；如果宽度和高度都为负数，则两个方向翻转图形。

放大图形：改变目标图形的宽度和高度。

旋转图形：要需要对原始图片按行和列的顺序或按列和行的顺序扫描象素点，然后在目标图形区颠倒行和列的顺序复制象素点。

例10.17　PaintPicture方法翻转放大位图。

例10.17a　本例使用PaintPicture方法实现百叶窗效果。

2. 使用DrawMode属性绘制图形

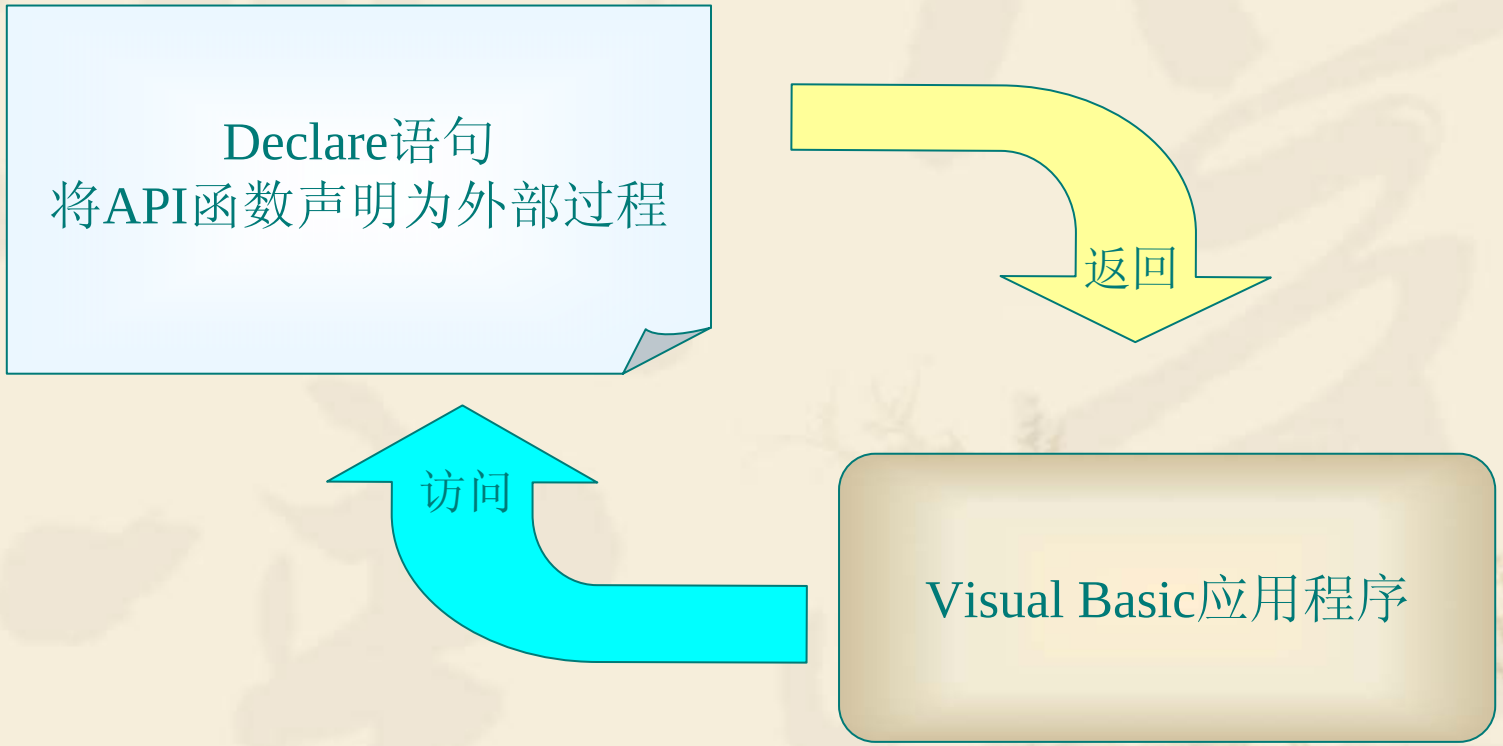
DrawMode属性可以将新象素与原有象素用不同的方法组合。两次Xor运算恢复原有象素。

例10.18　用鼠标选定图形中的区域复制到指定的图片框。

例10.18a　在Xor运算符下图形叠加的效果。

10.5.5 改变窗体对象外观

通过调用Windows的API函数可改变窗体的外观形状。VB应用程序要访问API函数，必须在VB应用程序中用Declare语句将API函数声明为外部过程。



将API函数声明为外部过程，操作流程如下：

通过“工程/添加模块”在当前工程内加入一个BAS模块文件。

执行VB程序组中的API文本浏览器，启动API函数查看工具。

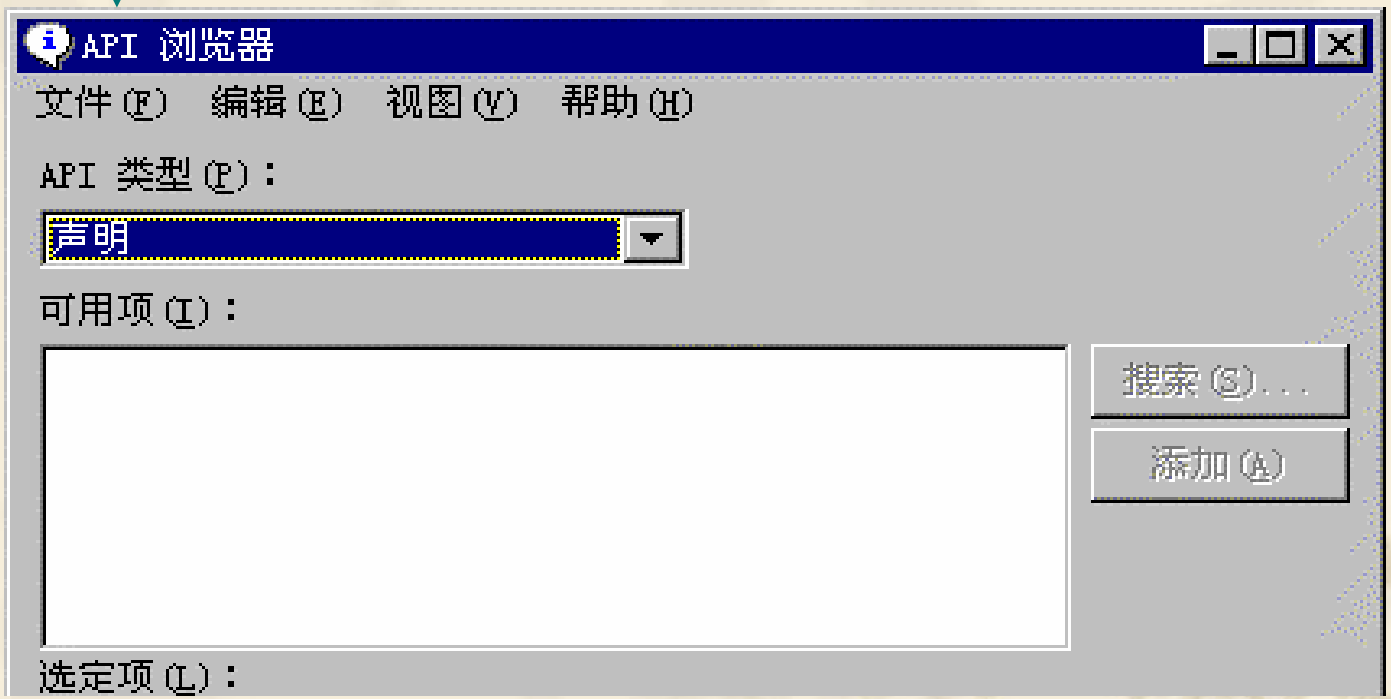
单击“文件|加载文本文件”命令，装入Win32api.Txt文件。

添加指定的API函数到选定项框中，并将函数声明复制到剪贴板。

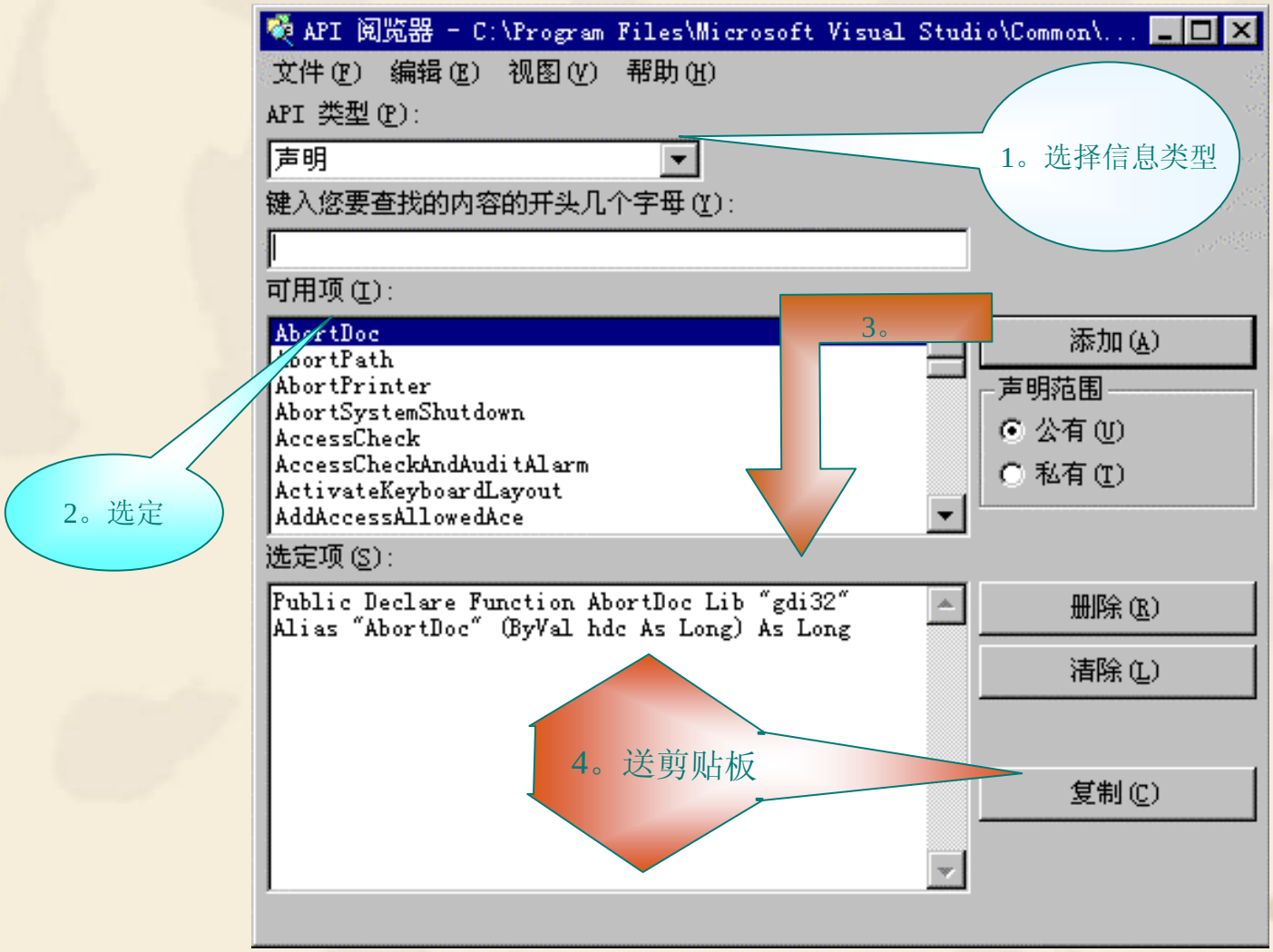
将剪贴板中的内容粘贴到BAS模块文件中。

API文本浏览器应用程序

文件/加载文本文件



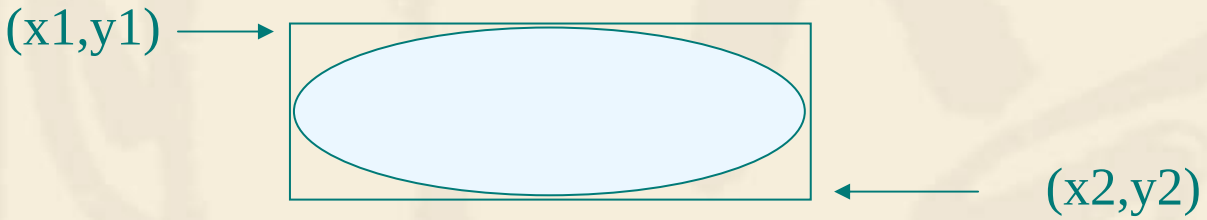
选定API函数复制到剪贴板



例10.19 建立一个椭圆形状的窗体。

API函数CreateEllipticRgn建立椭圆外形区域，SetWindowRgn显示该区域。

CreateEllipticRgn函数参数说明：



坐标参数采用象数值。实际显示的窗体区域由窗体的Height、Width属性确定。需要使用ScaleX和ScaleY方法对窗体的度量单位进行转换。

SetWindowRgn函数参数说明：

hWnd窗口句柄，hRgn为窗口形状，bRedraw图形重绘控制，为一逻辑值。

句柄(Handle)可看作一个对象的指针，通过它可访问该对象。可通过API函数或对象的属性返回句柄。

使用API的区域设置函数不仅可以改变窗体外观，也可以改变控件外观。

10.5.6 在程序中加入后台音乐

API函数sndPlaySound可以直接播放音频文件或系统声音。sndPlaySound函数有两个参数，lpszSoundName指定播放的音频文件或系统声音，uFlags设定播放状态。

uFlags设置

uFlags	设 置	说 明
SND_SYNC	&H00	同步播放
SND_ASYNC	&H01	非同步播放
SND_NODEFAULT	&H02	找不到指定的语音文件时也不播放预设的声音
SND_LOOP	&H08	重复播放
SND_NOSTOP	&H10	不要停止其他正在播放的语音

例10.20 本例示范用sndPlaySound函数播放音频文件。

调用形式：k = sndPlaySound(音频文件名,播放方式)

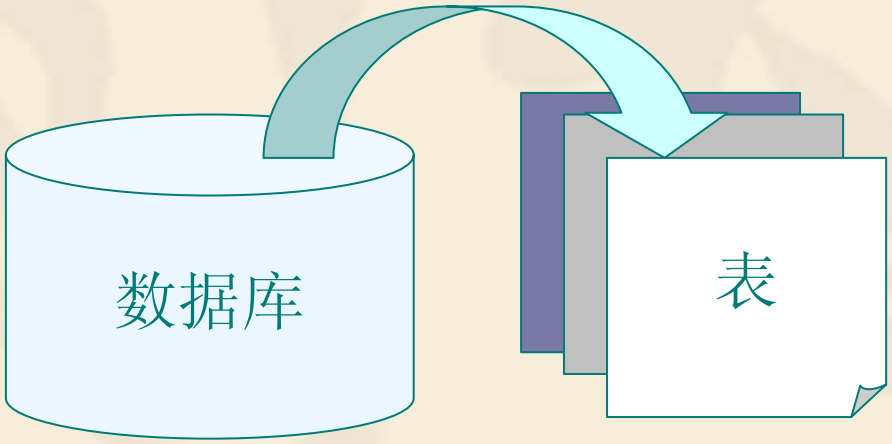
播放文件为空，就可停止音乐播放

注：本例要求有多媒体设备

第十一章 Visual Basic与数据库

（1学时）

11.1 数据库概念



数据库就是一组排列成易于处理和读取的相关信息的集合。

关系模型已经成为数据库设计事实上的标准。

关系型数据库模型

学号	姓名	性别	专业	出生年月
990001	万林	男	物理	82-1-21
990002	庄前	女	物理	82-9-21
990101	丁保华	男	数学	82-4-4
990102	姜沛棋	女	数学	81-12-2
990103	朱克良	男	数学	82-10-1
990201	程玲	女	计算机	82-11-14
990202	黎敏艳	女	计算机	83-2-21
991103	章万京	男	电气	82-6-3
991104	陈友良	男	电气	83-5-5

一个数据库可以由多个表组成，表与表之间可以用不同的方式相互关联。若第一个表中的一条记录内容与第二个表中多条记录的数据相符，但第二个表中的一条记录只能与第一个表的一条记录的数据相符，这样的表间关系类型叫做一对多关系。

基本情况表

学号	姓名
990001	万林
990002	庄前
…	…

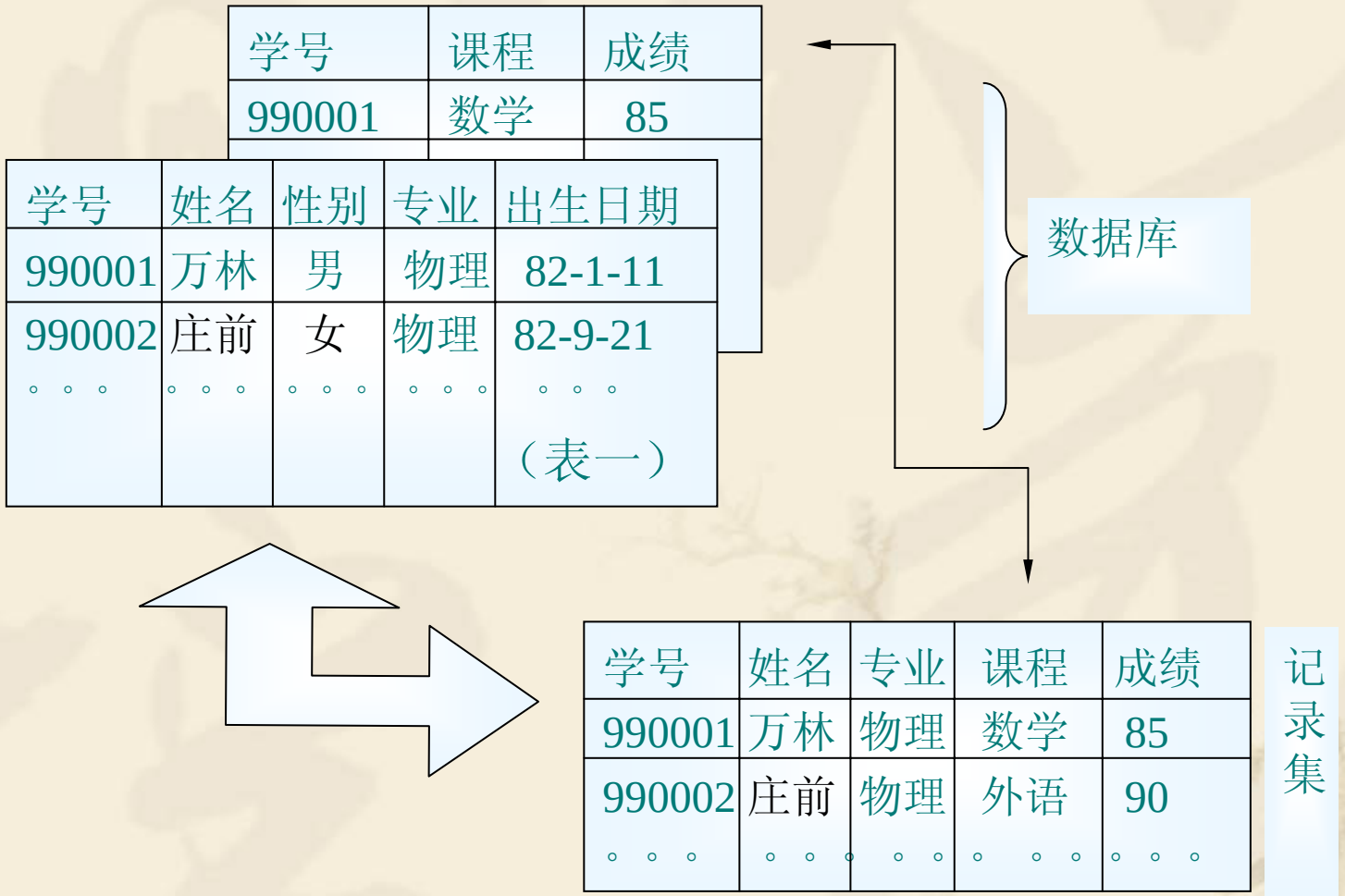
学生成绩表

学号	课程	成绩
990001	数学	85
990001	外语	90
…	…	…

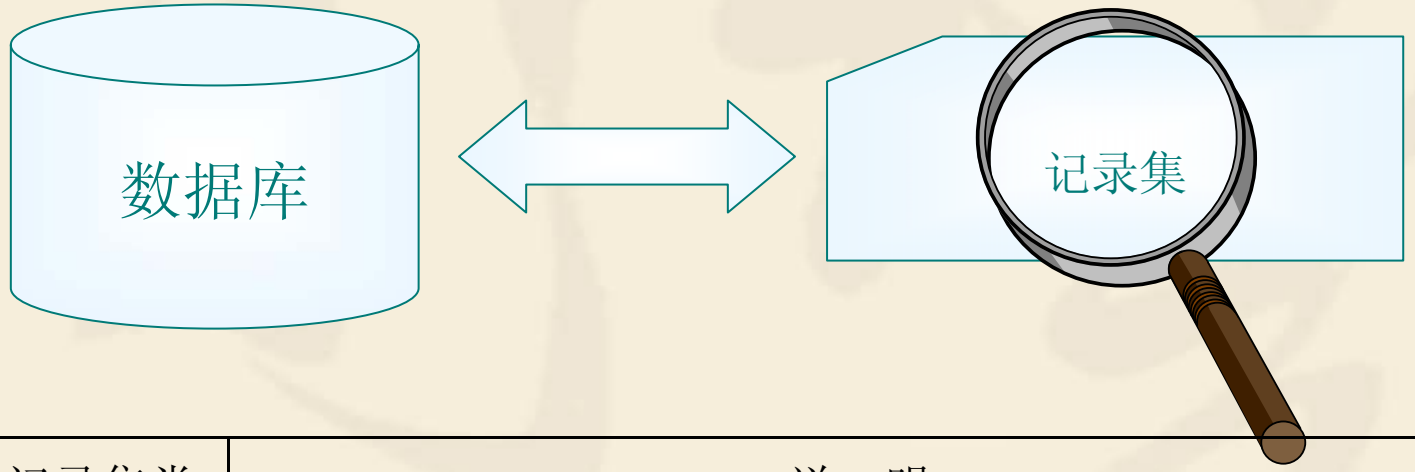
一对多关系

若第一个表的一条记录的数据内容可与第二个表的多条记录的数据相符，反之亦然，这样的表间关系类型叫做多对多关系。

可以将一个或几个表中的数据构成记录集Recordset对象，记录集也由行和列构成，它与表类似。



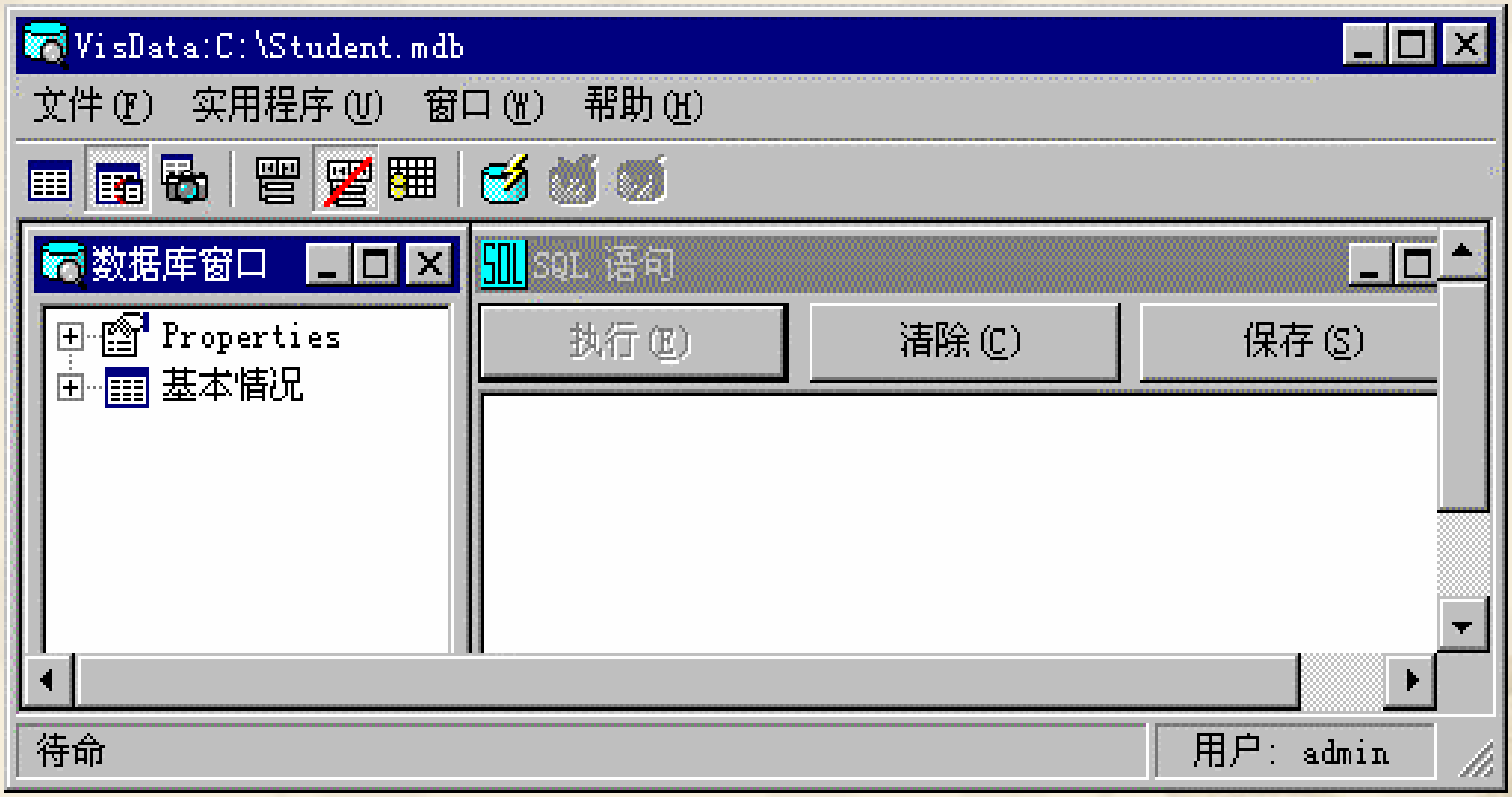
在VB中数据库内的表格不允许直接访问，而只能通过记录集对象进行记录的操作和浏览，因此，记录集是一种浏览数据库的工具。



记录集类	说　明
Table	是表格直接显示的数据。它比其他类型记录集处理速度快，内存开销较大。
Dynaset	一个或者几个表中的记录的引用，动态集和产生动态集的基本表可以互相更新。是最灵活的，功能最强的记录集。
SnapShot	数据库一瞬间的状态，显示的数据是静态、只读状态，内存开销最少。

11.2 数据库管理器

VB的数据库管理器(Visdata.exe)可用于管理数据库。在VB开发环境内单击外接程序菜单中的可视化数据管理器命令可打开可视数据管理器。



建立Student.mdb 数据库，所含学生基本情况表结构如下：

字段名	类 型	宽 度	字段名	类 型	宽 度
学号	Text	6	专业	Text	10
姓名	Text	10	出生年月	Date	8
性别	Text	2	照片	Binary	

数据库管理器使用小结：

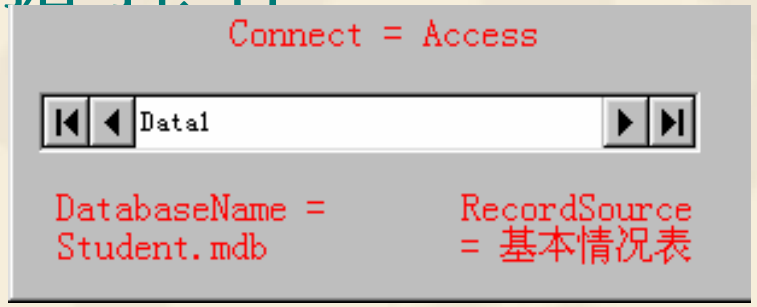
1. 建立新表：
   鼠标右键单击数据库窗口，弹出菜单，选择对应命令。
2. 打开、删除表，修改表结构和建立表间的关联等操作：
   右键单击数据库窗口内的表名，弹出菜单，选择对应命令。
3. 编辑记录：
   双击表名，打开表格输入窗，编辑、增删记录。

11.3 数据控件

11 .3.1 数据控件

工具箱内数据控件图标形状

画在窗体上的外观

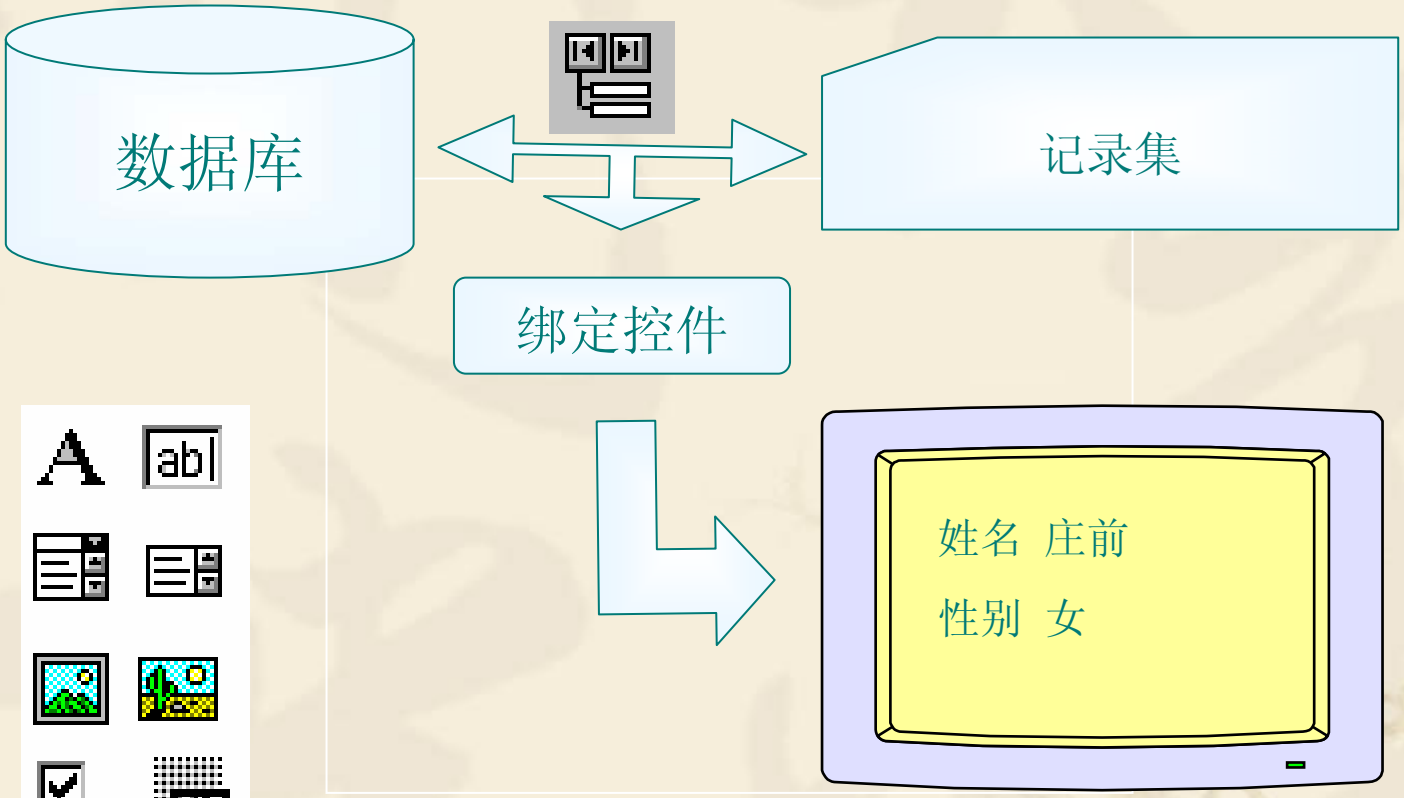


能够利用三种记录集对象访问数据库中的数据，连接方法：

连接属性	Data控件属性说明
Connect	指定数据控件所要连接的数据库类型。
DatabaseName	指定具体使用的数据库文件名，包括所有的路径名。
RecordSource	确定具体可访问的数据，这些数据构成记录集对象。
RecordType	确定记录集类型。

注：RecordSource属性可以是数据库中的单个表名，也可以是使用SQL查询语言的一个查询字符串。如果连接的是单表数据库，则DatabaseName属性应设置为数据库文件所在的子目录名，而具体文件名放在RecordSource属性中。

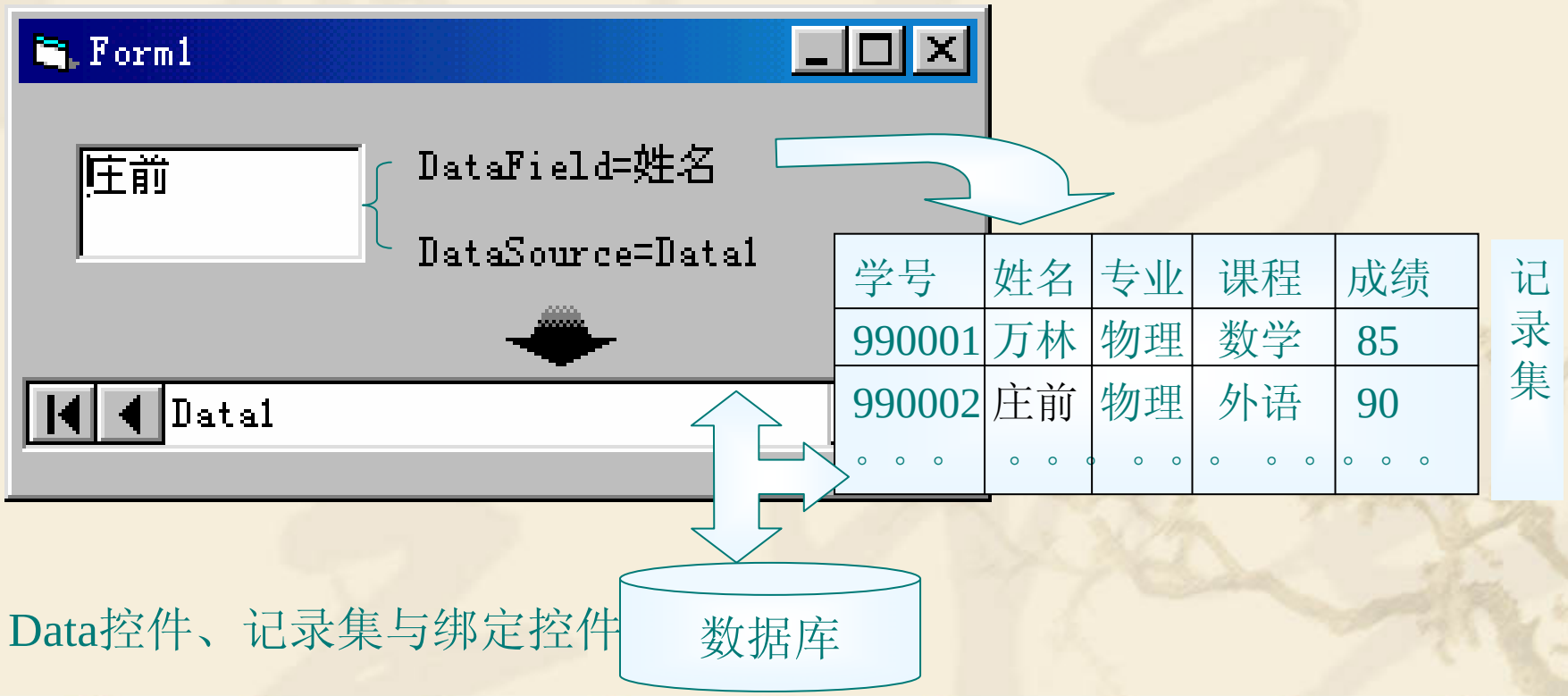
数据控件只能连接数据库产生记录集，不能显示记录集中的数据，要显示记录集中的数据必须通过能与它绑定的控件来实现。



常用绑定控件

绑定控件具有DataSource 和DataField两个重要属性，其作用如下：

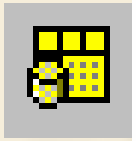
属　性	绑定控件绑定属性说明
DataSource	指定一个有效的数据控件连接到数据库上。
DataField	设置数据库有效的字段与绑定控件建立联系。



Data控件、记录集与绑定控件

例11.1 设计一个窗体显示在11.2节中建立的的Student.mdb数据库中基本情况表的内容。

例11.2 用一个数据网格控件MsFlexGrid显示Student.mdb数据库中基本情况表的内容。



默认控件名	其他属性设置
Data1	DatabaseName =“目录名 \Student.mdb“
	RecordsetTyp = 0
	RecordSource =“基本情况“
MsFlexGrid 1	Datasource = Data1

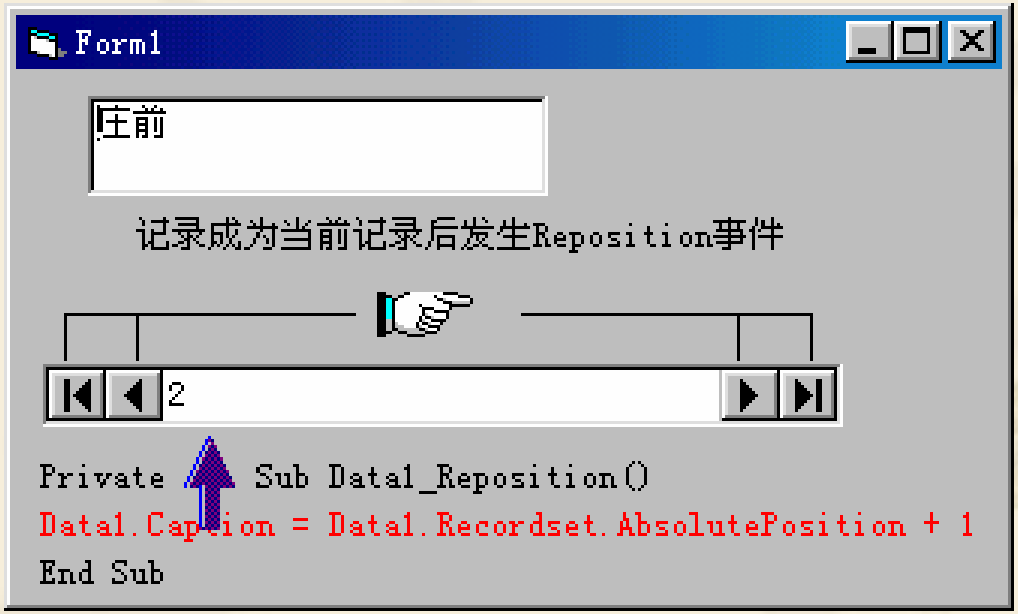
属性：

Rows 、Cols （网格的行或列数）；

FixedRows 、FixedCols （不可卷动的行或列数）。

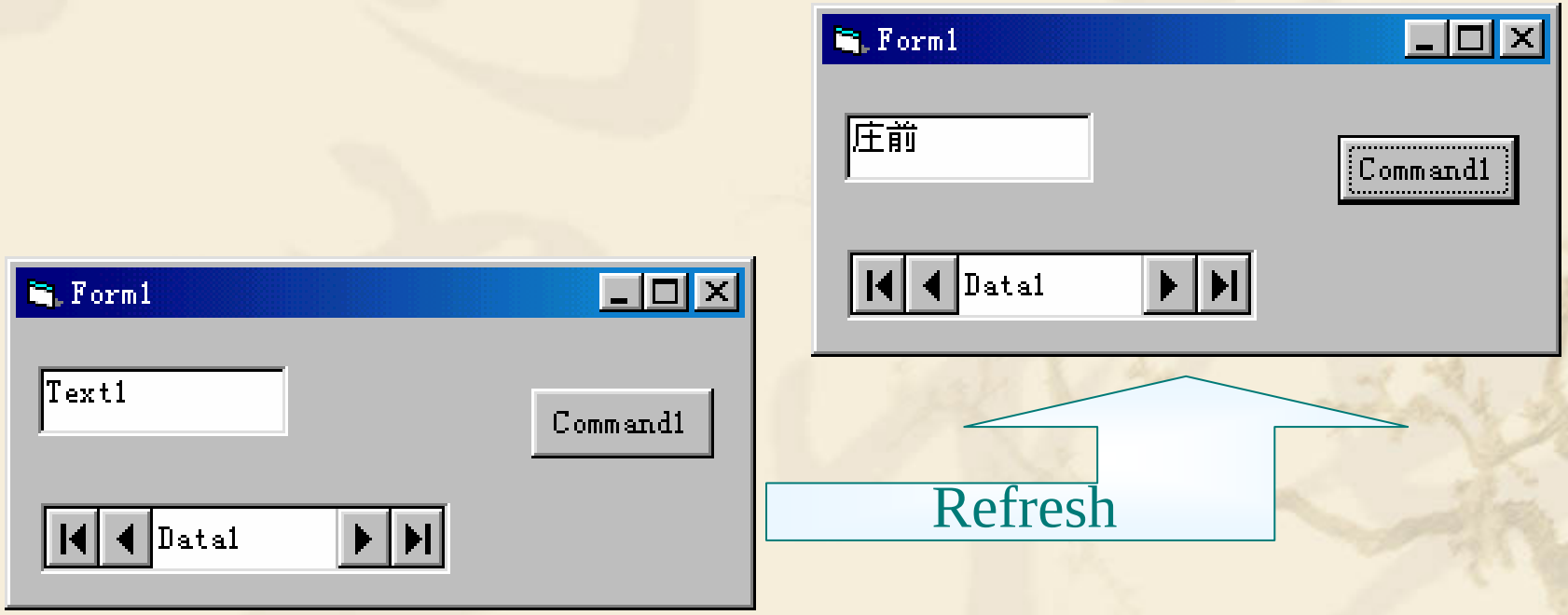
11.3.2 数据控件的事件

事　件	说　明
Reposition	发生在一条记录成为当前记录后，这个事件中显示当前指针的位置。
Vaildate	移动记录指针前、修改与删除记录前或卸载含有数据控件的窗体时触发该事件。参数Save可判断绑定控件内的数据是否发生变化。



11.3.3 数据控件的常用方法

方　法	说　明
Refresh	激活对数据控件属性的改变，使对数据库的操作有效。
UpdateControls	将数据从数据库中重新读到被数据控件绑定的控件内。
UpdateRecord	将绑定控件内的数据写入到数据库中而不触发数据控件的Vaildate事件。



11.3.4 记录集的属性与方法

1. 属性

2. 记录集的方法

•FindFirst、FindLast、FindNext、FindPrevious方法可在指定的Dynaset或Snapshot类型的记录集对象中查找。

•Seek方法在Table表中查找。

Nomarch属性可判定是否找到。

例11.3 在窗体上用Move方法代替数据控件对象的4个箭头的操作。

11.3.5 记录的增删改操作

学号	姓名	专业	课程	成绩
990001	万林	物理	数学	85
990002	庄前	物理	外语	90
990010	黎明	建筑	建筑学	90
填入新数据				

Data1.Recordset. AddNew　　增加记录

编辑记录
- 调用Edit方法。
- 给各字段赋值。
- 调用Update方法。

删除记录
调用Delete方法。
移动记录指针。

例11.5 对数据库提供增、删、改和查找功能。

11.4 ADO数据控件

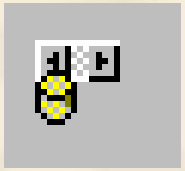
11.4.1 ADO对象模型

ADO是Microsoft处理数据库信息的最新技术，它是一种ActiveX对象，采用了被称为OLE DB的数据访问模式。它是数据访问对象DAO、远程数据对象RDO和开放数据库互连ODBC三种方式的扩展。ADO对象模型更为简化，不论是存取本地的还是远程的数据，都提供了统一的接口。

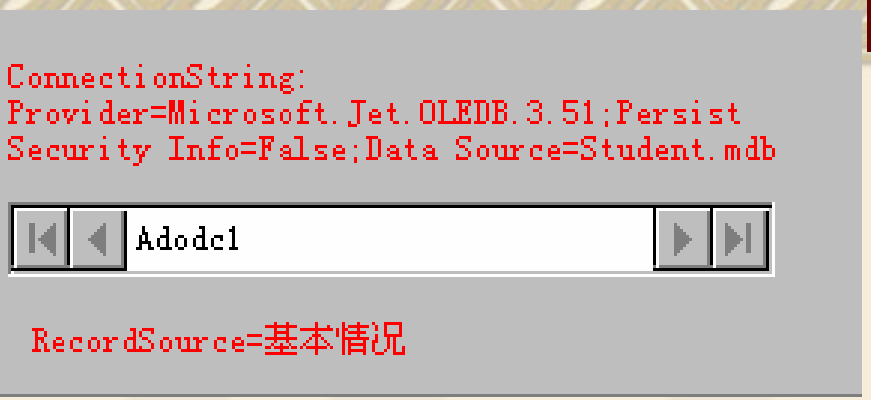
11.4.2 使用ADO数据控件

在使用ADO数据控件前，必须先通过“工程/部件”菜单命令选择“Microsoft ADO Data Control 6.0(OLE DB)”选项，将ADO数据控件添加到工具箱。ADO数据控件与Visual Basic的内部数据控件很相似，它允许使用ADO数据控件的基本属性快速地创建与数据库的连接。

工具箱内ADO控件图标形状



画在窗体上的外观

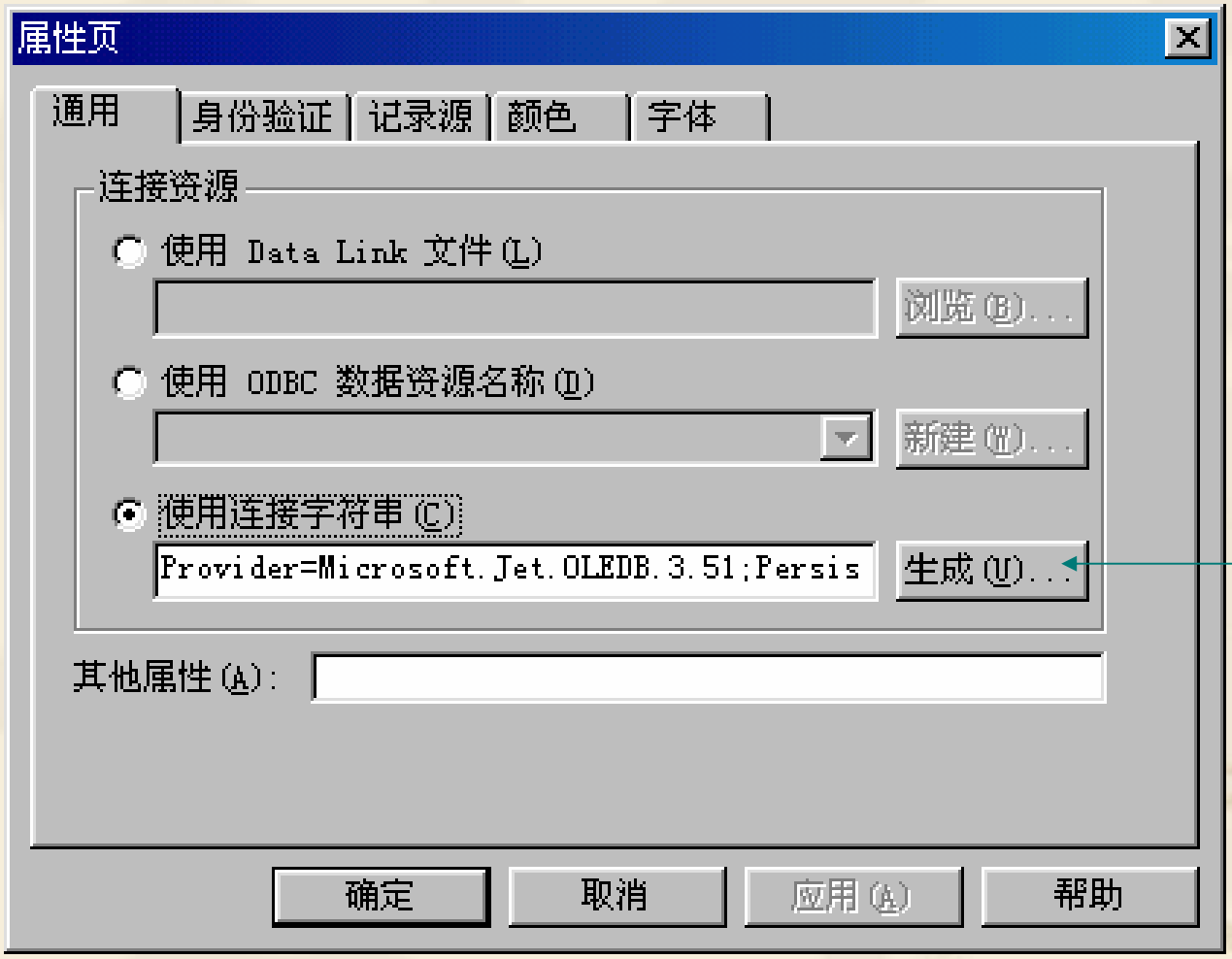


能够利用三种记录集对象访问数据库中的数据，连接方法：

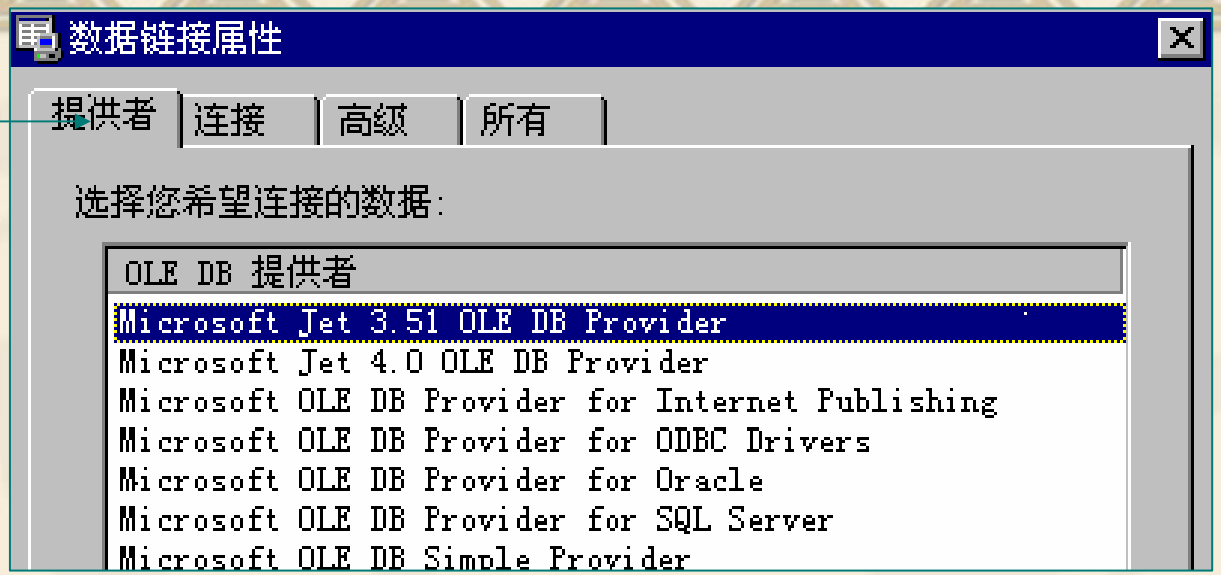
ADO控件连接设置

连接属性	ADO控件属性说明
ConnectionString	包含了用于与数据源建立连接的相关信息（ADO控件没有DatabaseName属性）。
RecordSource	确定具体可访问的数据，这些数据构成记录集对象Recordset。

连接操作------鼠标右击ADO控件，选择快捷菜单 “ADODC属性”命令，打开ADO控件属性页窗：

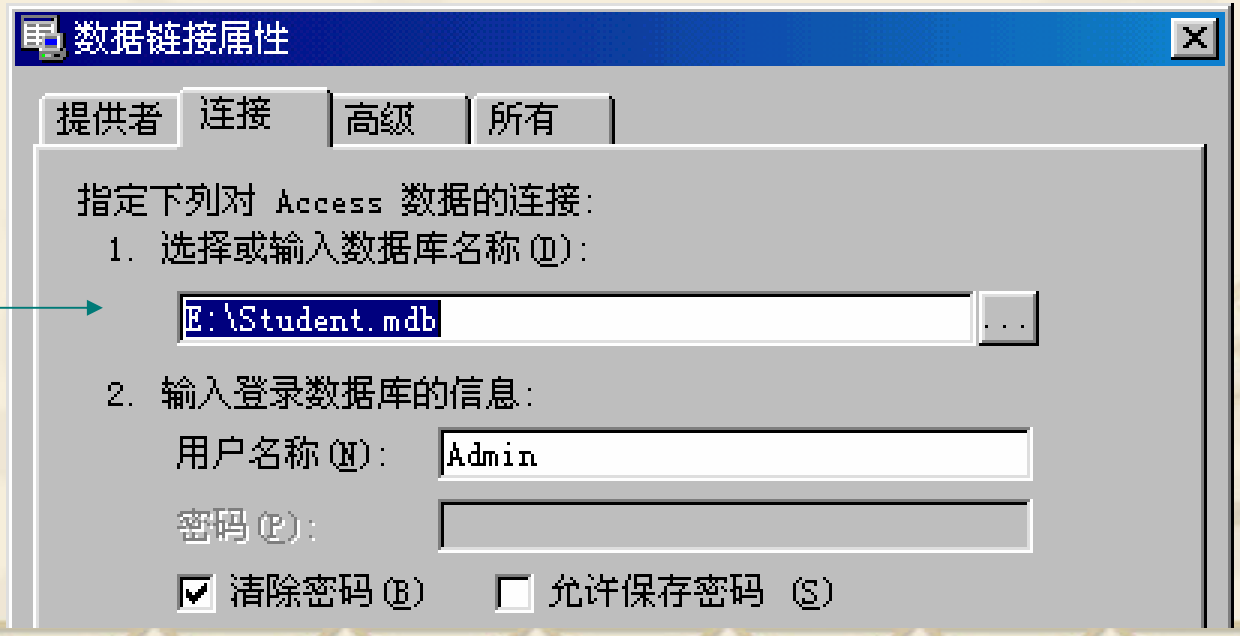


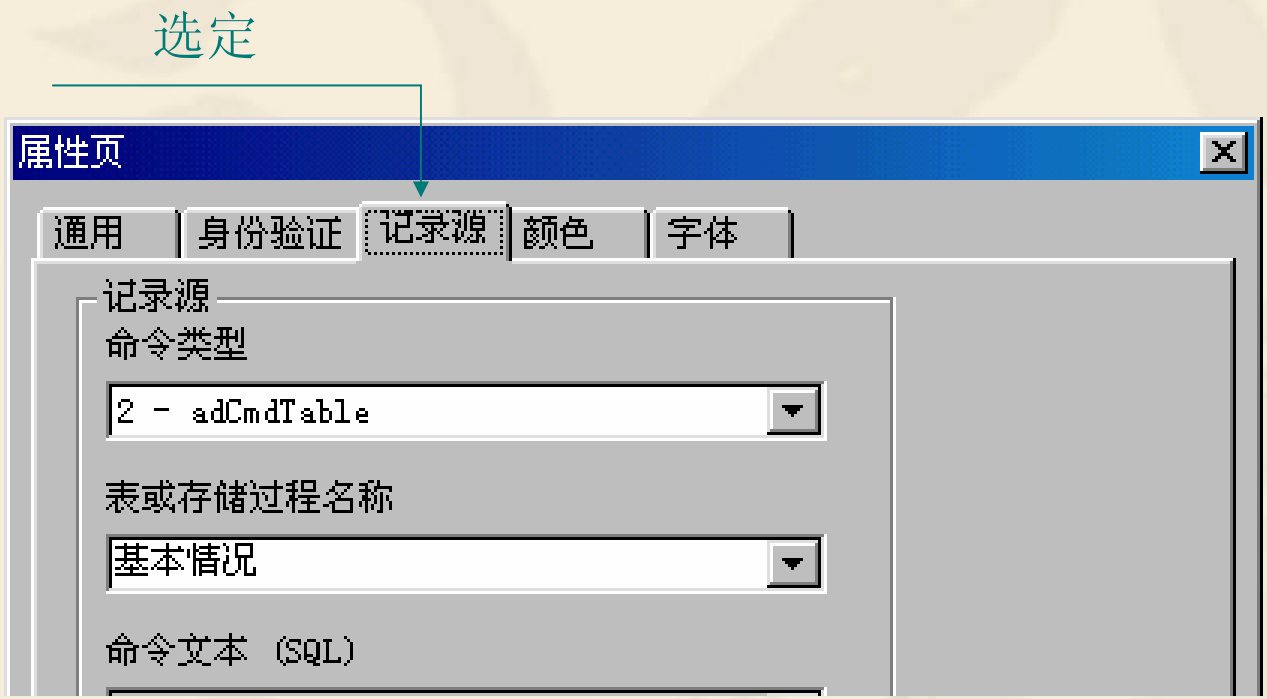
选定


数据链接属性
提供者
连接
高级
所有
选择您希望连接的数据:
OLE DB 提供者
Microsoft Jet 3.51 OLE DB Provider
Microsoft Jet 4.0 OLE DB Provider
Microsoft OLE DB Provider for Internet Publishing
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider

选定

数据库


数据链接属性
提供者
连接
高级
所有
指定下列对 Access 数据的连接:
1. 选择或输入数据库名称(D):
E:\Student.mdb
...
2. 输入登录数据库的信息:
用户名称(N):
Admin
密码(P):
清除密码(B)
允许保存密码(S)

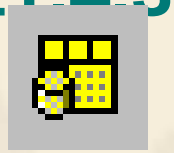


设置完成后，ADO控件的ConnectionString属性为：
Provider=Microsoft.Jet.OLEDB.3.51; Persist Security Info=False;
Data Source=Student.mdb

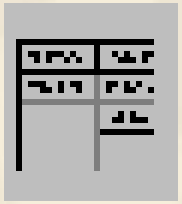
RecordSource属性为：基本情况（表）

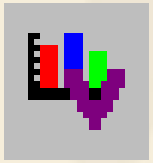
ADO控件的的其他操作与Data控件相同。

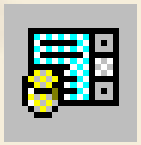
11.4.3 ADO控件上绑定控件的使用

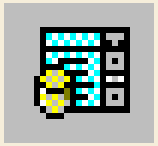

MSFlexGrid


DataGrid


MSHFlexGrid


MSChart


DataList


DataCombo

网格控件比较

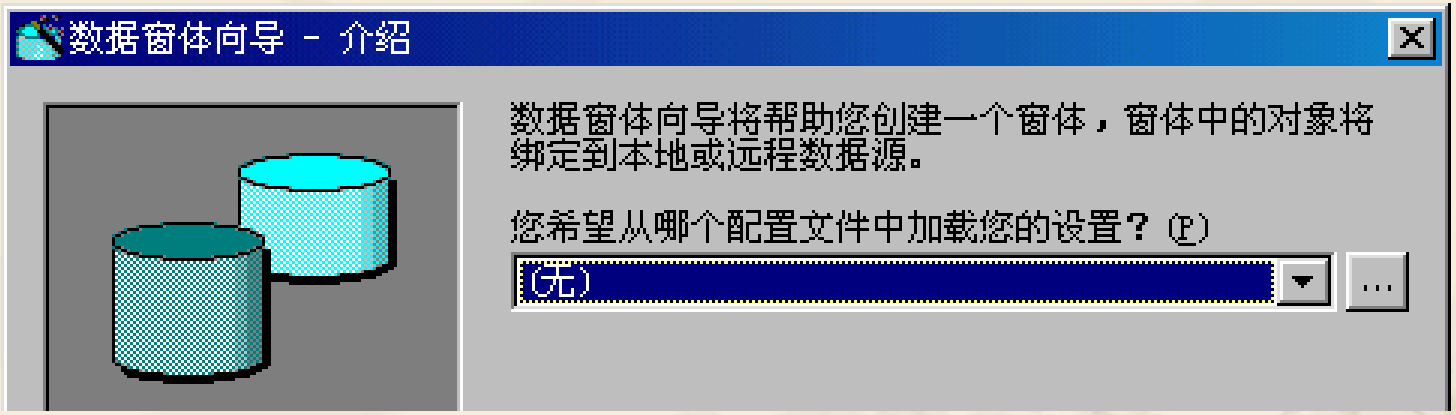
网格控件	分　类	功能说明
MSFlexGrid	标准	不能进行编辑，有图形功能。
MSHFlexGrid	OLEDB	不能进行编辑，可分层处理网格，有图形功能。
DataGrid	OLEDB	可以进行编辑操作，显示文本。

例11.6 使用ADO控件和DataGrid网格控件浏览数据库。

11.4.4 使用数据窗体向导

通过数据窗体向导能建立一个访问数据的窗口。在使用前必须执行“外接程序/外接程序管理器”命令，将“VB 6数据窗体向导”装入到“外接程序”菜单中。

步骤1：执行“外接程序”菜单中的“数据窗体向导”命令。



步骤2：选择数据库类型。


数据窗体向导 - 数据库类型
从列表中选择数据库格式(S)。
Access
Remote (ODBC)
帮助
取消
<上一步(B)
下一步(N)>
完成(F)

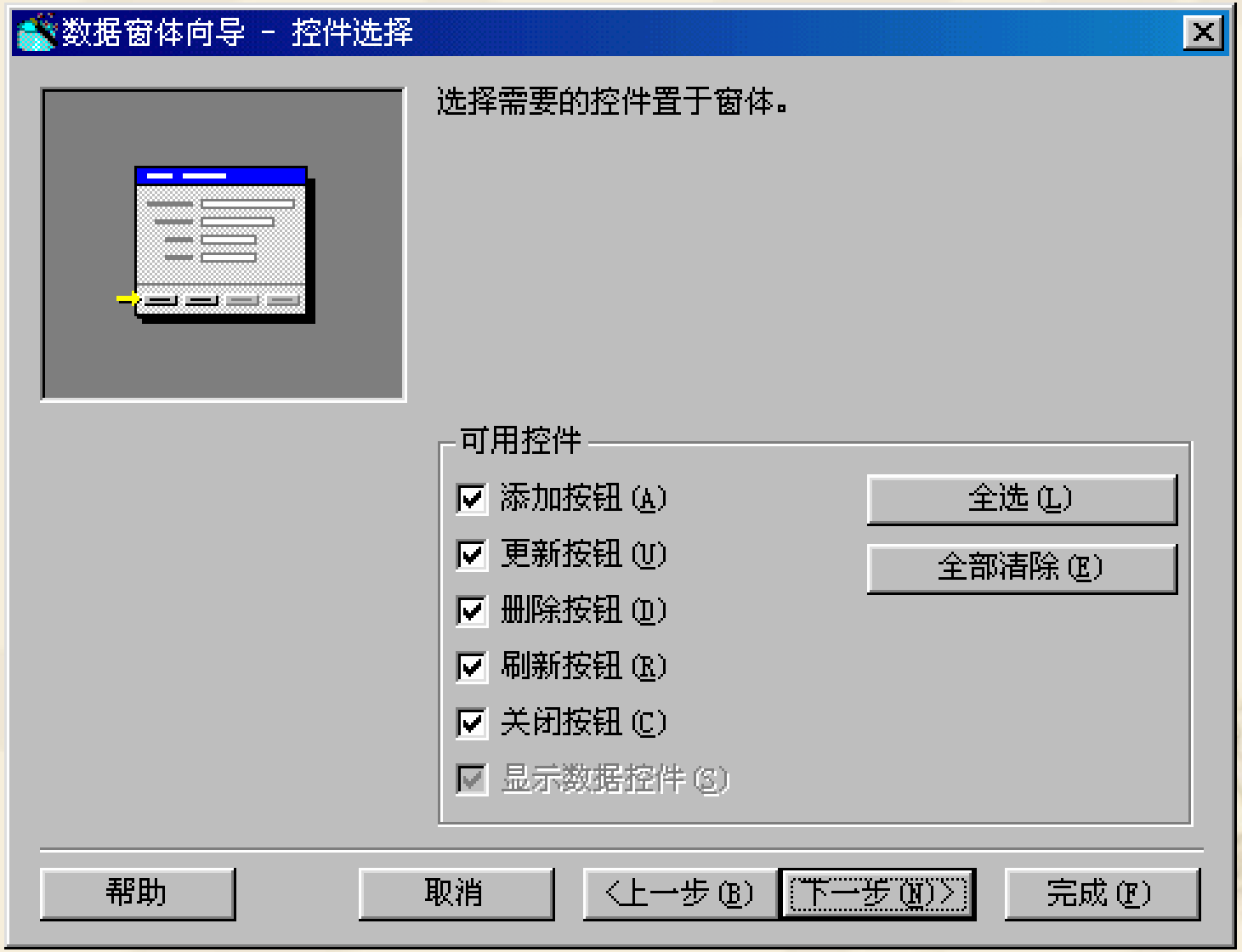
步骤3：选择具体的数据库文件。

数据库

步骤4：设置应用窗体的工作特性。

步骤5：选择记录源（所需要的实际数据）。

步骤6：选择所需要的操作按钮。 例11.7



11.5 结构化查询语言(SQL)

11.5.1 结构化查询语言

结构化查询语言SQL是操作数据库的工业标准语言。在SQL语言中，指定要做什么而不是怎么做。不需要告诉SQL如何访问数据库，只要告诉SQL需要数据库做什么。

利用SQL，可以确切指定想要检索的记录以及按什么顺序检索。可以在设计或运行时对数据控件使用SQL语句。用户提出一个查询，数据库返回所有与该查询匹配的记录。

结构化查询语言

常用SQL命令	描　述
CREAT	创建新的表、字段和索引
DELETE	从数据库表中删除记录。
SELECT	在数据库中查找满足特定条件的记录。
UPDATE	改变特定记录和字段的值。
常用SQL命令子句	**描　述**
FROM	用来为从其中选定记录的表命名。
WHERE	用来指定所选记录必须满足的条件。
GROUP BY	用来把选定的记录分成特定的组。
HAVING	用来说明每个组需要满足的条件。
ORDER BY	用来按特定的次序将记录排序。
合计函数	描　述
AVG	用来获得特定字段中的值的平均数
COUNT	用来返回选定记录的个数
SUM	用来返回特定字段中所有值的总和
MAX	用来返回指定字段中的最大值
MIN	用来返回指定字段中的最小值

11.5.2 使用SELECT语句查询

1. 使用SELECT语句

从数据库中的获取数据称为查询数据库，查询数据库通过使用SELECT语句。常见的SELECT语句形式为：

Select 字段表 From 表名 Where 查询条件 Group By 分组字段 Order By 字段[Asc|Desc]

可以在设计或代码中对数据控件的RecordSource属性设置SQL语句，也可将SQL语句赋予对象变量。

在建立SQL语句时，如果需要通过变量构造条件，则需要在应用程序中将变量连接到SELECT语句。例如：

"Select * From 基本情况 Where 专业 =' " & Text1 & " ' "

例11.8将例11.4中的查找功能改用SQL语句处理。

例11.9用SQL语句从两个数据表中选择数据构成记录集。

例11.10用SQL指令按专业统计Student.mdb数据库各专业的人数。

例11.11 在ADO数据控件上使用SQL语句。将例11.9中的Data控件改用ADO数据控件，用SQL语句从Student.mdb数据库的两个数据表中选择数据构成记录集。

例11.12 设计一个窗体，计算Student.mdb数据库内学生成绩表中每个学生的平均成绩，产生姓名、平均成绩和最低成绩三项数据，按平均成绩升序排列数据，并用该数据作图。

*2. 使用UPDATE语句修改记录

UPDATE创建一个更新查询来按照某个条件修改特定表中的字段值。其语法如下：

UPDATE [表集合] SET [表达式] WHERE [条件]

例11.12a 本例把学生平均成绩字段的值增加了10%，并刷新网格。

*3. 使用DELETE语句查询

可以创建删除查询来删除FROM子句中列出的、满足WHERE子句的一个或多个表中的记录，其语法所示如下：

DELETE [表字段] FROM [表集合] WHERE [条件]

例11.12b 删除例11.12a所产生temp表中平均成绩<80的全部记录，并刷新网格。

*11.5.3 使用对象变量访问数据库

DAO对象定义了一个可编程的对象集合。可按下列方法定义数据库对象和记录集对象，不必在窗体上放置数据控件。

方　法	说　明
Dim db As Database Set db = OpenDatabase(数据库)	打开数据库，返回一个Database类型的对象db。
Dim rs As Recordset Set rs = db.OpenRecordset(记录源)	返回一个记录集对象rs。

要想在程序中使用DAO对象，必须先为当前工程引用DAO对象的数据库引擎库。

引用方式：执行工程菜单的引用命令，启动引用对话框，在清单中选取“Microsoft DAO 3.51 Object Library”项目。

使用DAO模型访问数据库的例题。

例11.12d 使用DAO模型访问数据库，分页显示基本情况表的记录。

本程序通过Bookmark属性设置当前记录的书签，用于当前页上记录的定位。

例11.12e 使用DAO模型访问数据库，约束绑定控件。

记录集的字段对象 可以使用如下几种方法获取：

Recordset.Fields("字段名称" ）、Recordset("字段名称")

Recordset.Fields("数字"）、Recordset("数字"）

在DAO中记录集对象为对象变量，故字段可用rs(j)表示。

*11.5.4 ADO对象

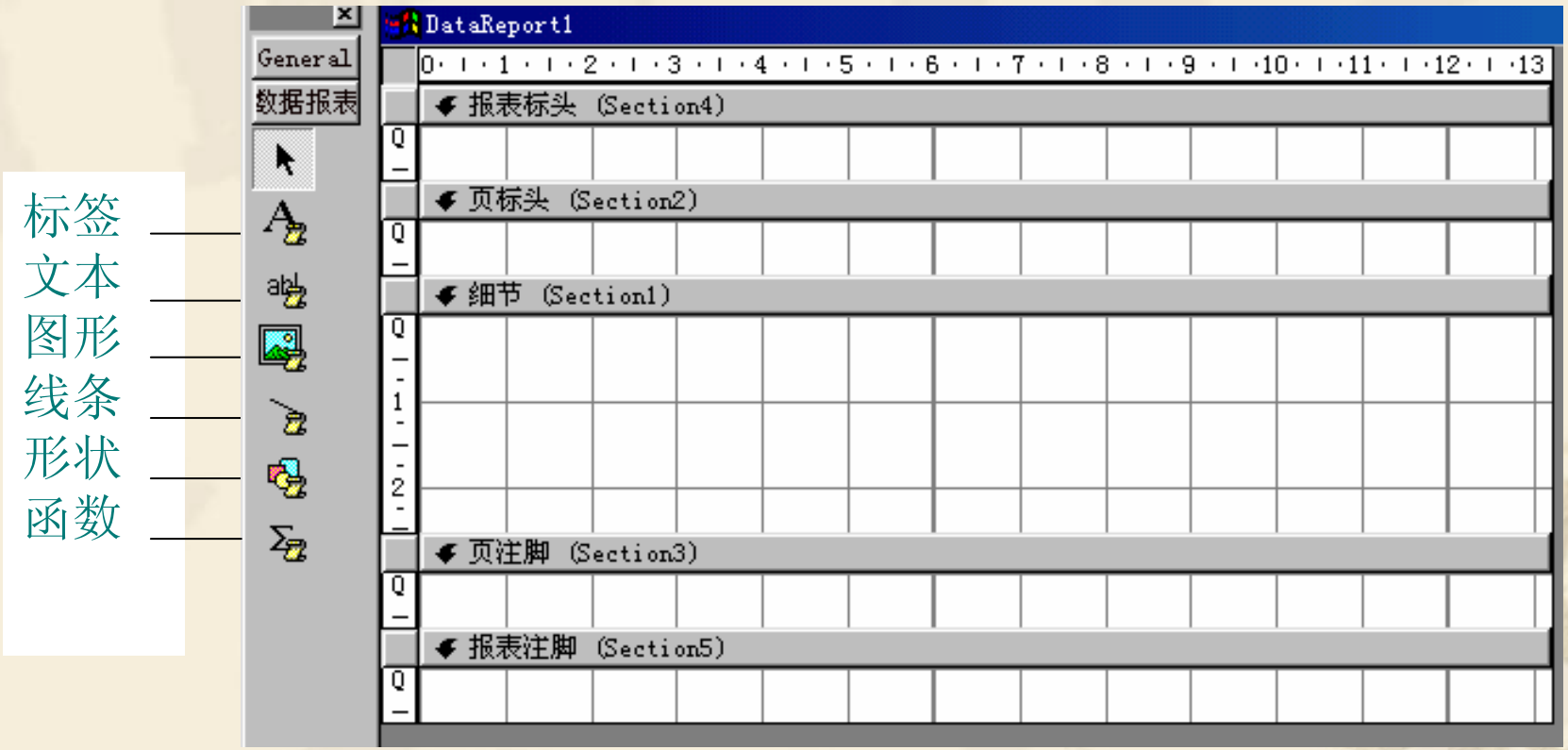
使用ADO对象访问数据库比DAO对象模型更为简化，不论是存取本地的还是远程的数据，都提供了统一的接口。

ADO的连接方法

方　法	说　明
Dim db As Connection Set db = New Connection db.Open 数据源连接字符串	连接数据源。 返回一个连接对象db。
Dim rs As Recordset Set rs = New Recordset rs.Open 数据源，连接对象，指针，锁类型	打开记录集。 返回查询结果。

11.6 报表制作

数据报表设计器属于ActiveX Designer组中的一个成员，在使用前需要执行“工程|添加Data Report”命令，将报表设计器加入到当前工程中，产生一个DataReport1对象，并在工具箱内产生一个“数据报表”标签。



- “标签”控件在报表上放置静态文本
- “文本”控件在报表上连接并显示字段的数据
- “图形”控件可在报表上添加图片
- “线条”控件在报表上绘制直线
- “形状”控件在报表上绘制各种各样的图形外形
- “函数”控件在报表上建立公式。
- 报表标头区包含整个报表最开头的信息，一个报表只有一个报表头，可使用“标签”控件建立报表名
- 报表注脚区包含整个报表尾部的信息，一个报表也只有一个注脚区
- 页标头区设置报表每一页顶部的标题信息；页注脚区包含每一页底部的信息；细节区包含报表的具体数据，细节区的高度将决定报表的行高。

例11.13 建立新工程，在窗体上放置两个命令按钮。

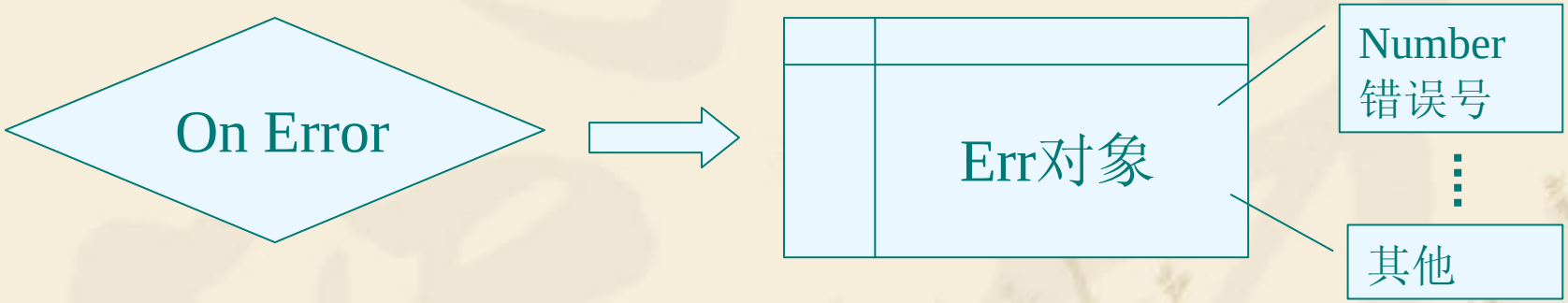
- 在当前工程内加入一个DataEnvironent1对象。完成与指定数据库的连接。在Connection1下创建Command1对象。
- 在当前工程中加入报表设计器DataReport1 ，设置报表设计器的DataSource属性为数据环境对象，DataMember属性为Command1对象。
- 将数据环境设计器中Command1对象内的字段拖动到数据报表设计器的细节区。
- 使用“标签”控件，在报表标头区插入报表名，页标头区设置报表每一页顶部的标题信息等。
- 使用“线条”控件在报表内加入直线，使用“图形”控件和“形状”控件加入图案或图形。
- 在命令按钮Click事件内加入代码DataReport1.Show显示报表，DataReport1.PrintReport打印报表。

11.7 错 误 处 理

错误处理步骤：

1. 用On Error语句设置错误陷阱，捕捉错误。
2. 编写错误处理程序，根据可预知的错误类型决定采取何种措施。

捕捉错误示意：



可根据Err.Number 的值来编写出错处理程序。

On Error语句有如下几种形式：

(1) On Error Resume Next ———— 忽略错误行，继续执行下一语句。

(2) On Error GoTo 标号————使程序转跳到语句标号所指示的程序块

(3) On Error GoTo 0————不使用错误处理程序块。

处理方法：

Resume ———— 重新执行引起错误的语句或指令。

Resume Next ———— 忽略错误行，继续执行下一语句。

Resume 标号 ———— 忽略错误行，转跳到由<标号>指明的语句继续执行。若<标号>为0，则表示终止程序执行。

例11.14 错误处理的基本方法

```
Sub OnErrorStatementDemo()
        Dim Response ,Msg, Style, Title As String
        Dim db As Database, rs As Recordset
        On Error GoTo ErrorHandler
        Set db = OpenDatabase("A:\Student.mdb")          '打开数据库
        Set rs = db.OpenRecordset("基本情况")            '设置记录集
                … …
        Exit Sub                                         '正常退出本程序
ErrorHandler:                                            '错误处理块入口
        Style = vbRetryCancel + vbCritical + vbDefaultButton2
        Select Case Err.Number
        Case 53                                  '出错号53为文件不存在
        Msg = "文件不存在!"
        Response = MsgBox(Msg, Style, "出错提示")
        If Response = vbCancel Then Exit Sub
        Case 71                                  '71号驱动器未准备好
        Msg = " A盘未准备好"
        Response = MsgBox(Msg, Style, "出错提示")
        If Response = vbCancel Then Exit Sub
        Case Else' 当发生其他不可预知的错误，退出本程序
        Exit Sub
        End Select
        Resume              '再次执行原出错语句
End Sub
```

例11.15 下列程序用于处理删除一个已打开的文件所产生的错误。

```
Sub ResumeStatementDemo()
        On Error GoTo ErrorHandler        ' 设置错误陷阱
        Open " testfile " For Output As #1
        Kill "testfile"                   ' 企图删除一个打开的文件
        Exit Sub
ErrorHandler:
        Select Case Err.Number
        Case 55                           ' 文件已打开错误
                Close #1                  ' 处理方法：关闭文件
        Case Else
                … …
        End Select
        Resume
End Sub
```

例11.16　采用忽略错误的方法处理错误。

```
Sub OnErrorStatementDemo()
        On Error GoTo ErrorHandler          ' 设置错误陷阱
        Open " testfile " For Output As #1
        Kill " testfile "                   ' 企图删除一个打开的文件
        On Error Goto 0                     ' 关闭错误陷阱
        On Error Resume Next                ' 忽略Kill "testfile"出错行指令
        Err.Clear                           ' 清除出错号
        Exit Sub
ErrorHandler:
        Select Case Err.Number
        Case 55
                Close #1
        Case Else
                … …
        End Select
        Resume
End Sub
```

例11.17 处理被零除、溢出和非法的过程调用三种情况产生的错误。

```
Function Divide (numer, denom) as Variant
        Const mnErrDivByZero = 11          ' 用于代表被零除
        Const mnErrOverFlow = 6            ' 溢出
        Const mnErrBadCall = 5             ' 非法的过程调用
        On Error GoTo MathHandler
        Dim Msg as String
        Divide = numer / denom             ' 计算
        Exit Function
MathHandler:
    If Err.Number = MnErrDivByZero Or Err.Number = ErrOverFlow _
Or Err = ErrBadCall Then
        Divide = Null                      '产生错误则返回 NULL
        Else
        ' 显示意想不到的错误信息。
        Msg = "Unanticipated error " & Err.Number
        Msg = Msg & ": " & Err.Description
        MsgBox Msg, vbExclamation
    End If
    Resume Next          ' 不管什么情况，Resume Next
End Function
```