目录

一

1、MingW 就是一个 C/C++的编译器。

编译器和编辑器：编译器其实就是把源代码编译成目标代码的程序；Tc 就是一个编辑器：能在里边敲代码，高亮显示，自动提示。

2、

C-Free 里边可以运行、也可以调试。设置断点很简单，就是在要设置断点的代码行前边点击一下鼠标，有了一个红色小点就说明设置好了断点。

3、控制台程序的入口是 main，Win32 程序的入口是 WinMain

4、以前用 TuborC 的同学会问“什么叫工程？”。在以前大家用 TurboC 写的程序一般也就是一个文件，但是用 C 语言写大程序的时候不可能把所有的代码都写在一个文件中，肯定要写很多文件。“工程（Project）”就是这些问题的一个集合。）在“工程名”中写入你为这个工程取的名字，最好有一定意义。

5、Windows 程序的骨干代码：

```
#include <windows.h>
int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance,
PSTR szCmdLine, int iCmdShow)
{
return 0 ;
}
```

6、

向 Windows 的世界问好

（1）MessageBox(NULL,TEXT("世界你好"),TEXT("问好"),MB_OK);

第一个参数暂时不讲；第二个是正文；第三个是标题（Caption）；第四个是类型。MB_OK 就是表示有一个 OK 按钮【确定】）

TEXT 是一个宏，当字符串中有中文的时候最好用 TEXT 来包围这个字符串，虽然不使用 TEXT 在 VC6 中没问题，但是在 VC7 中有问题，而且微软也建议使用 TEXT 宏，因此在涉及到中文的场合要使用它。他的作用就是把中文转化成不会乱码的格式。（ 暂时这么认为 ）。_T("问好")，其实_T 只是 TEXT 的一个缩写而已。而且_T 在有的低版本里不识别。TEXT()低版本也识别。

换个样子：

```
MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),MB_OKCANCEL);
```

可从设置希望在对话框中显示的按钮：

```
#define MB_OK
0x00000000L
#define MB_OKCANCEL
0x00000001L
#define MB_ABORTRETRYIGNORE
0x00000002L
#define MB_YESNOCANCEL
0x00000003L
#define MB_YESNO
0x00000004L
#define MB_RETRYCANCEL
0x00000005L
```

也可以设置对话框中显示的图标：

```
#define MB_ICONHAND
0x00000010L
#define MB_ICONQUESTION
0x00000020L
#define MB_ICONEXCLAMATION
0x00000030L
#define MB_ICONASTERISK
0x00000040L
```

比如：MessageBox(NULL, TEXT("世界你好") ,TEXT("你好"), MB_ICONQUESTION);

（ 2 ）可是如果我想显示“确定、取消”按钮的时候同时使用问号图标呢？

MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),MB_OKCANCEL| MB_ICONQUESTION);

MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),

MB_ABORTRETRYIGNORE | define MB_ICONEXCLAMATION);

（3）思考：

到这一步如果你只是在背诵“第一组和第二组中间放一个竖线就可以实现两种效果的组合形式”，那么你未来只会成为一个代码工人。我们要思考一下其背后的原理。

首先说“|”是什么意思？“|”是位运算里的“或”运算，只有对应的两个二进位有一位为 1 时，结果位才为 1 ，否则为 0。MB_OK、MB_OKCANCEL 等的低四位不同，但是高位永远为 0；MB_ICONHAND、MB_ICONQUESTION 等的低 5 至第 8 位不同，而其他位永远为 0。这样“MB_OK、MB_OKCANCEL”组的数值与“MB_ICONHAND、MB_ICONQUESTION”组的数值进行或运算后能分别保留各自的部分，也就是在结果值中同时体现两组的取值。

拿到 1000010 以后怎么判断是不是和 0000010 或运算来的？

1000010

0000010 与&

======

0000010

1000011

0000010 &

=======

0000010

这种风格叫“掩码”，在 Windows 编程中这种用法会经常用到。

大家想像一下 MessageBox 的内部实现是怎么样的？

```
if((mode& MB_OK)== MB_OK)
{
//显示 OK 按钮
}
if((mode&
MB_ICONQUESTION)== MB_ICONQUESTION)
{
```

//显示问好

}

itoa、atoi 定义在 stdlib.h 中。

（4）默认按钮

为什么要有默认按钮：一个方便用户，不用思考，直接点回车就可以选择默认按钮；防止用户误操作，默认的按钮应该是最优选的按钮。“是否保存文件？”默认应该是“确定”，“是否发射导弹？”默认应该是“取消”。

还可以指定哪个按钮是默认按钮，猜一下这些选项的取值的特征是什么？

#define
MB_DEFBUTTON1
0x00000000L#define
MB_DEFBUTTON2
0x00000100L#define
MB_DEFBUTTON3
0x00000200L#define
MB_DEFBUTTON4

0x00000300L 示例代码：

MessageBox(NULL, TEXT("你是人吗？"), TEXT("火星人"), MB_YESNO | MB_ICONQUESTION|
MB_DEFBUTTON2);

（5）返回值

MessageBox 是有返回值的，返回值为用户点击的按钮：

#define IDOK
1
#define IDCANCEL
2
#define IDABORT
3
#define IDRETRY
4
#define IDIGNORE
5
#define IDYES

6

#define IDNO

7

代码：

```
int ret = MessageBox(NULL, TEXT("你是人吗？"), TEXT("火星人"),
MB_YESNO | MB_ICONQUESTION);
if(ret==IDYES)
{
MessageBox(NULL, TEXT("火星人你好"), TEXT("问好"),MB_OK);
}
else
{
MessageBox(NULL, TEXT("欢迎回家来"), TEXT("问好"),MB_OK);
}
```

7、课后作业：自己动手写恶搞程序。

（1）运行以后弹出询问对话框（有【是】、【否】两个按钮，默认选择【否】按钮，以及问号图标）“你是好人吗？”，如果点击【是】，则弹出对话框（只有一个【确定】按钮以及一个警告图标）“你看你就不像好人，点击【确定】开始格式化 C 盘！”；如果点击【否】，则弹出对话框（有【重试】、【取消】两个按钮）“尝试把你变成好人失败，是否重试？”。

（2）自己发挥完善上边的恶搞程序。1、MingW 就是一个 C/C++的编译器。

编译器和编辑器：编译器其实就是把源代码编译成目标代码的程序；Tc 就是一个编辑器：能在里边敲代码，高亮显示，自动提示。

2、

C-Free 里边可以运行、也可以调试。设置断点很简单，就是在要设置断点的代码行前边点击一下鼠标，有了一个红色小点就说明设置好了断点。

3、控制台程序的入口是 main，Win32 程序的入口是 WinMain

4、以前用 TuborC 的同学会问“什么叫工程？”。在以前大家用 TurboC 写的程序一般也就是一个文件，但是用 C 语言写大程序的时候不可能把所有的代码都写在一个文件中，肯定要写很多文件。“工程（Project）”就是这些问题的一个集合。）在“工程名”中写入你为这个工程取的名字，最好有一定意义。

5、Windows 程序的骨干代码：

```
#include <windows.h>
int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance,
PSTR szCmdLine, int iCmdShow)
{
return 0 ;
}
```

6、

向 Windows 的世界问好

（1）MessageBox(NULL,TEXT("世界你好"),TEXT("问好"),MB_OK);

第一个参数暂时不讲；第二个是正文；第三个是标题（Caption）；第四个是类型。

MB_OK 就是表示有一个 OK 按钮【确定】）

TEXT 是一个宏，当字符串中有中文的时候最好用 TEXT 来包围这个字符串，虽然不使用 TEXT 在 VC6 中没问题，但是在 VC7 中有问题，而且微软也建议使用 TEXT 宏，因此在涉及到中文的场合要使用它。他的作用就是把中文转化成不会乱码的格式。（暂时这么认为）。_T("问好")，其实_T 只是 TEXT 的一个缩写而已。而且_T 在有的低版本里不识别。TEXT()低版本也识别。

换个样子：

MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),MB_OKCANCEL);

可从设置希望在对话框中显示的按钮：

```
#define MB_OK
0x00000000L
#define MB_OKCANCEL
```

0x00000001L
#define MB_ABORTRETRYIGNORE
0x00000002L
#define MB_YESNOCANCEL
0x00000003L
#define MB_YESNO
0x00000004L
#define MB_RETRYCANCEL
0x00000005L

也可以设置对话框中显示的图标：

#define MB_ICONHAND
0x00000010L
#define MB_ICONQUESTION
0x00000020L
#define MB_ICONEXCLAMATION
0x00000030L
#define MB_ICONASTERISK
0x00000040L

比如：MessageBox(NULL, TEXT("世界你好") ,TEXT("你好"), MB_ICONQUESTION);

（2）可是如果我想显示“确定、取消”按钮的时候同时使用问号图标呢？

MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),MB_OKCANCEL| MB_ICONQUESTION);

MessageBox(NULL, TEXT("世界你好"), TEXT("你好"),
MB_ABORTRETRYIGNORE | define MB_ICONEXCLAMATION);

（3）思考：

到这一步如果你只是在背诵“第一组和第二组中间放一个竖线就可以实现两种效果的组合形式”，那么你未来只会成为一个代码工人。我们要思考一下其背后的原理。

首先说“|”是什么意思？“|”是位运算里的“或”运算，只有对应的两个二进位有一位为 1 时，结果位才为 1 ，否则为 0。MB_OK、MB_OKCANCEL 等的低四位不同，但是高位永远为 0；MB_ICONHAND、MB_ICONQUESTION 等的低 5 至第 8 位不同，而其他位永远为 0。这样“MB_OK、MB_OKCANCEL”组的数值与“MB_ICONHAND、MB_ICONQUESTION”组的数值进行或运算后能分别保

留各自的部分，也就是在结果值中同时体现两组的取值。

拿到 1000010 以后怎么判断是不是和 0000010 或运算来的？

1000010

0000010 与&

======

0000010

1000011

0000010 &

=======

0000010

这种风格叫“掩码”，在 Windows 编程中这种用法会经常用到。

大家想像一下 MessageBox 的内部实现是怎么样的？

```
if((mode& MB_OK)== MB_OK)
{
//显示 OK 按钮
}
if((mode&
MB_ICONQUESTION)== MB_ICONQUESTION)
{
//显示问好
}
```

itoa、atoi 定义在 stdlib.h 中。

（4）默认按钮

为什么要有默认按钮：一个方便用户，不用思考，直接点回车就可以选择默认按钮；防止用户误操作，默认的按钮应该是最优选的按钮。“是否保存文件？”默认应该是“确定”，“是否发射导弹？”默认应该是“取消”。

还可以指定哪个按钮是默认按钮，猜一下这些选项的取值的特征是什么？

```
#define
MB_DEFBUTTON1
0x00000000L#define
```

MB_DEFBUTTON2
0x00000100L#define
MB_DEFBUTTON3
0x00000200L#define
MB_DEFBUTTON4

0x00000300L 示例代码：

```
MessageBox(NULL, TEXT("你是人吗？"), TEXT("火星人"), MB_YESNO | MB_ICONQUESTION|
MB_DEFBUTTON2);
```

（5）返回值

MessageBox 是有返回值的，返回值为用户点击的按钮：

```
#define IDOK
1
#define IDCANCEL
2
#define IDABORT
3
#define IDRETRY
4
#define IDIGNORE
5
#define IDYES
6
#define IDNO
7
```

代码：

```
int ret = MessageBox(NULL, TEXT("你是人吗？"), TEXT("火星人"),
MB_YESNO | MB_ICONQUESTION);
if(ret==IDYES)
{
MessageBox(NULL, TEXT("火星人你好"), TEXT("问好"),MB_OK);
}
else
{
MessageBox(NULL, TEXT("欢迎回家来"), TEXT("问好"),MB_OK);
}
```

7、课后作业：自己动手写恶搞程序。

（1）运行以后弹出询问对话框（有【是】、【否】两个按钮，默认选择【否】按钮，以及问号图标）“你是好人吗？”，如果点击【是】，则弹出对话框（只有一个【确定】按钮以及一个警告图标）“你看你就不像好人，点击【确定】开始格式化 C 盘！”；如果点击【否】，则弹出对话框（有【重试】、【取消】两个按钮）“尝试把你变成好人失败，是否重试？”。

（2）自己发挥完善上边的恶搞程序。

二

1、上节课作业点评：

海豚天使作业：写的非常好，99 分。唯一的改进点就是“if else 中的代码哪怕只有一行也要用大括号括起来”。

VansOS：一个可取之处是“TEXT()”的应用；把 while……continue 用的很好；符合了“if else 中的代码哪怕只有一行也要用大括号括起来”的要求。100 分。

3、怎么打开 VC6、VC7、DevC++的工程？

C-Free4.0 用【工具】→【工程转换】。

C-Free4.1 的话，直接把工程文件拖进来就可以。

大家以后从网上下载了别的写的 C 语言程序，有可能用 VC 写的，所以要转换一下。下了 VC 的程序不用担心，C-Free 是支持的。语言、IDE 都是互通的，“**一通百通**”。

4、为什么代码自动提示不出来？怎么增加自动提示的数量？

【工具】→【编辑器选项】→【代码提示】。显示最大条数、输入几个字符后才激活。建议改成 10、1

5、怎么修改新建文件的默认文件名为 c。

【工具】→【环境选项】、修改“新建文件类型”

6、每次都输入 WinMain 以及那些参数，很麻烦，有没有快速方式？

添加自己的“模板”。【工具】→【编辑器选项】→【代码模板】，点击【添加】按钮，为模板选一个名字，然后将代码模板粘贴到“代码”框中。

使用方式：在编辑器中点击鼠标右键，选择【模板】，然后选择创建的代码模板。

7、怎么生成 exe 程序。怎么把做好的 exe 程序发给别人？

主菜单【工具】→【定位到工程文件夹】跳转到工程的文件夹，也可以在我的电脑里直接进入工程文件夹。mingw2.95 目录下生成的 exe 文件就是生成好的可执行文件 ,发给别人就可以。

8、播放声音文件

（1）用途：**游戏里播放音乐；自己动手给女友做音乐贺卡（图片、滚动的字母、温馨的音乐）**

（2）可以使用 **PlaySound**()函数播放声音文件，该函数原型位于#include <mmsystem.h>中，因此要使用 PlaySound，首先需要添加对这个头文件的引用。

提示：**mm 就是 MultiMedia 的简写，多媒体**

（3）函数原型为： BOOL PlaySound(LPCSTR pszSound, HMODULE hmod,DWORD fdwSound);

返回值表示是否播放成功。

参数 pszSound 是要播放声音的文件名 ,只支持 WAV 等格式的文件 ;去 mp3.baidu.com 用“wav”为关键词可以搜到很多 wav 音乐，也可以用工具将其他格式的转换为 wav 格式的。

参数 hmod 是应用程序的实例句柄，一般传递 NULL 就可以；

参数 fdwSound 是标志的组合掩码，可选值有 SND_FILENAME、SND_ASYNC、SND_SYNC 等。SND_FILENAME 表示 pszSound 参数指定的是文件名（pszSound 还可以指定资源、内存音乐、系统音乐等等）；SND_ASYNC：用异步方式播放声音，PlaySound 函数在开始播放后立即返回；SND_SYNC：同步播放声音，在播放完后 PlaySound 函数才返回；SND_LOOP 一遍遍的重复播放声音，必须与 SND_ASYNC 标志一块使用。

（4）使用举例：

```
PlaySound(TEXT("C:\\WINDOWS\\Media\\Windows XP 启动.wav"),NULL,SND_FILENAME|SND_SYNC);
```

注意的问题：文件名中的反斜线要用“\\”，因为 C 语言中“\”默认是转义符，如果要表示“\”则需要使用“\\”，对这点不明白的请回去翻一下 C 语言的书；对中文字符串要使用 TEXT 宏。

（5）为啥构建不通过？

#include <mmsystem.h>只是保证编译通过，还要设定 link。

解决方案：工程上点击右键，选“工程设置”，连接，添加“winmm”库。

解决此问题时参考的文章：http://hi.baidu.com/big_foot/blog/item/4822fcd11a7f7cd6562c84e7.html。它虽然描述的是 VC 中解决此问题的方式，但是记住“**一通百通**”这个道理，C-Free 中添加 Link 库的方式和 VC 非常相似，只是操作步骤不同而已。

附录：VC 中添加 Link 库的方式：

project->setting->Link 下的 Object/library modules 里加入 winmm.lib 即可”，他这里用的是英文版，中文版应该是主菜单的【工程】→【设置】→【连接】，将“winmm.lib”加入到“对象/库模块”中。

（6）接上面的问题：关于 link 库

一个程序由源代码变成 exe 文件有两步：编译 Compile；连接 Link。编译是把 c 文件编译成.o、.obj 文件，而连接则是把这些.o、.obj、.lib 等文件连接到一起成为 exe 文件。这点不清楚的请回去查看 C 语言的教材。

那什么是*.lib 文件呢？大家以前用 TC 开发程序的时候用的都是 C 语言内置的函数，所以不存在使用非内置函数的情况。但是在开发大程序的时候要大量用到非内置的函数，比如 PlaySound、MessageBox 等等。在使用这些函数的时候需要使用两部分，一部分是*.h 头文件，它定义了函数的参数和返回值，另一部分是*.lib 文件，是用来进行程序链接用的。**C 程序的构建分为编译（Compile）和连接（Link）两个过程**，Compile 是把源代码编译成*.obj 文件，每个源

码文件都对应一个*.obj 文件，而连接则是把这些*.obj 文件以及使用到的非内置函数的*.lib 文件连接成一个*.exe 文件。所以无论是忘了 include *.h 文件，还是丢了*.lib 文件，都会构建出错。从上面的报错信息可以看出来是 Compile 成功，但是 Link 失败。一定要记得 C 程序编译的这两个过程，面试、笔试的时候常考。

（7）同步播放音乐

```
PlaySound("C:\\WINDOWS\\Media\\Windows XP 关机.wav",NULL,SND_FILENAME|SND_SYNC);
```

（8）体会同步播放和异步播放的差异：

```
PlaySound("C:\\WINDOWS\\Media\\Windows XP 关机.wav",NULL,SND_FILENAME|SND_SYNC);
MessageBox(NULL,"同步播放完毕","信息",MB_OK);
PlaySound("C:\\WINDOWS\\Media\\Windows XP 关机.wav",NULL,SND_FILENAME|SND_ASYNC);
MessageBox(NULL,"异步播放立即返回","信息",MB_OK);
```

同步模式：音乐播放过程中函数不返回，播放完成才返回

异步播放在做游戏等需要播放时间较长的音乐时使用，因为程序不会在音乐播放过程中有假死的情况。

有没有同学注意到第二个对话框如果快速关闭的话音乐会中途停止？这是因为程序退出音乐就中断播放了。用户关了游戏，音乐也不能继续播放，应该停止，所以很合理

（9）我可以做一个小程序，别人在我小程序运行的时候一直听播放的音乐，用异步方式可以保证音乐在后台播放。可以只能播放一遍，能不能音乐来回来去的不断播放呢？答案是使用 SND_LOOP 标志。

Loop：循环

```
PlaySound("C:\\WINDOWS\\Media\\Windows XP 关机.wav",
NULL,SND_FILENAME|SND_ASYNC|SND_LOOP);
```

（10）如果我做一个小游戏，那么希望向上边那样不断播放，可以到达一个关卡的时候希望停止原来的音乐怎么办？

```
PlaySound(TEXT("C:\\WINDOWS\\Media\\Windows XP 启动.wav"),NULL,SND_FILENAME|SND_ASYNC);
MessageBox(NULL,TEXT(""),TEXT(""),MB_OK);
PlaySound(TEXT("C:\\WINDOWS\\Media\\Windows XP 关机.wav"),NULL,SND_FILENAME|SND_ASYNC);
MessageBox(NULL,TEXT(""),TEXT(""),MB_OK);
```

只能同时播放一段音乐，启动新的、旧的就被停了

（11）如果只是想停止目前的播放而不播放新音乐呢？

只要给 PlaySound 的第一参数传递 NULL 就可以停止目前的播放了。

```
PlaySound("C:\\WINDOWS\\Media\\Windows XP 关机.wav",
NULL,SND_FILENAME|SND_ASYNC|SND_LOOP);
MessageBox(NULL,"点确定终止音乐","信息",MB_OK);
PlaySound(NULL,NULL,SND_FILENAME);
MessageBox(NULL,"音乐被终止","信息",MB_OK);
```

（12）关于 PlaySound 函数更多的介绍见：http://zhidao.baidu.com/question/41366091.html

9、关于 API

（1）什么叫 API(应用程序接口 Alication Interface 的简称)？什么叫 Win32 API？API 就是操作系统提供的一堆库函数，没啥稀奇的。printf、scanf 是 C 语言内置的函数，其他的非内置的库函数都叫 API。

（2）我想找实现某个功能的 API 怎么办？以“关闭显示器”为例讲解自学过程。

（3）怎么我查看 MessageBox 函数其实是 MessageBoxW、MessageBoxA 两个函数的宏定义？

Win32API 中还有大量这种风格的函数。W 结尾的是把字符当成 Unicode 处理的，A 结尾的是把字符当成 ASCII 处理的。使用时不要直接调用 W 结尾的或者 A 结尾的。编译器会在编译时确定调用哪个。

（4）Win32API 是语言无关的，这点是非常重要的。Win32API 是操作系统提供的库函数，**可以在 C 语言中调用，也可以在 C++中调用，还可以在 C#、Java、Delphi、Python、汇编等各种语言中调用**，因此**学会了我们这里的 C 语言 Windows 程序开发以后只要熟悉一下其他语言的语法就可以很快的用其他语言开发** Windows 程序，VC、C#、Delphi 等语言中的程序界面、网络操作、文件操作等功能都是对 Win32API 的简单包装而已。学了咱们如鹏网的《C 语言也能干大事》以后就掌握了编程开发的"**内功心法**"，达到了**一切语言都是纸老虎**的**大侠**状态，就可以蔑视那些问"**学 VC 还是学 C#？**"的**菜鸟**，**你怎能不心动？！**

10、课后作业：改进上节课那个恶搞程序，在问不同问题的时候播放不同的音乐。同学提交作业的方式就是把代码以回帖的方式回复本帖。

三

1、作业点评：CALF 的程序满足了所有要求，而且对 do...while、switch...case 等 C 语言的语法用的很好，注意了很多细节问题。因此给 100 分。

2、同学问题：怎么给程序添加图标

把一个 ico 图标放到工程下；使用 ResEd 在工程下新建一个资源文件（工程），然后点击【工程】→【资源】，添加一个 ICON 类型的资源，

文件名选择刚才的 ico 图标，然后保存这个资源文件，保证这个资源的 ID 为最小（一般设置为 0 就可以）。在 C-Free 里的 OtherFiles 中点右键，选择“添加文件到文件夹”，将 rc 文件加入，重新编译即可。

注意：有的同学反应添加图标以后程序还是没有图标，有可能是操作不当，有可能是工具的 Bug。因此如果没有效果的话不要在这里浪费太多时间，因为这并不是关键点，不要在这浪费太多时间。

3、**结构体及指针**（这部分内容具体请参考 C 语言的教材）

```
struct MyStruct
{
int f1;
int f2;

}//最常用的定义结构体的方式。

typedef int MYInt; //使用 typedef 定义类型别名

typedef MyStruct JSJ321Struct; //结构体别名的声明

typedef MyStruct *PMyStruct;//PMyStruct 是 MyStruct 指针的声明。

typedef struct MyStruct
{
int f1;
int f2;
```

} JSJ321Struct;//把 MyStruct 的定义和定义 MyStruct 的别名为 JSJ321Struct 结合到了一起。

typedef struct MyStruct
{
int f1;
int f2;

} *PMyStruct;//把 MyStruct 的定义和定义 MyStruct 的指针别名 PMyStruct 的定义结合到了一起。

typedef struct MyStruct
{
int f1;
int f2;
} JSJ321Struct ,*PMyStruct;// 把 MyStruct 的定义和定义 MyStruct 的别名为 JSJ321Struct 以及定义 MyStruct 的指针别名 PMyStruct 的定义结合到了一起。

4、最普通的结构体声明。

MyStruct s1;

5、给结构体所有字段置零

复习一下 C 语言中 memset 以及结构体的应用。C 语言中声明一个结构体变量之后，在使用这个变量之前要首先用 memset 来把各个位清零。使用 C 语言内置的函数 memset 来完成：

memset(&s1,sizeof(s1),0);

在这里我们使用 ZeroMemory“函数”，ZeroMemory 其实并不是函数，看一下 ZeroMemory 的定义：

#define ZeroMemory RtlZeroMemory ； ZeroMemory 其实编译的时候会宏展开为 RtlZeroMemory。

#define RtlZeroMemory(d,l) RtlFillMemory((d),(l),0)// RtlZeroMemory 又是 RtlFillMemory 的一个宏定义，而且是带参数的宏定义。

#define RtlFillMemory(d,l,f) memset((d), (f), (l))
// RtlFillMemory 其实是 memset 的宏定义

ZeroMemory 其实就是 memset，纸老虎而已，不是一个全新的函数。

6、打开文件对话框的使用

OPENFILENAME ofn;
char szFile[MAX_PATH];

```
ZeroMemory(&ofn,sizeof(ofn));
ofn.lStructSize = sizeof(ofn);
ofn.lpstrFile = szFile;
 ofn.lpstrFile[0] = TEXT('\0');
ofn.nMaxFile = sizeof(szFile);
ofn.lpstrFilter = TEXT("ALL\0*.*\0Text\0*.TXT\0");
ofn.nFilterIndex = 1;
ofn.lpstrFileTitle = NULL;
ofn.nMaxFileTitle = 0;
ofn.lpstrInitialDir = NULL;
ofn.Flags = OFN_EXPLORER |OFN_PATHMUSTEXIST | OFN_FILEMUSTEXIST;
if (GetOpenFileName(&ofn))
{
MessageBox(NULL,szFile,"",MB_OK);
}
```

下面将对这段代码作解释：

（1）、ZeroMemory 的含义见上边

（2）char szFile[MAX_PATH];//定义一个长度为 MAX_PATH 的 char 数组。

MAX_PATH 是系统定义的一个宏，260。Windows 操作系统文件路径的最大长度。

（3）ofn.lpstrFile = szFile;

//szFile 是一个字符数组，那么 szFile 这个名字就代表这个字符数组的首地址

这是一个指针的应用，把接收选择文件名的数组的地址传递到 GetOpenFileName 函数中。

（4）ofn.lpstrFile[0] = TEXT('\0');效果等价于 ZeroMemory(szFile,sizeof(szFile)/sizeof(char));

（5）**文件类型过滤器**

*.*就表示所有文件

*.txt 就表示所有文本文件

abc*.*就表示所有以 abc 开头的文件

abc*.txt 就表示所有以 abc 开头的文本文件（txt）

ofn.lpstrFilter = TEXT("ALL\0*.*\0Text\0*.TXT\0");用“\0”分割出几个段，每两段是一组过滤器，每组

的第一个段代表过滤器的显示的值，而第二段表示真正的过滤器

（6）ofn.nFilterIndex = 2;默认选择第几个过滤器，是 1 开始的，不是 0.这也是试验出来的。

（7）windows 的一个惯例：结构体的指针类型别名一般是 LP+结构体的名字

GetOpenFileName(&ofn)而不是 GetOpenFileName(ofn)，因为 GetOpenFileName 函数要的是 OPENFILENAME 的指针

（8）只要用户还没关闭对话框，那么 GetOpenFileName 函数是停住的，同步模式的 PlaySound 以及 getchar、scanf 是一样的。

用户关闭对话框，GetOpenFileName 函数返回，返回值是 BOOL，等于 TRUE 表示用户选择了文件，如果等于 FALSE 就表示用户选择了【取消】按钮

四

注意：有同学反应添加控件以后程序编译没报错，但是运行没有反应，请看此贴：《【常见问题】程序编译不报错，为什么点击运行无反应？ 》http://www.rupeng.com/forum/thread-601-1-3.html

1、GetOpenFileName

希望同学们不要只记我教的结果，而是要跟着我的思路进行思考，看解决问题的方式。否则就又变成“我教同学们学”的填鸭式教育了。

(2)课上练习：打开对话框，用户选择一个音乐文件，然后用 PlaySound 播放。

（3）打开多个文件：

增加 OFN_ALLOWMULTISELECT 选项后测试一下。发现如果选择一个文件后显示的还是这个文件的路径，可是如果选择多个文件显示的就只有目录的路径，怎么回事？

调试一下，看看 szFile 在内存中的样子。调试的是发现内存中的样子是目录的路径然后加上各个文件名，中间用“\0”分割，现在明白为啥值显示目录的路径了吗？同学们举手回答。

大家明白这一点就可以了，有兴趣的可以课后来做多文件选择的处理。做播放器的时候经常需要这个功能，也就是播放列表。有一篇文章可以参考一下：http://xxkkff.blog.51cto.com/162016/26222。

2、GetSaveFileName

保存文件对话框

只要把上边函数换成 GetSaveFileName。OFN 本来是 OpenFileName 的简写，它 SaveFileName 也跑来凑热闹，吼吼。如果需要保存文件覆盖提示怎么办呢？看看 OFN 有选项吗？OFN_OVERWRITEPROMPT

3、对话框程序

（1）创建一个对话框程序并运行

（2）先打开 dlg 文件然后点击主菜单的【工具】→【资源编辑器】，也可以在文件上点右键选择【打开方式】→【ResourceEditor】。

外行看热闹、内行看门道，思考一下第一个打开方式的原理，很多软件都这么搞，支持外接软件

在资源编辑器中编辑对话框界面并且拖放控件上去

（3）什么叫做“对话框程序”，和以前学的“窗口程序”有什么区别吗？

“对话框程序”是一种特殊的“窗口程序”，与“窗口程序”比起来学习难度比较小，最重要的是**可以使用可视化的对话框编辑器**。对话框程序能够完成大部分功能，但是有小部分功能还是要用“窗口程序”，**对话框程序能完成的功能是“窗口程序”的一个子集，学习“对话框程序”中的知识点可以完全应用到“窗口程序”上去**。让大家**快乐学习，快速做出东西**，所以咱们先学习“对话框程序”的开发。

（4）常用控件介绍。RichEdit、UDC 控件要是使用的话要加代码，后面会讲。

4、嵌入式资源

（1）资源的概念。

word 里边插入图片有：嵌入和链接两种方式。

web 里边是连接方式。

为什么要有资源。把图片和 EXE 文件放到一起，一个文件夹底下。我更喜欢的方式，只要一个 EXE 文件，你把用到的所有图片、歌曲音乐、动画 EXE 里边去。这些东西就是资源。资源和代码区分开。EXE 文件两部分：代码和资源。

（2）惯例：所有资源的名字都是 ID 开头，并且大写，ID 后边一般跟着一个资源类型的大

写字母，对话框 D、Bmp 图片 B、图标 I。接着是下划线，最后是真正的名称。

5、对话框中图片、动画资源的应用（1）BitMap：在资源中加入 BITMAP 类型的资源，然后拖一个 IMAGE 控件过来，并且修改 IMAGE 控件的 Type 属性为 BitMap，并且给资源一个名字（一般以 IDB_开头，并且大写，这是规范，不是必须）并且在控件的 Image 属性中选择刚才添加的 BITMAP 资源。虽然可以直接选择，但是还是建议直接把图片拷贝到工程目录下，因为。。。jpg 的格式怎么转换为 bmp 格式的呢？用画图工具（mspaint）。（2）动画：在资源中加入 AVI 类型的资源，然后拖一个 Animation 控件过来，在 AVIClip 属性中选择刚才添加的 AVI 资源，并且给资源一个名字（一般以 IDV_开头，并且大写，这是规范，不是必须）。如果需要自动播放，则修改 AutoPlay 属性为 True。AVI 动画资源非常少，但是 GIF 动画非常多，可以用工具将 GIF 转换为 AVI 格式。注意很多电影的格式也是 AVI 后缀，但是那并不是咱们这里说的 AVI 动画。AVI 格式是一个大杂烩。GIF 转 AVI 的工具：http://www.onlinedown.net/soft/7689.htm

注意：有同学反应"程序加入动画控件以后程序就运行不起来了，可以构建，但是点击运行却没反应"，请参考下面的文章：

程序编译没错，但运行无反应?

http://www.rupeng.com/forum/thread-601-1-3.html

这个软件有一个 Bug，使用的时候必须在 c 盘下创建一个名字为"aa"的文件夹

课下作业：自己发挥，比如给女友做一个贺卡（下节课讲教大家怎么加音乐，当然是用 PlaySound 函数）

6、菜单资源

制作好菜单资源后，在对话框的 Menu 属性中选择刚才制作好的对话框。

7、只有控件、菜单，可以点击控件、菜单以后的动作怎么做呢？就要写代码了。

同学提问：

（1）BMP 的图片不是很大吗，这么插入资源会不会使得最后编译的程序体积很大

答：对，确实会使得文件比较大，但是一般软件里嵌入的图片都是起到装饰作用，所以东西不会太多。而且你生成的 EXE 文件还可以压缩。

（2）IDB_TEST1 BITMAP DISCARDABLE "D:/My Documents/001.BMP"

刚才讲的是链接式的还是嵌入式的啊，为什么 dialogs 的代码里面会有这个 bmp 文件的路径啊 不是嵌入进去了吗？如果这个时候我吧 001.bmp 挪走还会显示吗？

答：这个路径是编译时用到的，所以在编译的时候如果把图片挪走就会编译失败。但是一旦编译成功就不需要 001.bmp 了。建议把所有资源都放到工程目录下，这样可以随便移动工程。

五

1、在编写控制台程序的时候一切流程都是有先后关系、并行的，而且所有函数都是由我们来调用的，比如下面的实例性代码：

```
printf("确定请输入 y，取消输入 n");
char c = getchar();
if(c=='y')
{
///
}
else if(c=='n')
{
///
}
```

我们可以用 getchar 来等待用户输入一个值。但是到了 Windows 编程中就不一样了，同一时刻用户即可能点击【OK】按钮，又可能点击【Cancel】按钮，又可能在文本框中输入几个字，还可能在窗口上双击几下，这样就无法同时等待用户的这些动作。为了解决这个问题，Windows 引入了消息机制（也可以叫做回调机制或者事件机制）。在程序启动的时候把函数 func1 要响应【OK】按钮 1 的点击动作、函数 func2 要响应【Cancel】按钮的点击动作、函数 func3 要响应窗口的双击动作等等这些信息告诉 Windows，然后当用户执行相应操作的时候 Windows 就会来主动调用你注册的函数，主动通知你。不再是程序调用操作系统的函数，而是操作系统反过来调用你的函数。Don't call me ,I'll call you!（也被人称为“好莱坞法则”）。

2、关于上面的这个问题要慢慢来理解，下面就来通过第一个例子来初步理解这个 Don't call me ,I'll call you!

创建一个对话框程序，然后来分析代码。看 Main_OnCommand 方法，初探 windows 的消息机制。

```
void Main_OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify)
{
```

```
switch(id)
{
case IDC_OK:
MessageBox(hwnd,"You click OK!","Test003",MB_OK);
EndDialog(hwnd, id);
break;
case IDC_CANCEL:
MessageBox(hwnd,"You click Cancel!","Test003",MB_OK);
EndDialog(hwnd, id);
break;
default:break;
}
}
```

按钮被按下的时候 Main_OnCommand 方法被调用 ,hwnd 是对话框句柄(什么是句柄后面讲，通俗的说就是通过它能够操纵对话框)，id 是控件的 id，后两个参数暂时不关心。

Main_OnCommand 方法中根据 id，也就是被点击按钮的名字来决定不同的动作，EndDialog 用来关闭对话框。

3、定制自己的对话框，向世界问好

首先打开资源编辑器并且打开对话框 IDD_MAIN，然后删除对话框上的两个按钮和标签，同时删除 main.c 中的两个 Case 语句变成：

```
void Main_OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify)
{
switch(id)
{
default:break;
}
}
```

然后重新添加一个自己的按钮，在属性中修改 Name 属性为 IDC_BTNHELLO。

控件名字的潜规则：所有控件的 Name 都以“IDC_”开头，然后后边跟着控件类型的简称 (按钮简称 BTN、文本框简称 EDT 等等),最后才是控件的真正的名字。修改按钮的 Caption 属性(也就是按钮上显示的文字) 为“问好”。

在代码中怎么得到 IDC_BTNHELLO 呢？刚才被删掉的 IDC_OK 是什么东东呢？回忆配置 ResEd 的时候配置的“名称输出格式”和“默认输出文件名”以及“保存时自动输出”。每次保存对话框的时候 ResEd 都会帮我们把控件的名字输出到 rsrc.inc 文件中，打开工程文件夹下的 rsrc.inc，内容如下：

```
#define IDC_BTNHELLO
1001
```

Dialog 编辑器会自动递增 id 的取值。然后生成 rsrc.inc，其实就是 h 头文件，取这些定义的时候要先 include 这个 inc 文件。可以看到 rsrc.inc 文件中就是这些控件名字的定义，使用的时候只要 include 这些文件就可以。“rsrc.inc”和头文件一样。因此首先在 main.c 中添加“#include "rsrc.inc""

编辑 Main_OnCommand 方法：

```
void Main_OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify)
{
switch(id)
{
case IDC_BTNHELLO:
MessageBox(NULL,TEXT("世界你好"),TEXT("问好"),MB_OK);
break;
default:break;
}
}
```

运行程序，点击“问好”按钮，咱们第一个响应按钮动作的程序就做好了。

4、自己动手写计算器 1.0 版

（1）、得到控件中用户输入的文本

```
GetDlgItemText(hwnd,IDC_EDTNAME,str,sizeof(str));
```

第一个参数是对话框的句柄；第二参数就是控件的 id（name）；第三个参数就是字符串数组的指针；

疑问:为什么不能像 MessageBox 一样把字符串做为返回值返回呢？？？

这就涉及到 C 语言中函数返回指针的问题了

```
int i=20;
char cStr[3];
itoa(i,cStr,10);
char* f1()
{
return "adsfadsfasd";
}
```

在函数内部返回函数内构建的指针有可能出现数据混乱.

当 f1 函数在执行的时候，这段内存是被占用的，一旦函数返回，那么内存就标记为可以被其他人、函数占用。

如果你把这段内存中的指针返回了，那么一旦其他地方用了这段内存，那么你引用的就是错误的数据了。

用 GetWindowText、GetDlgItemText 的时候为什么要传数组名呢？复习：数组名就是指针，函数只有得到指针，才能直接操作数组中的数据。

（2）c 语言中字符串连接：strcat(name,"你好");

但是在编写 windows 程序的时候最好使用 lstrcat 代替 strcat。

lstrcat(name,"你好");

同样代替的有：strlen→lstrlen；strcmp→lstrcmp

（3）向用户问好

```
TCHAR name[256];
GetDlgItemText(hwnd,IDC_EDTNAME,name,sizeof(name)/sizeof(TCHAR));
lstrcat(name,"你好");

MessageBox(NULL,name,TEXT("问好"),MB_OK);
```

（4）

C 语言里边字符串转换为数字：atoi：ascii to int

（5）

编写 windows 程序的时候最好用 TCHAR 来代替 char，可以避免中文的问题

```
case IDC_BTNHELLO:
TCHAR name[256];
GetDlgItemText(hwnd,IDC_EDTNAME,name,sizeof(name)/sizeof(TCHAR));
int i = atoi(name);
int j=i*2;
TCHAR result[256];
itoa(j,result,10);

MessageBox(NULL,result,TEXT("问好"),MB_OK);
```

（6）Get、Set：配对的，设置对话框的值用 SetDlgItemText：

代码：

```
case IDC_BTNADD:
//取第一个文本框的字符串，然后得到整数表示

// 取第二个文本框的字符串，然后得到整数表示

//计算两个整数的和

//把和重新转换成字符串，然后 Messagebox 出来。
TCHAR cNumber1[256];
GetDlgItemText(hwnd,IDC_EDTNUM1,cNumber1,sizeof(cNumber1)/sizeof(TCHAR));
int n1 = atoi(cNumber1);
TCHAR cNumber2[256];
GetDlgItemText(hwnd,IDC_EDTNUM2,cNumber2,sizeof(cNumber2)/sizeof(TCHAR));
int n2 = atoi(cNumber2);
int n3 = n1+n2;
TCHAR cResult[256];
itoa(n3,cResult,10);
SetDlgItemText(hwnd,IDC_EDTRESULT, cResult);
break;
```

（7）代码中有 UINT、TCHAR、LRESULT、HWND 之类的新的数据类型，其实它们只是一些类型的别名而已，可以通过宏定义看出来。但是考虑到可移植性，尽量不要使用它们的真实类

型

（8）思考：这个计算两个数的和程序有什么缺陷？没有阻止用户输入非数字

（9）作业：做一个面积计算器，用户输入半径，在另外一个文本框中显示面积。

5、得到系统中有哪些逻辑驱动器

DWORD GetLogicalDrives(VOID);

返回值的二进制位标志着存在哪些驱动器。其中 ,位 0 设为 1 表示驱动器 A 存在于系统中；位 1 设为 1 表示存在 B 驱动器；以次类推。

比如：

00001100：有 C 盘，D 盘

000011100：c、d、e

1101：A（软驱）、C、D

（1）写程序中的错误排除

"dwDrives"was not declared in the scope

declare:声明；scope:范围

"dwDrives"没有被声明在这个范围内。

（2）以二进制显示 GetLogicalDrives 的返回值：

```
stdlib.h
DWORD ds = GetLogicalDrives();
char str[33];
itoa(ds,str,2);
MessageBox(NULL,str,"",MB_OK);
```

DWORD 是什么类型？在 DWORD 上点击右键，选择“转到 DWORD 的定义”，其实 DWORD 是“unsigned long”。因此 str 定义成 33 位（还有最后一位的“\0”）。

（3）课后习题：判断是否存在 D 盘。

（4）课后习题：依次显示系统中所有的盘符。比如显示出“CDEF”。

（5）课后习题：显示系统中有多少个逻辑驱动器。

这三道课后作业都是在锻炼位运算。一定要重视，不要一位没啥意思，很多公司的笔试面试都会考查这一点，这也是继续深入研究 C 和其他语言的基础。

七

1、对 diamondboy 作业的点评。思路非常好，而且使用了表驱动算法。

2、位运算的基础：

| ：或。1|0=1;1|1=1;0|0=0;

& ：与。1&0=0;0&0=0;1&1=1;

^ ：非。^1=0,^0=1;

3、位运算例子：

101
110 |
====
1 1 1
001
101|
===
1 0 1

4、

101
110 &
====
1 0 0
1100
1000 &
====
1 0
00
0100
1000&
====
0
0
0 0

要判断一个数的二进制的第 n 位是否为 1 的方法就是：把一个 n 位为 1，其他位为零的数

与待判断的书进行与运算，只要结果不为 0，就说明 n 为为 1，否则为 0.让其他为为 0 就是利用与运算的特性来屏蔽其他位的干扰，0 最具侵略性（与运算）；n 为为 1 则是为了验证那一位是否真的为 1.

1000=8

因此判断是否存在 D 盘这道题的位运算的解法。

参考答案：

```
DWORD ds = GetLogicalDrives();
//8 是 1000
if(ds&8)
{
MessageBox(NULL,"存在 D 盘","",MB_OK);
}
else
{
MessageBox(NULL,"不存在 D 盘","",MB_OK);
}
```

5、显示系统中有多少个逻辑驱动器。

分析规律：

1100。数有多少个 1 就可以了。

右移一位：

110

再右移一位：11

再右移一位：1

再右移一位：0

1110。数有多少个 1 就可以了。

右移一位：

111

再右移一位：11

再右移一位：1

再右移一位：0

不断的右移，只要某一位为 1，那么早晚会移到末尾一位，只要在末尾等着他就可以了。

不断右移位，然后判断最后一位是否是 1，如果是的话就计数器加 1，知道移位变成 0 为止。

```
//得到数字 n 的二进制中 1 的个数

//写一个单独的函数，这样就达到了抽象的目的，不要把所有代码挤在一个函数中
int Get1Count(DWORD n)
{
int counter=0;
while(n>0)
{
//判断末位是否是 1
if(n&1)
{
counter++;
}
//向右移位，直至为 0
n=n>>1;
}
return counter;
}
```

“依次显示系统中所有的盘符。”这道题请大家课后继续思考。**光看没用，要是看一看就管用的话去看看 NBA 比赛我就能成为灌篮高手，要是看一看就管用的话去看看菲尔普斯游泳我就能成水中飞鱼，要经常练、实际的把代码自己写出来。**

今天内容：作业点评，给女友的音乐贺卡（exe 嵌入音乐，涉及到 c 编程中全局变量的处理），“自己动手写计算器 2.0 版”。如果时间允许会讲“自己动手写小时钟”，涉及到的知识点是：定时器的使用，涉及到的 C 知识就是函数指针（回调函数）的应用。

吃糖葫芦：既有红果、又有橘子瓣，数有多少个红果？我是个盲人，从一头开始吃，遇到酸味的就是红果。

jason 作业点评：

依次显示系统中所有的盘符

```
case IDC_BTNLIST:
TCHAR str[256];
DWORD d2=GetLogicalDrives();
Getlist(str,d2);
SetDlgItemText(hwnd,IDC_EDTLIST,str);
break;

//函数实现部分
void Getlist(TCHAR *p,DWORD n)
{
int a=65;
//'A'的 ASCII 值
int i=0;
while(n>0)
{
if(n&1)
{
*p=char(a);
p++;
}
n=n>>1;
a++;
*p=0;
}
}
```

jason 写的非常好，改进意见把“int a=65”替换成“int a= 'A';”，这样就不用写死 65 这个“神秘数

字"了。

我的参考答案，思路和 jason 的非常类似：

```
DWORD ds = GetLogicalDrives();
int counter=0;
while(ds>0)
{
if(ds&1)
{
char ch = 'A'+counter;
char chs[2] = {ch,'\0'};
MessageBox(NULL,chs,"",MB_OK);
}
counter++;
ds=ds>>1;
}
```

怎么播放嵌入式的音乐。给女友的音乐贺卡

插入 WAVE 类型的资源，给资源一个名字，比如 IDW_MUSIC 然后在 Main_OnInitDialog 方法里调用：

```
PlaySound(TEXT("IDW_MUSIC"),aInstance,SND_RESOURCE|SND_ASYNC| SND_LOOP);
```

其中 SND_RESOURCE 掩码位表示音乐文件资源是嵌入式资源

在 mian.c 的头部加上全局变量声明：

```
HINSTANCE aInstance;
```

在 main.h 中加入

```
extern HINSTANCE aInstance;
```

在工程的 WinMain 所在的代码中加入：

```
aInstance = hInstance;
```

在.c 文件中声明全局变量，在.h 中使用 extern 来"导出"变量。

大的项目中不可能只有一个文件，因此这是一个比较重要的地方。

一个 c 文件里不能直接调用另一个 C 文件里的全局变量

我简单提醒一下，你再对照课本。比如 A.c 里定义了一个全局便来那个，如果 B.c 要调用，

首先要在 A.h 里 extern 这个变量，然后在 B.c 中 include A.h

全局变量也成为外部变量，他是在函数外部定义的变量，不属于哪个函数，他属于整个源程序文件，其作用域是整个源程序。

晕了

原来的控制台程序就 1 个文件，没发现那么多问题，现在的 Dialog 程序一弄都是一堆，大批的问题就来了

2、得到 HINSTANCE 的两种方式

（1）像上面讲的一样：可以在 WinMain 中使用全局变量保存实例句柄，在 main.c 中声明：

```
HINSTANCE aInstance;
```

在 main.h 中声明：

```
extern HINSTANCE aInstance;
```

然后在 WinMain 中：

```
aInstance = hInstance;
```

（2）用 GetWindowLong 从窗口句柄取得执行实体句柄：

```
HINSTANCE ai = (HINSTANCE)GetWindowLong(hwnd,GWL_HINSTANCE);
```

做任何事情都不可能只有一条路，不要你学过一条路，看别人走另外一条路你就糊涂了，作选择的权利是人最应该珍惜的权利，应该根据实际情况选择不同的路。就像是不是要考研之类的选择一样，没人比你自己更清楚应该怎么办。

下面就来实现“自己动手写计算器 2.0 版”，允许选择运算符号。

ComboBox 的操作

（1）添加项：

填入 Combo 最简单的方法是借助 CB_ADDSTRING 消息：

```
SendDlgItemMessage (hwnd,IDC_CBO1, CB_ADDSTRING, 0, (LPARAM) szString) ;
```

(2) 确定目前选项的索引 :iIndex = SendDlgItemMessage (hwnd,IDC_CBO1, CB_GETCURSEL, 0, 0) ;

如果没有项目被选中，那么从函数中传回的 iIndex 值为 CB_ERR。额外知识，不讲，自学（不想变成填鸭式教育，课上只讲思想性的东西，会给大家学习资料，希望大家培养自学能力）

（3）删除项：

在指定索引值的同时使用 LB_DELETESTRING 参数，这就可以从清单方块中删除字符串：

SendDlgItemMessage (hwnd,IDC_CBO1, LB_DELETESTRING, iIndex, 0) ;

（4）取得有多少项：

iCount = SendDlgItemMessage (hwnd,IDC_CBO1, LB_GETCOUNT, 0, 0) ;(5) 选定某一项

SendDlgItemMessage (hwnd,IDC_CBO1, LB_SETCURSEL, iIndex, 0) ;将 iParam 设定为-1 则取消所有选择

实现四则运算器：

（1）首先在对话框初始化的时候向 ComboBox 加入内容：

```
SendDlgItemMessage(hwnd,IDC_CBOOPERATOR,CB_ADDSTRING,0,(LPARAM)TEXT("+"));
SendDlgItemMessage(hwnd,IDC_CBOOPERATOR,CB_ADDSTRING,0,(LPARAM)TEXT("-"));
SendDlgItemMessage(hwnd,IDC_CBOOPERATOR,CB_ADDSTRING,0,(LPARAM)TEXT("*"));
SendDlgItemMessage(hwnd,IDC_CBOOPERATOR,CB_ADDSTRING,0,(LPARAM)TEXT("/"));
```

（2）Calc 函数的实现：

```
void Calc(HWND hwnd)
{
TCHAR cNumber1[256],cNumber2[256];
GetDlgItemText(hwnd,IDC_EDTNUM1,cNumber1,sizeof(cNumber1)/sizeof(TCHAR));
GetDlgItemText(hwnd,IDC_EDTNUM2,cNumber2,sizeof(cNumber2)/sizeof(TCHAR));
int iIndex = SendDlgItemMessage(hwnd,IDC_CBOOPERATOR,CB_GETCURSEL,0,0);
int i1 = atoi(cNumber1);
int i2 = atoi(cNumber2);
int iResult;
BOOL bError = FALSE;
switch(iIndex)
{
case 0:
iResult = i1+i2;
break;
case 1:
```

```
iResult = i1-i2;
break;
case 2:
iResult = i1*i2;
break;
case 3:
if(0==i2)
{
MessageBox(hwnd,TEXT("被除数不能是 0"),TEXT("错误"),MB_OK|MB_ICONERROR);
bError = TRUE;
}
else
{
iResult = i1/i2;
}
break;
default:
MessageBox(hwnd,TEXT("操作符选择错误"),TEXT("错误"),MB_OK|MB_ICONERROR);
bError = TRUE;
break;
}
if(FALSE==bError)
{
TCHAR cResult[256];
itoa(iResult,cResult,10);
SetDlgItemText(hwnd,IDC_EDTRESULT,cResult);
}
}
```

课后作业：实现实际值到二进制的转换

六

接下来我们要实现小时钟。涉及到的主要知识点是定时器的使用，涉及到的 C 知识就是函数指针（回调函数）

使用定时器

回调就是不是你调别人，而是别人调你。

让 Windows 直接将定时器消息发送给您程序的另一个函数。

我们把以下的 callback 函数称为 TimerProc（您能够选择与其它一些用语不会发生冲突的任何名称），它只处理 WM_TIMER 消息：

```
VOID CALLBACK TimerProc (
HWND hwnd, UINT message, UINT iTimerID, DWORD dwTime);
SetTimer (hwnd, iTimerID, iMsecInterval, TimerProc) ;
```

定时器还有其他两种用法，大家可以参考《Windows》程序设计这本书（不想变成填鸭式教育，课上只讲思想性的东西，会给大家学习资料，希望大家培养自学能力）

19、得到当前日期：

```
void GetLocalTime(LPSYSTEMTIME);
```

课上练习：

```
SYSTEMTIME stLocal;
char chBuf[256];
GetLocalTime(&stLocal);

//显示时间的间隔。

wsprintf(chBuf,TEXT("%u/%u/%u %u:%u:%u:%u 周%d\r\n"),
stLocal.wYear, stLocal.wMonth, stLocal.wDay,
stLocal.wHour, stLocal.wMinute, stLocal.wSecond,
stLocal.wMilliseconds,stLocal.wDayOfWeek);
MessageBox(NULL,chBuf,"",MB_OK);
SetTimer(hwnd,1,1000,TimerProc);
void CALLBACK TimerProc (HWND hwnd, UINT message, UINT iTimerID, DWORD dwTime)
{
```

```
SYSTEMTIME time;
GetLocalTime(&time);
TCHAR strTime[256];
//%后面的 0 表示不够的位数补零，5 是总位数，就这样

wsprintf(strTime,"%04d-%02d-%02d %02d:%02d:%02d",time.wYear,time.wMonth,time.wDay,time.wHour,time.w
Minute,time.wSecond);
SetDlgItemText(hwnd,IDC_EDTTIME,strTime);
}
```

后续的课程会讲得到空余内存、硬盘信息（总大小、空余、类型）、得到当前登录用户名、得到操作系统类型→自己动手写 windows 优化大师 1.0 版

菜单、文件处理、windows 控件的操作→自己动手写记事本

有人说了“干嘛知道这么多？使用 C#、VB、Delphi 之类的工具拖拽控件，像搭积木一样就可以搭建出一个程序来”。确实如此，但是这样的程序员恐怕竞争力不强。只有深入了解内部原理，才能让自己对技术融会贯通，也才能让程序员之路走得更稳健、更长久。永远记住价值规律：越容易的东西越便宜。

（6）句柄（Handle，注意这个概念后面要经常用到，需要仔细理解）

句柄是 WONDOWS 用来标识被应用程序所建立或使用的对象的唯一整数，WINDOWS 使用各种各样的句柄标识诸如应用程序实例、窗口、图片、图标等等。WINDOWS 句柄有点象 C 语言中的文件句柄。在运行时唯一确定一个对象！

为什么要搞句柄这个东西呢？比如我们要加载一个图片，这时就要调用系统的加载图片的 API 函数，API 函数会将图片加载到内存的一个区域，然后通过什么方式将这个区域的地址告诉调用者呢？有的同学会说：直接将内存区域的指针返回不就行了吗？对于图片、文件等对象来说，分配到堆！而不是栈！存在下面的问题：这些对象并不是立即被加载到内存的，只有访问到的时候才加载到内存中；为了高效的利用内存，操作系统可能会移动数据在内存中

的位置。因此不能直接返回内存的指针，操作系统会为这个图片、文件对象分配一个唯一的标识号，然后就可以按照这个标识号进行资源的访问了。这个标识号就是句柄，句柄其实就是一个对象的编号，数据类型其实就是整数，只不过用 typedef 搞了一个别名而已。

怎么得到控件句柄呢？根据 id 得到。

消息机制、回调函数和函数指针（现在明白 C 程序设计中的函数指针是做什么用的了吧）

控件句柄和控件 id 的区别：控件句柄在每次运行的时候都是不一样的，而控件 id 则是确定的；就像 fopen 函数返回的句柄一样，每次运行的句柄都不是一样的，但是文件名是确定的。

句柄一般在运行时动态根据 id 算出来。

hFile = fopen("c:/a.txt")

5、设置、读取控件的文字（文本框、标签、CheckBox、Button）

设置：

方法 1：SetDlgItemText(hwnd,IDC_EDTNAME,TEXT("世界你好"));

方法 2： HWND hwndEdtName = GetDlgItem(hwnd,IDC_EDTNAME);

SetWindowText(hwndEdtName,TEXT("世界你好"));

读取：

方法 1：

```
HWND hwndEdtName = GetDlgItem(hwnd,IDC_EDTNAME);
TCHAR str[256];
ZeroMemory(str,sizeof(str));
GetWindowText(hwndEdtName,str,sizeof(str));
MessageBox(hwnd,str,"",MB_OK);
```

方法 2：

```
TCHAR str[256];
ZeroMemory(str,sizeof(str));
GetDlgItemText(hwnd,IDC_EDTNAME,str,sizeof(str));
MessageBox(hwnd,str,"",MB_OK);
```

SetDlgItemText 和 GetDlgItem、SetWindowText 的关系

我在家里是老大，所以别人要打我的话有两种方式：

1、直接找到我，然后打我。这就是直接抓住我的把柄 SetWindowText

2、找到我爸，然后找我爸爸的 id=1 的儿子，也就是我，然后再打我。这就是通过我把的把柄和我的 id=1 来定义我

SendDlgItemMessage 和 SendMessage 一个道理。

3、可以使用 GetDlgCtrlID 来根据控件句柄得到控件的 ID，也可以使用 GetDlgItem (hwndParent, id)来根据控件的 id 得到控件的句柄。函数中的「Dlg」部分指的是对话框，但实际上这是一个通用的函数。

七

1、修改文字的颜色呢？这块要涉及到比较复杂的东西，不过其实可以先用画图板画一个文字图片，然后放上去呀，呵呵。变通嘛

2、下面几个函数只是给大家文档，具体怎么用则需要自己试验。这也是训练大家自学能力和解决问题能力的一步，不能永远让老师"填鸭式教育"。

3、得到驱动器的簇信息、剩余空间大小、总大小

函数原型：

```
BOOL GetDiskFreeSpaceExA(
LPCSTR lpDirectoryName,
PULARGE_INTEGER lpFreeBytesAvailableToCaller,
PULARGE_INTEGER lpTotalNumberOfBytes,
PULARGE_INTEGER lpTotalNumberOfFreeBytes
);
```

lpDirectoryName 是驱动器的名称。

lpFreeBytesAvailableToCaller 是用户可用的磁盘空间。

lpTotalNumberOfBytes 是磁盘总共的空间。

lpTotalNumberOfFreeBytes 是磁盘空闲的空间。以上都是字节为单位。

lpFreeBytesAvailableToCaller 不一定等于 lpTotalNumberOfFreeBytes。操作系统里边可以设置某个用户可以用的最大磁盘，比如说磁盘还有 100G 可用，但是我限制用户 yzk 最多能用 20G 硬盘，因此如果用 yzk 登录的话只有不到 20G 的可用空间。因此 lpTotalNumberOfFreeBytes>=lpFreeBytesAvailableToCaller。

PULARGE_INTEGER 是 64 位的整数，也就是"大整数 BigInteger"，因为 int 是 32 位的，如果用 int 表示磁盘空间的话最多能表示 2 的 32 次方也就是 2G。现在的磁盘都超过了 2G 了，所以要用 64 位的大整数来表示。

（1）示例代码：

```
int i;
        scanf("%d",&i)
ULARGE_INTEGER nFreeBytesAvailable;
ULARGE_INTEGER nTotalNumberOfBytes;
ULARGE_INTEGER nTotalNumberOfFreeBytes;
if (GetDiskFreeSpaceEx("C:",
    &nFreeBytesAvailable,
    &nTotalNumberOfBytes,
   &nTotalNumberOfFreeBytes))
  {
    TCHAR chBuf[256];
    wsprintf(chBuf,"当前用户可用 ： %I64d,总大小 ： %I64d,当前可用空
间 ： %I64d\r\n",
       nFreeBytesAvailable,
       nTotalNumberOfBytes,
       nTotalNumberOfFreeBytes);
    MessageBox(NULL,chBuf,"",MB_OK);
  }
  else
  {
       MessageBox(NULL,"获取失败","",MB_OK|MB_ICONASTERISK);
  }
```

注意这里 wsprintf 的用法，wsprintf 和 printf 类似，不过 wsprintf 是将结果打印到字符串中，而不是屏幕中。“%I64d”和 printf 中的“%d”一样，它表示此处是 64 位大整数的占位符。

我们在向着 Windows 优化大师一步步前进。耶！！！

（2）课后作业：

上面代码的显示效果很难看，请以 MB 或者 GB 的形式显示结果。而且如果结果小于 1G 则以 MB 为单位显示，否则以 GB 为单位显示。

4、得到磁盘的类型（只介绍，自己课下练习）

UINT GetDriveType(

LPCTSTR lpRootPathName // root directory

);

函数功能：判断磁盘类型

参数说明

lpRootPathName 包含了根目录路径的字符串指针

返回值：

DRIVE_UNKNOWN 未知的磁盘类型

DRIVE_NO_ROOT_DIR 说明 lpRootPathName 是无效的

DRIVE_REMOVABLE 可移动磁盘

DRIVE_FIXED 固定磁盘

DRIVE_REMOTE 网络磁盘

DRIVE_CDROM 光驱

DRIVE_RAMDISK 为 RAM

（1）示例代码：

```
UINT t = GetDriveType("c:");
    switch (t)
    {
    case DRIVE_UNKNOWN:
        MessageBox(NULL,TEXT("未知磁盘类型"),"",MB_OK);
        break;
case DRIVE_NO_ROOT_DIR:
        MessageBox(NULL,TEXT("磁盘名无效"),"",MB_OK);
        break;
case DRIVE_REMOVABLE:
        MessageBox(NULL,TEXT("可移动磁盘"),"",MB_OK);
```

```
            break;
        case DRIVE_FIXED:
            MessageBox(NULL,TEXT("固定磁盘"),"",MB_OK);
            break;
        case DRIVE_REMOTE:
            MessageBox(NULL,TEXT("网络磁盘"),"",MB_OK);
            break;
        case DRIVE_CDROM:
            MessageBox(NULL,TEXT("光驱"),"",MB_OK);
            break;
        case DRIVE_RAMDISK:
            MessageBox(NULL,TEXT("RAM"),"",MB_OK);
            break;
        default:
            MessageBox(NULL,TEXT("GetDriveType 的返回值非法"),"",MB_OK);
    }
```

八

1、如何修改文字的颜色呢？这块要涉及到比较复杂的东西，不过其实可以先用画图板画一个文字图片，然后放上去呀，呵呵。变通嘛

2、下面几个函数只是给大家文档，具体怎么用则需要自己试验。这也是训练大家自学能力和解决问题能力的一步，不能永远让老师"填鸭式教育"。

3、得到驱动器的簇信息、剩余空间大小、总大小

函数原型：

```
BOOL GetDiskFreeSpaceExA(
LPCSTR lpDirectoryName,
PULARGE_INTEGER lpFreeBytesAvailableToCaller,
PULARGE_INTEGER lpTotalNumberOfBytes,
PULARGE_INTEGER lpTotalNumberOfFreeBytes
);
```

lpDirectoryName 是驱动器的名称。

lpFreeBytesAvailableToCaller 是用户可用的磁盘空间。

lpTotalNumberOfBytes 是磁盘总共的空间。

lpTotalNumberOfFreeBytes 是磁盘空闲的空间。以上都是字节为单位。

lpFreeBytesAvailableToCaller 不一定等于 lpTotalNumberOfFreeBytes。操作系统里边可以设置某个用户可以用的最大磁盘，比如说磁盘还有 100G 可用，但是我限制用户 yzk 最多能用 20G 硬盘，因此如果用 yzk 登录的话只有不到 20G 的可用空间。因此 lpTotalNumberOfFreeBytes>=lpFreeBytesAvailableToCaller。

PULARGE_INTEGER 是 64 位的整数，也就是“大整数 BigInteger”，因为 int 是 32 位的，如果用 int 表示磁盘空间的话最多能表示 2 的 32 次方也就是 2G。现在的磁盘都超过了 2G 了，所以要用 64 位的大整数来表示。

（1）示例代码：

```
        int i;
        scanf("%d",&i)
ULARGE_INTEGER nFreeBytesAvailable;
ULARGE_INTEGER nTotalNumberOfBytes;
ULARGE_INTEGER nTotalNumberOfFreeBytes;
if (GetDiskFreeSpaceEx("C:",
    &nFreeBytesAvailable,
    &nTotalNumberOfBytes,
   &nTotalNumberOfFreeBytes))
{
    TCHAR chBuf[256];
    wsprintf(chBuf,"当前用户可用：%I64d,总大小：%I64d,当前可用空间：%I64d\r\n",
        nFreeBytesAvailable,
        nTotalNumberOfBytes,
        nTotalNumberOfFreeBytes);
    MessageBox(NULL,chBuf,"",MB_OK);
}
        else
        {
            MessageBox(NULL,"获取失败","",MB_OK|MB_ICONASTERISK);
        }
```

注意这里 wsprintf 的用法，wsprintf 和 printf 类似，不过 wsprintf 是将结果打印到字符串中，而不是屏幕中。“%I64d”和 printf 中的“%d”一样，它表示此处是 64 位大整数的占位符。

我们在向着 Windows 优化大师一步步前进。耶！！！

（2）课后作业：

上面代码的显示效果很难看，请以 MB 或者 GB 的形式显示结果。而且如果结果小于 1G 则以 MB 为单位显示，否则以 GB 为单位显示。

4、得到磁盘的类型（只介绍，自己课下练习）

UINT GetDriveType(

LPCTSTR lpRootPathName // root directory

);

函数功能：判断磁盘类型

参数说明

lpRootPathName 包含了根目录路径的字符串指针

返回值：

DRIVE_UNKNOWN 未知的磁盘类型

DRIVE_NO_ROOT_DIR 说明 lpRootPathName 是无效的

DRIVE_REMOVABLE 可移动磁盘

DRIVE_FIXED 固定磁盘

DRIVE_REMOTE 网络磁盘

DRIVE_CDROM 光驱

DRIVE_RAMDISK 为 RAM

（1）示例代码：

```
UINT t = GetDriveType("c:");
switch (t)
{
case DRIVE_UNKNOWN:
    MessageBox(NULL,TEXT("未知磁盘类型"),"",MB_OK);
    break;
case DRIVE_NO_ROOT_DIR:
    MessageBox(NULL,TEXT("磁盘名无效"),"",MB_OK);
```

```
        break;
case DRIVE_REMOVABLE:
        MessageBox(NULL,TEXT("可移动磁盘"),"",MB_OK);
        break;
case DRIVE_FIXED:
        MessageBox(NULL,TEXT("固定磁盘"),"",MB_OK);
        break;
case DRIVE_REMOTE:
        MessageBox(NULL,TEXT("网络磁盘"),"",MB_OK);
        break;
case DRIVE_CDROM:
        MessageBox(NULL,TEXT("光驱"),"",MB_OK);
        break;
case DRIVE_RAMDISK:
        MessageBox(NULL,TEXT("RAM"),"",MB_OK);
        break;
default:
        MessageBox(NULL,TEXT("GetDriveType 的返回值非法"),"",MB_OK);
}
```

思考：在 switch 语句最后的 default 不加可以吗？

（2）课后作业：

遍历系统中所有逻辑驱动器，如果驱动器是可移动硬盘或者固定硬盘则显示其可用磁盘空间（如果结果小于 1G 则以 MB 为单位显示，否则以 GB 为单位显示）。此题综合了前边学的所有知识点。

5、得到当前登录用户名：

```
BOOL GetUserName(LPWSTR lpBuffer, LPDWORD nSize);
```

参数含义：

lpBuffer 是获取名称的字符缓冲区。

pcbBuffer 是一个 DWORD 的指针，进入的时候指的是缓冲区的大小，方法返回后指的是

帐号的长度。

```
TCHAR chBuf[256];
DWORD dwRet = sizeof(chBuf)/sizeof(TCHAR);

ZeroMemory(chBuf,256);// ZeroMemory 用来将缓冲区 chBuf 中 n 个字符初始化为 0。

//获取当前登录用户的名称
GetUserName(chBuf,&dwRet);
MessageBox(NULL,chBuf,"",MB_OK);
```

6、得到内存信息：

为什么不像 GetDiskFreeSpaceEx 那样用很多参数？潜规则：一个函数的参数不要多于 5 个。

```
VOID GlobalMemoryStatus(LPMEMORYSTATUS);
    #include <windows.h>
    int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance,
                        PSTR szCmdLine, int iCmdShow)
    {
        MEMORYSTATUS memStatex;
        const int nBufSize = 512;
        TCHAR chBuf[nBufSize];
        GlobalMemoryStatus(&memStatex);

        ZeroMemory(chBuf,nBufSize);

        //内存使用率。

        wsprintf(chBuf,TEXT("内存使用率: %u%%兆\n"),memStatex.dwMemoryLoad);
        MessageBox(NULL,chBuf,"",MB_OK);

        //总共物理内存。

        wsprintf(chBuf, TEXT ("总共物理内存: %u 兆
\n"),memStatex.dwTotalPhys/(1024*1024) );
        MessageBox(NULL,chBuf,"",MB_OK);

        //可用物理内存。
```

```
        wsprintf(chBuf,  TEXT  (" 可 用 物 理 内 存 :  %u  兆
\n"),memStatex.dwAvailPhys/(1024*1024) );
        MessageBox(NULL,chBuf,"",MB_OK);

        //全部的虚拟内存。

        wsprintf(chBuf,  TEXT  (" 全  部  的  内  存  :  %u  兆
\n"),memStatex.dwTotalVirtual/(1024*1024));
        MessageBox(NULL,chBuf,"",MB_OK);

        //看一下这个算法有什么需要完善的地方？（这是笔试、面试重点）

        //1、计算 1024*1024 的地方太多了，应该用一个常量定义 1024*1024 的值

        //这样一处修改，别的地方就修改了

        //2、1024*1024 写成 1048576 不更好吗？省得每次都计算，节省时间？？？

        //NO!还记得《编译原理》那门课中讲的编译器优化吗？编译器会自动将
1024*1024

        //帮我们在编译时算成 1048576，所以这里保留了可读性，同时又不失快速，
嘻嘻

        //这两天更应该注意，都是常考点。
        return 0 ;
    }
```

Phys：physical。物理的。

Avail：Available。可用的。

Virtual：虚拟的。

你的内存是 1G ,这个就是物理内存。操作系统还会把硬盘的一个文件模拟成一个内存，这个内存就是虚拟内存。尽量避免数据在虚拟内存中，因为硬盘访问速度非常慢。你开一个游戏的时候，同时把 word 最小化，windows 就有可能把 word 的内存数据放到虚拟内存中，

把游戏的内存放到物理内存中。windows 里边 c:/ pagefile.sys（需要关闭打开文件夹选项的“隐藏操作系统的文件”）

联系以及复习操作系统这门课中关于虚拟内存的章节，面试时可能会考。

课上练习：思考这两个问题，并且完成优化。联想一道笔试题。

7、取得操作系统的版本：

BOOL GetVersionExW(LPOSVERSIONINFOW);

讲取得版本的方式，然后让学生写下面的代码：

```
OSVERSIONINFO osvi;
    TCHAR * strOSVersion;
    ZeroMemory( &osvi, sizeof( osvi ) );
    osvi.dwOSVersionInfoSize = sizeof( osvi );
    GetVersionEx(&osvi);

    //首先判断是 NT 架构的还是旧架构的

    switch (osvi.dwPlatformId)
    {
  case VER_PLATFORM_WIN32_NT:
    if (osvi.dwMajorVersion == 5&&osvi.dwMinorVersion == 0)
    {
       strOSVersion = TEXT("Windows 2000");
    }
    else if (osvi.dwMajorVersion == 5&&osvi.dwMinorVersion == 1)
    {
       strOSVersion = TEXT("Windows XP");
    }
    else if(osvi.dwMajorVersion == 6&&osvi.dwMinorVersion == 0)
    {
                        strOSVersion = TEXT("Windows Vista");
    }
    else if (osvi.dwMajorVersion == 4&&osvi.dwMinorVersion == 0)
    {
       strOSVersion = TEXT("Windows NT");
    }
    break;
  case VER_PLATFORM_WIN32_WINDOWS:
    if (osvi.dwMajorVersion == 4&&osvi.dwMinorVersion == 10)
    {
```

```
    strOSVersion = TEXT("Windows 98");
  }
  else if (osvi.dwMajorVersion == 4&&osvi.dwMinorVersion == 90)
  {
    strOSVersion = TEXT("Windows ME");
  }
  else
  {
      strOSVersion = TEXT("Windows 95");
    }
    break;
  default:
    break;
}
```

8、得到系统自启动以来的毫秒数：

DWORD GetTickCount(void);

（1）这个函数其实是返回 pc 的 8253rtc 的中断次数。对于 1000HZ 的配置来说，49 天翻转一次。

（2）

```
DWORD dwStart = GetTickCount();
TCHAR buf[256];

wsprintf(buf,"自系统启动以来%d 秒",dwStart/1000);

MessageBox(NULL,buf,"",MB_OK);
```

（3）计算自系统启动以来有多少天、多少小时、多少分。

为什么休眠以后会重置？联系计算机组成原理、数字电路这门课。

9、取得复选按钮的是否选中：

SendDlgItemMessage

SendMessage (hwnd, BM_GETCHECK, 0, 0)

设置复选按钮是否选中：

SendMessage(hwndCancelBtn,BM_SETCHECK,FALSE,0);

课上练习：怎么实现点击按钮让复选框在“选中”、“未选中”中切换呢？

[Error] D:\俺的文档\C-Free\Projects\ctrl002\main.c:59: error: redeclaration of `BOOL bChecked'

declare：声明。declaration：声明（名）。redo：重做，redraw：重绘，repeat：重复

redeclaration：重复声明

括号是变量的作用域

Case 语句哪怕只有一个行，也要加大括号。好的习惯。

[Error] D:\GREENI~1\MINGW3~1.5\include\winuser.h:3909: error: too many arguments to function `LONG SendDlgItemMessageA(HWND__*, int, UINT, WPARAM, LPARAM)'

too many arguments to function

arguments：参数

function：函数

太多的参数给函数。

6、隐藏、显示、禁用、启用控件：

隐藏：ShowWindow (hwndChild, SW_HIDE) ;

显示：ShowWindow(hwndEdtName,SW_NORMAL);

判断控件是否可见：IsWindowVisible (hwndChild) ;

禁用控件：EnableWindow (hwndChild, FALSE) ;

启用控件：EnableWindow(hwndEdtName,TRUE); ：

判断控件是否可用：IsWindowEnabled (hwndChild) ;

7、ListBox

（1）添加项：

如果采用LBS_SORT样式 那么填入清单方块最简单的方法是借助LB_ADDSTRING消息：

SendMessage (hwndList, LB_ADDSTRING, 0, (LPARAM) szString) ;

如果没有采用 LBS_SORT ，那么可以使用 LB_INSERTSTRING 指定一个索引值，将字符串插入到清单方块中：

SendMessage (hwndList, LB_INSERTSTRING, iIndex, (LPARAM) szString) ;

示例代码：

SendMessage(hwndLB,LB_INSERTSTRING,0,(LPARAM)TEXT("你好"));

（2）删除项：

在指定索引值的同时使用 LB_DELETESTRING 参数 这就可以从清单方块中删除字符串：

SendMessage (hwndList, LB_DELETESTRING, iIndex, 0) ;

（3）取得有多少项：

iCount = SendMessage (hwndList, LB_GETCOUNT, 0, 0) ;(4)选定某一项 SendMessage (hwndList, LB_SETCURSEL, iIndex, 0) ;将 iParam 设定为-1 则取消所有选择(5) 确定目前选项的索引 :iIndex = SendMessage (hwndList, LB_GETCURSEL, 0, 0) ;如果没有项目被选中，那么从呼叫中传回的 iIndex 值为 LB_ERR。（6）将某项目复制到文字缓冲区中：iLength = SendMessage (

hwndList, LB_GETTEXT, iIndex,

(LPARAM) szBuffer) ;确定清单方块中字符串的长度：iLength = SendMessage (hwndList, LB_GETTEXTLEN, iIndex, 0) ;对以 NULL 字符终结的字符串长度来说，szBuffer 数组必须够大。您也许想用 LB_GETTEXTLEN 先分配一些局部内存来存放字符串。

课下练习：

用户在【姓名】中输入姓名，点击【输入】按钮，如果姓名已经存在，则提示“姓名重

复"不能添加，否则将此姓名加入 ListBox。

用户在【姓名】中输入姓名，点击【查找】按钮，如果姓名存在于 ListBox 中，则将此姓名选中；

用户点击【显示当前姓名】按钮，则将用户当前选择的姓名 MessageBox 出来，如果没有任何被选中，则提示"没有人员被选中。"

10、RadioButton 的分组

各个组内按钮的 id 要从小到大排列，而且各组之间的 id 段不能重叠。举例：

第一个组：

```
#define IDC_RBN1 1002
#define IDC_RBN2 1003
```

第二个组：

```
#define IDC_RBN3 1007
#define IDC_RBN4 1008
#define IDC_RBN5 1010
```

一定要设置所有 RadioButton 的 Auto 属性为 False。然后在响应 WM_COMMAND 的代码中如下编写：

```
case IDC_RBN1:
case IDC_RBN2:
CheckRadioButton(hwnd,IDC_RBN1,IDC_RBN2,id);
break;
case IDC_RBN3:
case IDC_RBN4:
case IDC_RBN5:

CheckRadioButton(hwnd,IDC_RBN3,IDC_RBN5,id); //选中
```

IDC_RBN3 至 IDC_RBN5

段的 id 为 id 的控件。

```
break;
```

9、菜单资源

（1）制作好菜单资源后，在对话框的 Menu 属性中选择刚才制作好的对话框。

（2）响应菜单操作。菜单点击后会发送 WM_COMMAND 消息，使用 HANDLE_MSG 进行映射后 id 就是被选择菜单的 id。例子：

```
case IDM_QUIT:
EndDialog(hwnd,id);
```

本节课源代码下载（请自己敲代码，而不是直接下载老师的代码）：

这节课是自己动手写 QQ、自己动手写飞鸽传书的基础，以后想从事网络开发、信息系统开发、游戏开发、嵌入式系统开发等的同学必须学习。

演示一个简单的和网易 Email 服务器对话的过程。

1、网络基本概念

讲解 QQ 的基本原理

讲解浏览器上网的基本原理

服务器：Server。提供服务的计算机。

客户端：Client。和服务器端交互的一个设备。

IP 地址：计算机在网络中的地址。相当于每个家庭的电话。211.97.1.2

别名、域名：www.RuPeng.com。域名就相当于你家的地址。问 114，你家的电话（告诉地址）

DNS：网络中的 114。Domain Name System。

端口：Port。不同的程序连接服务器时候用的插座。（程序和服务器通信的时候的一个开得的一个口子）

Socket（套接字）：导线、网络连接。

2、

WinSock、WinSocket。

Socket:平台无关的。Socket 标准。所有操作系统都实现了这个标准。Send、recv。。。标准的。一通百通!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

WinSock：添加了一些 Windows 独有的函数。而且不用这些函数的话 Socket 程序还跑不起来。羞臊微软!!!

WinSock 编程:平台无关，一通百通。WSA(Windows Socket API)

Sz1.tencent.com
Sz2.tencent.com

3、

LastError

调用 WindowsAPI 的时候，每步执行完毕都会把执行结果放到 LastError 中去，所以应该在函数执行完毕立即去取。

DeleteFile(f1)
DeleteFile(f2)
GetLastError()

LastError 只能容纳一个错误码

DeleteFile(f1)
GetLastError()
DeleteFile(f2)
GetLastError()

错误码。2233、344555

FormatMessage：把错误码转换为错误消息。

Windows 错误处理

GetLastError

```
void ShowError()
    {
    TCHAR* lpMsgBuf;

    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER| //自动分配消息缓冲区
```

```
FORMAT_MESSAGE_FROM_SYSTEM, //从系统获取信息

NULL,GetLastError(), //获取错误信息标识

MAKELANGID(LANG_NEUTRAL,SUBLANG_DEFAULT),//使用系统缺省语言

(LPTSTR)&lpMsgBuf, //消息缓冲区

0,
NULL);
MessageBox(NULL,lpMsgBuf,"",MB_ICONERROR);
}
```

在.h 中声明函数，在.c 中实现!!!

都会有哪些错误？

4、模板代码：

（1）、添加 winsock2.h

（2）、连接库“wsock32”。怎么知道添加它呢？

(3)、模板代码

```
WSADATA wsaData;

//初始化 Socket 库

WSAStartup(MAKEWORD(2,0),&wsaData);

//创建一根电线

SOCKET sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
SOCKADDR_IN sa;
sa.sin_family=AF_INET;

//设置电线连接服务器端的端口

sa.sin_port = htons(IPPORT_SMTP);

//123.125.50.135 是 ping smtp.163.com 出来的，后面会讲使用 gethostbyname 来直接
从

//主机名得到 ip 地址

sa.sin_addr.S_un.S_addr = inet_addr("123.125.50.135");
```

```
//为什么不用设置客户端的端口，难道不需要客户端的端口吗？
if(connect(sock,(SOCKADDR *)&sa,sizeof(sa))==SOCKET_ERROR)
{
ShowError();
return;
}
//做事情。把大象放冰箱总共分几步
closesocket(sock);
WSACleanup();
```

QQ 服务器的端口 8888。需要指定客户端的端口是 9999.

端口有一个特点：排他性！端口已经被别的程序占用，不能再用这个端口。

开两个 QQ。11111111111 22222222222 9999。如果端口可以被多个程序使用的话，QQ 聊天服务器发过来的消息被多个 QQ 收到。很荒谬! 排他性！

如果在程序里写死了客户端用的端口。。。所以才需要 Socket 替客户端动态分配一个端口。所以不需要在程序中显示指定客户端的端口号！

客户端向服务器端发消息：send

服务器端想客户端发消息，客户端这边应该接收，recv。（ receive ）

SMTP 服务器，你连上来以后服务器端主动向你发一条消息

```
recv(sock,buffer,256,0);
```

第一个是使用的 socket，“导线”，套接字

接收数据的缓冲区的指针

接收数据的缓冲区的大小

0

```
send(sock,cQuit,lstrlen(cQuit),0);
```

第一个是使用的 socket，“导线”，套接字

接收数据的缓冲区的指针

接收数据的缓冲区的大小

0

(4)一上来人家就主动和你唠嗑：

```
char buffer[256];
ZeroMemory(buffer,sizeof(buffer)/sizeof(char));

//接收问候语

recv(sock,buffer,256,0);
MessageBox(hwnd,buffer,"",0);
```

（5）朋友再见

```
//注意不能忘了末尾的回车

TCHAR cQuit[] = "QUIT\n";
send(sock,cQuit,lstrlen(cQuit),0);
ZeroMemory(buffer,sizeof(buffer)/sizeof(char));

//接收 GoodBye

recv(sock,buffer,256,0);
MessageBox(hwnd,buffer,"",0);
```

九、第一次亲身体验一通百通的威力！

一、为什么现在要转换到 VC6 了？

（1）、亲身体验一通百通的威力。

（2）、C-Free 做入门级的学习工具比较合适，所以在《C 语言也能干大事》一开始我们选择它做入门工具。但是 C-Free 在工程化的大程序方面能力欠佳，而《C 语言也能干大事》后面将是大量的实战项目，比如自己动手写 QQ、自己动手写银行系统、自己动手写 VC 等，所以要转用 VC。

（3）用 c-Free 的目的是让同学们知道不是 C/C++就是 VC，VC=编辑器+编译器；C-Free=编译器+可替换的编译器。C-Free 与 VC。

（4）用 VC 开发出来的工程比较好和同学们交流，比较好做为课程设计、毕业设计的作业直接发给老师。

别害怕。只用 10 分钟就能让你从 C-Free 过渡到 VC，这就是一通百通的魅力。跟我来吧。

为什么不同 VC 更高的版本？

VS 更高版本和 VC6 比起来几乎没什么差别，特别是对于咱们《C 语言也能干大事》这门课约等于一模一样，而且高版本体积要大很多，要占 2G 多，装了很多用不着的东西，而 VC6 只有 200 多 MB。

二、VC 开发环境的搭建

1、VC 的下载

如果已经安装好 VC 的就不用再重新安装了。

如果没有安装的同学可以下载如鹏网专门为大家定制的“如鹏网版 VC6”，去掉了一些已经作废的组件，丝毫不影响使用，尺寸只有 103MB。

下载地址：http://down1.rupeng.com/download/software/RuPengvc6_chs.rar

注意如果使用 Vista 的同学请关闭 Vista 的 UAC 这个废物。

2、

MessageBox 归来

Win32Alication→简单的 Win32 程序。

三、对话框图形化环境搭建

1、从 Win32 Alication 向导生成对话框程序

自己配置太麻烦了，每次建一个程序就需要建一遍对话框、MainProc、添加 LoadLibrary 等等。

2、配置如鹏网为同学们定制的 Win32 DialogBased Alication 向导

已经下载“如鹏网版 VC6”的同学从安装包的 Tools 目录下就可以找到 Win32DlgBasedAWizard.zip

没有下载“如鹏网版 VC6”的同学从下面的地址单独下载：http://down1.rupeng.com/download/software/Win32DlgBasedAWizard.zip

只有 200 多 K 大小。

配置方法：

将 DlgBaseAWizard.pdb 和 DlgBaseAWizard.awx 拷贝到 VC6 的安装目录的 Common\MSDev98\Template 目录下即可，一般 VC6 都安装到了 C 盘，所以一般只要拷贝到 C:\Program Files\Microsoft Visual Studio\Common\MSDev98\Template 即可。然后重启 VC6，在 VC6 的向导页面中就可以看到“Win32 DialogBased Alication”的向导选项了。

Win32 DialogBased Alication 并没有做什么神奇的工作，只不过是帮大家快速搭建一个对话框工程而已，没有什么自己的特性，大家手动也能完成，这个工程拿到没有“Win32 DialogBased Alication”的 VC 中也能运行和开发。

注：“Win32 DialogBased Alication”就是在“自己动手写开发工具”，希望大家早日从用工具的人变成造工具的人。人类进化的一大标志就是学会制造工具！哈哈！

3、

Win32 DialogBased Alication 帮大家做了什么

添加了一个对话框、添加了模板代码、添加了对 IP 地址控件、日历控件等高级高级控件的显示支持代码。

4、争取大家自己从头不用 Win32 DialogBased Alication 建一个对话框工程。

五、配置 VC6，使之更好用

1、修改代码自动完成的快捷键，因为默认的 Ctrl+Space 和输入法切换冲突

工具栏上点击右键，选择“自定义”（customize）；在对话框中切换到“键盘”（keyboard），“类别”选择“编辑”，在命令中找到“CompleteWord”（列表中快速查找的技巧，在几乎任何软件中都通用，在列表中快速敲入单词），然后焦点设置到“按下新快捷键”上，按快捷键“Alt+/”（也可以是其他的自己习惯的，只要是不和其他的冲突就可以），主要是按快捷键，不是敲入字符“Alt+/”，学了后面的课程你就会知道这是一个 HotKey 控件，而不是 TextBox。

然后注意不要忘了点击【分配】。

要学会使用快捷键，我看很多同学写程序太费劲了，经常用鼠标点来点去，慢的像一头牛，CtrlC、CtrlV 什么的起码不应该总是用鼠标，牛逼的程序员很少用鼠标。

2、

关闭工程的“设置”→“浏览信息”→“创建浏览信息文件”（ Browsing Info ）

Project→Setting→BrowseInfo→build BrowseInfo File

因为它有可能造成 VC 死锁

3、

如果 VC 还是有偶尔死锁，那么请杀掉 MSDev 进程，然后重启 VC。

移植几个程序：计算器；音乐图片贺卡；设置程序图标；

十、Socket 网络开发 2

这节课是对上节课内容的复习和实战，并且融合了 C 语言中指针、函数、类型转换、结构体等知识。

一、复习：

复述 QQ 的基本原理；在一个典型的网络程序中分为哪两端？

IP 地址、域名、DNS、端口是做什么用的？

演示一个简单的和网易 Email 服务器对话的过程。

Socket 是 Windows 特有的吗？什么是 WinSock？

WinSock 和 Socket 标准的区别和联系。

Windows 中函数的错误信息是通过什么来报告的？

创建客户端到服务器端的 Socket 连接的时候需要指定服务器端的端口吗？为什么？需要指定客户端的端口吗？为什么？客户端不用显式指定端口，但并不是说不需要端口，而是这个端口是由操作系统分配的。

有一个端口能同时被多个程序占用吗？

发送、接收数据各用什么函数？

二、开发网络定时器

1、看看 Windows 的网络授时功能（客户端）

2、网络授时原理

客户端连接到授时服务器上，授时服务器会马上将当前时间发送给客户端，原理很简单。

网络授时分为好几种协议，比如返回的是日期字符串，有的返回的是自 1900 年 1 月 1 日至今的秒数。“2009-03-08 22:13:58”、39393933993

扩展阅读：http://zhidao.baidu.com/question/20121991.html

2、

国家授时中心提供的网络授时返回的是自 1900 年 1 月 1 日至今的秒数。

3、

先移植第一节课的代码

可供选择的授时服务器（不止这些）：

www.time.ac.cn

国家授时中心

Clock.sgi.com

SGI

Tick.mit.edu

麻省理工学院

Time.nist.gov

NIST

Compile：编译

Link：连接

编译是单一的文件编译成.o、.obj 文件，然后 Link 是.obj 连接成.exe

Link 库！！！！

一通百通的力量！！！！！！！！！！！！！！

C-Free、VC 没有任何差别！！！！！！！！！！！！

小技巧：在命令提示符下不能 CtrlC、CtrlV，但是选中后点右键就是复制、粘贴

4、

授时服务器是 37 端口 IPPORT_TIMESERVER。返回的是什么？

```
unsigned long ulTime;

recv(sock,(char *)&ulTime,sizeof(unsigned long),0);
```

深刻理解这段代码的含义

```
ulTime = ntohl(ulTime);
```

从**网络字节顺序**转换为**主机字节顺序。**这个是重点，很多公司面试的时候都会问这个，以后工作中也会遇到。

什么是网络字节顺序、什么是主机字节顺序。

不同的 CPU 中处理整数的方式不一样，有的是低位在前、有的是高位在前。计算机组成原理、数字电路。。。

网络字节顺序则是统一标准。

如果直接接收的话就会造成接受者接收的数据正好是相反的。

扩展阅读：http://hi.baidu.com/cdmember_daihw/blog/item/ad83090f1fb4ffebab6457e6.html

中国是东 8 区。猜的：时间服务器返回的是格林尼治时间。

Windows 中时间用 SYSTEMTIME 结构体表示时间，怎么将 ulTime 转换为 SYSTEMTIME？网上找到下面的代码：

```
#define HIGHTIME
21968699 // 21968708 // Jan 1, 1900 FILETIME.highTime
#define LOWTIME
4259332096 // 1604626432 // Jan 1, 1900 FILETIME.lowtime
```

```
SYSTEMTIME st;

UINT64 uiCurTime, uiBaseTime, uiResult;

uiBaseTime = ((UINT64) HIGHTIME << 32) + LOWTIME;

uiCurTime = (UINT64)dwTime * (UINT64)10000000;

uiResult = uiBaseTime + uiCurTime;
FileTimeToSystemTime((LPFILETIME)&uiResult, &st);
```

把它封装成函数，怎么设计接口？深刻理解指针以及 Windows 中 LP 的惯例。

把时间打印出来，怎么打印？？？？？？

怎么时间不对

设置系统时间

```
//SetSystemTime(&st);
```

作业：定时的进行网络对视。SetTimer 函数，1 分钟对时一次。时钟。

Over！

十一

网络编程由于比较难，留给同学们一段时间去学习，数据库开发讲若干节以后再回头来讲自己动手写邮件发送器（网络编程的应用）和网络编程的服务器端开发。

这个知识点要达到的目的：能够使用常用的 SQL 语句，能够使用 ODBC 操作数据库，熟悉结果集、SQL 注入漏洞、范式等一通百通的东西，为以后有必要的情况下用 ADO、ADO.Net、JDBC 学一通百通的东西。

课程参考教材：杨中科的《程序员的 SQL 金典》

这门课和《数据库原理》的关系，这门课是对《数据库原理》的应用，通过这门课将能够学会数据库原理的实际应用，不再云里雾里。即使没学过《数据库原理》也没关系，这门课并不假定大家学过《数据库原理》，相当于预习效果。

企业里数据库开发是很大的一部分，不会什么也要会数据库开发。

开发环境的搭建：

（1）如鹏版绿色 MYSQL 的下载：

http://down1.rupeng.com/download/software/RuPengGreenMYSQL.rar

（2）ODBC 连接 MYSQL 的驱动 MYODBC 下载：

http://down1.rupeng.com/download/software/MYODBC.rar

（3）管理工具 SQLyog 下载：

http://down1.rupeng.com/download/software/SQLyog.rar

依次安装三个软件。如鹏版绿色 MYSQL 的使用，使用 SQLyog 连接数据库。

注意：如果初始化 MYSQL 以后移动了 MYSQL 的文件夹，那么需要重新运行“运行前先初始化.exe”。

1、执行“启动 MySQL.bat”

2、运行 SQLyog。如鹏版 MYSQL，用户名密码都是 root。

什么是数据库，数据库能做什么？

可以记录到文件中，比如：

姓名　年龄　工号　职位

Kider

20

001

超版

CALF

21

002

版主

劣势：

1、必须对文件操作非常熟悉

2、比如对算法非常熟悉；一旦算法不好的话很容易造成性能问题。

3、并发操作。

广义的来讲，能够存储数据的地方都可以叫数据库。DBMS（DataBase Management System，数据库管理系统），由 DBMS 来负责管理数据，使用者只要"描述"要进行什么操作就可以了，DBMS 相当于仓库管理员，只要向仓库管理员发出命令就可以，他怎么做我们不用关心，只要告诉他 What To Do（要做什么），不关心 How To Do（怎么做）。

DBMS 和数据库的关系。大家都很懒，所以很多时候说数据库其实指的是 DBMS。行业内的潜规则。

不同的仓库管理员有自己的不同的优势。不同品牌的 DBMS 也有自己的不同的特点。

MYSQL、MSSQLServer、DB2、Oracle、Access。。。

OS/2

Sybase SQLServer、Microsoft SQLServer

Sybase 和 MS 的恩恩怨怨。

主流数据库（DBMS）有哪些？

SQL<>SQLServer<>MSSQLServer。最常见的错误。行业内的潜规则。

和仓库管理员打交道的方式：1、跑过去和他说；2、打电话；3、通过电脑传递信息（自动化的方式）；

操作数据库的两种方式：管理工具和程序代码。

什么是 Catalog（分类）（数据库、表空间）

表（Table）

把仓库分成不同的区域。不同的区域放不同类型的物品。

将一个数据库（Catalog）分成不同的表，每个表放不同的数据。公司机密信息数据库：发票信息；合同信息；公司帐号；。。。。建不同的表。

列（Column）

已经将生肉放到单独的生肉区域里，每块肉都有不同的特性，取肉部位、重量。。。。为每块肉都贴一个标签，

规定每一列的格式，重量应该是数字。生产日期应该是日期类型。数据类型（DataType），Int、Char、double 一个道理。

数据库，表、列、数据类型、记录。

SQLYog 中左边是数据库。

Int：整数；varchar：字符串（String），Var（可变的），Char（字符）；

Boolean、char、date

怎么使用 Varchar，要指定长度，就像申明 char buff = new char[200]

使用工具创建数据库表，插入一条记录。表的命名规则、字段命名规则。常见数据类型说明。

下节课内容：SQL 语句、数据的增删改查。

课下参考《程序员的 SQL 金典》进行深一步的学习和预习。

本课程的参考教材：杨中科的《程序员的 SQL 金典》第三章

什么是 SQL 语句。

SQL 语句是和 DBMS“交谈”专用的语句，和 C、Java 等的代码不一样，SQL 语句你需要你告诉 DBMS What To Do 就可以，不需要告诉它 How To Do。

本课程的命名规范：所有表以 T_开头，字段（列、Column、Field）以 F 开头，这样可以很容易的看清哪个是表、哪个是字段。而且不容易和关键字冲突。

增删改查：Insert、Delete、Update、Select。

SQL 语句是大小写不敏感的

1、简单的 Insert 语句。向数据库中插入数据。Into 后的列名和 values 一一对应，字符串用单引号

```
Insert Into T_Person(FName,FAge) values('Jim',25)
Insert into T_Person(FName) values('lily')
```

2、怎么样查看表中数据

```
Select * from T_Person
Select * from  表名
```

3、简化的 Insert 语句格式，不推荐

```
Insert into T_Person values('poly',33)
```

4、数据的更新 Update

```
Update T_Person Set FAge=30
```

5、多字段更新

6、带 Where 语句的 Update

```
update T_Person Set FAge=50 where FName='Tom' OR FName='Jim'
```

7、数据的删除

Delete From T_Person

是删除 T_Person 表中的数据，并不是删除 T_Person 整个表结构。表还在，数据没了，人去楼空。

8、带 Where 语句的删除

```
delete from T_Person where FName='Jim'
delete from T_Person where FAge>30
```

下节课内容：数据的检索。

课下参考《程序员的 SQL 金典》进行深一步的学习和预习。

十二、数据库开发

本课程的参考教材：杨中科的《程序员的 SQL 金典》第四章

1、SELECT 基本用法

a)

检索所有列、检索指定列

b)

按条件过滤。通配符、Between and

```
select * from T_Employee where FSalary>4000 and FSalary<8000
select * from T_Employee where FSalary BETWEEN 4000 And 8000
```

c)

数据汇总

```
select SUM(FSalary),AVG(FSalary),MAX(FSalary),MIN(FSalary) from T_Employee
select SUM(FSalary),AVG(FSalary),MAX(FSalary),MIN(FSalary) from T_Employee where FSalary<8000
```

select COUNT(*) from T_Employee where FSalary>5000。满足条件的数据个数

d)

排序

Ascending：ASC，升序

Descending：Des，降序

```
select * from T_Employee order by FSalary desc
```

2、C 语言中访问数据库

a)

ODBC 简介。ODBC 是微软提供的访问数据库的一种标准接口，通过 ODBC 可以连接 MSSQLServer、MYSQL、DB2、Oracle、Access 等各种数据库，通过统一的函数进行访问，也就是访问各种数据库都可以使用统一的函数。屏蔽了连接不同数据库的差异性。

b)

除了 ODBC 之外还有 ADO、ADO.net 等，Java 中有 JDBC 等。都有连接、结果集、游标、事务、参数化 SQL 等概念，一通百通。

c)

没安装 MYODBC(MYSQL 的 ODBC 驱动)的首先安装 MYODBC。

http://down1.rupeng.com/download/software/MYODBC.rar

d)

sql.h、sqlext.h、sqltypes.h

e)

添加连接库"odbc32.lib odbccp32.lib"。如果使用的是 rupeng 的 DialogBased 向导则已经自动添加。如果使用其他向导或者 C-Free、PellesC 等开发工具，则需要手动添加

f)

看模板代码。不用关心每个函数，用的时候 copy 以后改一改就可以，理解只要流程、主要概念即可，不要深究。有兴趣的可以研究 CHECKDBSTMTERROR 宏的实现。

g)

访问数据库可能遇到的错误：连接错误、执行错误。

h)

连接数据库、执行 SQL、断开连接。连接字符串。数据库错误处理。

SQLHDBC hdbc：代表一个数据库连接句柄。和 Socket 里边的 Socket 连接类似，要访问数据库，先要连接到数据库。SQLHDBC：SQL、H 句柄、DB（DataBase）、C（Connection）

SQLHSTMT hstmt：代表一个 SQL 语句。STMT（Statement，语句）

SQLRETURN：执行结果。

ODBC 中字符串用 SQLCHAR，SQLCHAR 其实就是 char 的别名。

SQLCHAR ConnStrIn[MAXBUFLEN]和 char ConnStrIn[MAXBUFLEN]一样。

ConnStrIn：连接字符串，你要连接到的数据库的驱动、ip 地址、用户名密码、数据库名（Catalog）等等都在连接字符串里描述。DRIVER：使用的驱动名；SERVER：ip 地址；UID：用户名（UserID）；PWD（Password）：密码。DataBase：数据库名，Catalog。

SQLDriverConnect：创建到数据库的连接，使用 ConnStrIn 连接。result 表示执行结果。如果失败了 SQL_ERROR==result。 ShowDBConnError(hwnd,hdbc);来显示“连接错误”。

result = SQLPrepare(hstmt,(SQLCHAR*)"insert into T_Person(FAge,FName) values(20,'kider')",SQL_NTS)。创建 SQL 语句的句柄。

result =SQLExecute(hstmt);：执行 SQL 语句。

SQLFreeStmt(hstmt,SQL_CLOSE);：释放 SQL 语句

SQLDisconnect(hdbc);：断开数据库连接。

课下参考《程序员的 SQL 金典》进行深一步的学习和预习。

模板代码：

购买此贴后可以得到整个工程的源代码：

```
#include "stdafx.h"
    #include <windows.h>
    #include <windowsx.h>
    #include <sql.h>
    #include <sqlext.h>
    #include <sqltypes.h>
    #include "resource.h"
    #include "MainDlg.h"
    #define LOGIN_TIMEOUT 30
    #define MAXBUFLEN 255
    #define CHECKDBSTMTERROR(hwnd,result,hstmt) if(SQL_ERROR==result){ShowDBStmtError(hwnd,hstmt);return;}
    BOOL WINAPI Main_Proc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
    {
```

```
    switch(uMsg)
    {
        HANDLE_MSG(hWnd, WM_INITDIALOG, Main_OnInitDialog);
        HANDLE_MSG(hWnd, WM_COMMAND, Main_OnCommand);
  HANDLE_MSG(hWnd,WM_CLOSE, Main_OnClose);
    }
    return FALSE;
}
BOOL Main_OnInitDialog(HWND hwnd, HWND hwndFocus, LPARAM lParam)
{
    return TRUE;
}
void Main_OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify)
{
    switch(id)
    {
        case IDC_OK:
    DBTest(hwnd);
        break;
        default:
  break;
    }
}
void Main_OnClose(HWND hwnd)
{
    EndDialog(hwnd, 0);
}
void ShowDBError(HWND hwnd,SQLSMALLINT type,SQLHANDLE sqlHandle)
{
    char pStatus[10], pMsg[101];
    SQLSMALLINT SQLmsglen;
    char error[200] = {0};
    SQLINTEGER SQLerr;
    long erg2 = SQLGetDiagRec(type, sqlHandle,1,
                    (SQLCHAR *)pStatus,&SQLerr,(SQLCHAR *)pMsg,100,&SQLmsglen);
    wsprintf(error,"%s (%d)\n",pMsg,(int)SQLerr);

    MessageBox(hwnd,error,TEXT("数据库执行错误"),MB_ICONERROR|MB_OK);

}
void ShowDBConnError(HWND hwnd,SQLHDBC hdbc)
{
ShowDBError(hwnd,SQL_HANDLE_DBC,hdbc);
}
```

```
void ShowDBStmtError(HWND hwnd,SQLHSTMT hstmt)
{
ShowDBError(hwnd,SQL_HANDLE_STMT,hstmt);
}
void DBTest(HWND hwnd)
{
   SQLHENV henv = NULL;
   SQLHDBC hdbc = NULL;
   SQLHSTMT hstmt = NULL;
   SQLRETURN result;
   SQLCHAR ConnStrIn[MAXBUFLEN] = "DRIVER={MySQL ODBC 5.1 Driver};SERVER=127.0.0.1;UID=root;PWD=root;DATABASE=test;CharSet=gbk;";
   SQLCHAR ConnStrOut[MAXBUFLEN];
  //分配环境句柄
   result = SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv);
  //设置管理环境属性
   result = SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0);
  //分配连接句柄
   result = SQLAllocHandle(SQL_HANDLE_DBC, henv, &hdbc);
  //设置连接属性
   result = SQLSetConnectAttr(hdbc, SQL_LOGIN_TIMEOUT, (void*)LOGIN_TIMEOUT, 0);
  //连接数据库
   result = SQLDriverConnect(hdbc,NULL,
                  ConnStrIn,SQL_NTS,
                  ConnStrOut,MAXBUFLEN,
                  (SQLSMALLINT *)0,SQL_DRIVER_NOPROMPT);
   if(SQL_ERROR==result)
   {
      ShowDBConnError(hwnd,hdbc);
      return;
   }
//初始化语句句柄
   result = SQLAllocHandle(SQL_HANDLE_STMT, hdbc, &hstmt);
   //SQL_NTS telling the function the previous parameter is Null-Terminated String,
//please alculate the string length for me
```

```
    result = SQLPrepare(hstmt,(SQLCHAR*)"insert into T_Person(FAge,FName)
values(20,'kider')",SQL_NTS);
    CHECKDBSTMTERROR(hwnd,result,hstmt);
      result =SQLExecute(hstmt);
      CHECKDBSTMTERROR(hwnd,result,hstmt);
    SQLFreeStmt(hstmt,SQL_CLOSE);
      SQLDisconnect(hdbc);
      SQLFreeHandle(SQL_HANDLE_DBC,hdbc);
      SQLFreeHandle(SQL_HANDLE_ENV,henv);

MessageBox(hwnd,TEXT("执行成功"),TEXT("标题"),MB_OK);

}
```

十四

1、处理数据库查询

序号从1开始，不是0

```
SQLINTEGER cbsatid=SQL_NTS;
   while (SQLFetch(hstmt)!=SQL_NO_DATA_FOUND)
   {
            SQLCHAR name[20];
     SQLGetData(hstmt,1,SQL_C_CHAR,name,20,&cbsatid);
   }
```

```
SQLINTEGER cbsatid=SQL_NTS;
    while (SQLFetch(hstmt)!=SQL_NO_DATA_FOUND)
    {
    SQLINTEGER i;
    SQLGetData(hstmt,2,SQL_C_LONG,&i,sizeof(SQLINTEGER),&cbsatid);
    }
```

乱码问题解决方式，连接字符串使用GBK，修改数据库默认编码为gbk

4、保存HDBC到全局变量中，长连接、短连接。比如Java、C#中访问数据库复杂多少。一通百通。

要掌握调试的技巧，分析问题可能出错的原因和出错的地方，然后调试分析。一定不能和书上不一致，就懵了

5、自己动手写用户管理系统：新增密码字段、新建用户，检验登录；

```
SQLINTEGER cbsatid=SQL_NTS;
    //需要一行一行的读取，这种方式就叫做通过游标读取，无论是在JDBC、ADO/ADO.net。。。结果集
    //调用SQLFetch一次就向下读取一行，直到返回值为SQL_NO_DATA_FOUND的时候表示读到了最后
```

```
//是不是和 C 语言文件访问中 EOF 有点像。
  while (SQLFetch(hstmt)!=SQL_NO_DATA_FOUND)
  {
 TCHAR  name[20]; //字符数组，SQLCHAR 其实就是 char 的一个别名，所以和 char  name[20]
一样。
  //调用 SQLGetData 来取列（Column）的内容
  //第一个参数就是代表 SQL 语句的 hstmt、第二是要读取的列的序号（从 1 开始）
  //第三个是列的类型(SQL_C_CHAR 字符串 ,在 SQLExt.h 中所有以 SQL_C_开头的都是可选值)
  //第四个参数就是要接收的值的指针 !!!
  //第五个是指针指向的缓冲区的大小
  //第六个不用管，那么调就行
    SQLGetData(hstmt,1,SQL_C_CHAR,name,sizeof(name)/sizeof(SQLCHAR),&cbsatid);
 MessageBox(hwnd,name,TEXT(""),MB_OK);
  }
while (SQLFetch(hstmt)!=SQL_NO_DATA_FOUND)
  {
 SQLINTEGER  i; //字符数组，SQLCHAR 其实就是 char 的一个别名，所以和 char  name[20]一
样。
  //调用 SQLGetData 来取列（Column）的内容
  //第一个参数就是代表 SQL 语句的 hstmt、第二是要读取的列的序号（从 1 开始）
  //第三个是列的类型(SQL_C_CHAR 字符串 ,在 SQLExt.h 中所有以 SQL_C_开头的都是可选值)
  //第四个参数就是要接收的值的指针 !!!
  //第五个是指针指向的缓冲区的大小
  //第六个不用管，那么调就行
    SQLGetData(hstmt,2,SQL_C_LONG,&i,sizeof(i),&cbsatid);
TCHAR name[20];
```

```
SQLGetData(hstmt,1,SQL_C_CHAR,name,sizeof(name)/sizeof(TCHAR),&cbsatid);
TCHAR s[20];

wsprintf(s,TEXT("%s 年龄是:%d"),name,i);


MessageBox(hwnd,s,TEXT(""),MB_OK);
  }
result = SQLPrepare(hstmt,(SQLCHAR*)"select FUserName,FPassword from T_User",SQL_NTS);
CHECKDBSTMTERROR(hwnd,result,hstmt);
   result =SQLExecute(hstmt);
   CHECKDBSTMTERROR(hwnd,result,hstmt);
SQLINTEGER cbsatid=SQL_NTS;
TCHAR inputUserName[20];

GetDlgItemText(hwnd,IDC_EDITUSERNAME,inputUserName,sizeof(inputUserName)/sizeof(TCHAR));
TCHAR inputPassword[20];

GetDlgItemText(hwnd,IDC_EDITPASSWORD,inputPassword,sizeof(inputPassword)/sizeof(TCHAR));
BOOL found=FALSE;
while (SQLFetch(hstmt)!=SQL_NO_DATA_FOUND)
   {
 TCHAR userName[20];
 SQLGetData(hstmt,1,SQL_C_CHAR,userName,sizeof(userName)/sizeof(TCHAR),&cbsatid);
 TCHAR password[20];
 SQLGetData(hstmt,2,SQL_C_CHAR,password,sizeof(password)/sizeof(TCHAR),&cbsatid);
 if(0==lstrcmp(inputUserName,userName))
 {
  if(0==lstrcmp(inputPassword,password))
  {

  MessageBox(hwnd,TEXT("输入正确，登陆成功！"),TEXT("提示"),MB_OK);

  found = TRUE;
  break;
  }
 }
  }
if(FALSE==found)
{

 MessageBox(hwnd,TEXT("输入错误"),TEXT("报错"),MB_OK|MB_ICONERROR);

}
```

缺点是？？？

数据量大了以后运行速度很慢。比如说咱们如鹏 100 万会员。

Wsprintf

课下作业：

（1）

细化报错信息：当用户名不存在的时候提示“用户名不存在”，当用户名存在而密码错误的时候报告“密码错误”。

（2）

保存用户的时候，如果用户已经存在则更新已有的用户信息；否则新增用户信息。

(3)删除指定用户名的用户；

(4)有能力的同学研究 ListView 控件的使用，列出当前数据库中的所有用户信息，选择某个用户可以删除、可以更新。

下节课内容：

作业点评：ListView 控件的使用与用户管理系统。

SQL 注入漏洞及参数化 SQL

1、

程序中打开新对话框，其实有代码可参考，WinMain。

DialogBox(hInstance, MAKEINTRESOURCE(IDD_MAIN), NULL, Main_Proc);

只要复制一份 MainDlg.c 和 MainDlg.h 改一下就行了

DialogBox(hInstance, MAKEINTRESOURCE(IDD_MYDLG), NULL, Dlg1_Proc);

怎么传递参数？使用 DialogBoxParam 函数，对比差别。

传入 int；传入字符串；传入结构体。

传出。

怎样判断按下的按钮？通过返回值配合

怎么样在对话框关闭的时候把参数回传回去？SetWindowLong 设置的东西是和窗口的实例绑在一起的。什么是窗口实例？一个窗口打开两个、一个窗口打开两次(隐藏/打开、销毁/打开)。

SetWindowLong(hwnd,GWL_USERDATA,(LPARAM)param);

1、在资源中插入新的对话框，并且调整控件

2、

```
HINSTANCE hInstance = (HINSTANCE)GetWindowLong(hwnd,GWL_HINSTANCE);
DialogBox(hInstance, MAKEINTRESOURCE(IDD_LOGINDIALOG), NULL, Main_Proc);
```

3、每个对话框都有自己的一套 MainDlg.c 类似的代码，Main_Proc、Main_OnInitDialog 等等

修改 Main_Proc、Main_OnInitDialog 等的前缀为 LoginDlg_****，函数名不能重复

复制一份 LoginDlg.h

#ifndef _MAIN_H 等也要修改

DialogBoxParam

parameter:参数

#include "LoginDlg.h"

在 OnInitDialog 中处理传入参数，lParam 就是传入的参数

通过 DialogBoxParam 来传入参数，在对话框中的***_OnInitDialog 的 LPARAM lParam 来取参数。

long 类型，指针就是 long 了。

传递字符串

```
TCHAR* buff = "abcd";
DialogBoxParam(hInstance, MAKEINTRESOURCE(IDD_LOGINDIALOG), NULL,
LoginDlg_Proc,(LPARAM)buff);//LoginDlg_Proc
```

在对话框之间传字符串（指针）

```
HINSTANCE hInstance = (HINSTANCE)GetWindowLong(hwnd,GWL_HINSTANCE);
```

```
LoginData ld;
ld.userName = "yzk";
ld.password = "123456";
//TCHAR* buff = "abcd";
DialogBoxParam(hInstance, MAKEINTRESOURCE(IDD_LOGINDIALOG), NULL,
LoginDlg_Proc,(LPARAM)&ld);//LoginDlg_Proc
BOOL LoginDlg_OnInitDialog(HWND hwnd, HWND hwndFocus, LPARAM lParam)
{
LoginData* ld = (LoginData*)lParam;
SetDlgItemText(hwnd,IDC_EDITUSERNAME,ld->userName);
SetDlgItemText(hwnd,IDC_EDITPASSWORD,ld->password);
return TRUE;
}
```

DialogBoxParam 默认是阻塞运行的

可以设置与窗口句柄关联的数据，SetWindowLong 可以看做是把一些数据与窗口管理起来，“让它代为保管”

SetWindowLong、GetWindowLong 来在窗口中保存指针就可以了

只有一个文本框的对话框，有两个按钮【确定】、取消，允许对话框的使用者指定一个函数用来进行校验，能够校验值的正确性，如果不正确还会显示错误信息。提示：函数指针。

EndDialog(hwnd, 1);

在调用代码中通过 DialogBoxParam 的返回值就可以得到 EndDialog 设定的参数

不要重复同样的错误。不要忘了 break。通过调试功能发现的问题。在怀疑的地方加断点调试

```
LBN_DBLCLK:DoubleClick
if(LBN_DBLCLK==codeNotify)
{
//MessageBox(hwnd,TEXT("双击"),TEXT(""),MB_OK);

//LB_GETCURSEL 消息
int index = SendDlgItemMessage(hwnd,IDC_LIST1,LB_GETCURSEL,0,0);
TCHAR buff[255];
SendDlgItemMessage(hwnd,IDC_LIST1,LB_GETTEXT,index,(LPARAM)buff);
MessageBox(hwnd,buff,TEXT(""),MB_OK);
```

```
}
```

课后作业：允许用户定制校验策略的 InputDialog

2、

Codenotify=LBN_DBLCLK 用户双击列表框中的字符串。

```
if(LBN_DBLCLK==codeNotify)
{
int index = SendDlgItemMessage(hwnd,IDC_LIST1,LB_GETCURSEL,0,0);
TCHAR buff[255];
SendDlgItemMessage(hwnd,IDC_LIST1,LB_GETTEXT,index,(LPARAM)buff);
MessageBox(hwnd,buff,TEXT(""),MB_OK);
}
```

3、

LBN_SELCHANGE

http://msdn.microsoft.com/en-us/library/bb773169(VS.85).aspx