

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验六 WIN32编程**

**实验时间： 2017-11.21/11.28，18：30-21：50 实验地点： 南一楼804室6号实验台**

**指导教师： 许向阳**

**专业班级：信息安全 201601班**

**学 号： U201614804 姓 名： 黄永恒**

**同组学生： 无 报告日期： 2017年 11 月 29日**

**原创性声明**

  本人郑重声明：本报告的内容由本人独立完成，有关观点、方法、数据和文献等的引用已经在文中指出。除文中已经注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品或成果，不存在剽窃、抄袭行为。

特此声明！

学生签名：

日期：

成绩评定

|  |  |  |
| --- | --- | --- |
| 实验完成质量得分（70分）（实验步骤清晰详细深入，实验记录真实完整等） | 报告撰写质量得分（30分）（报告规范、完整、通顺、详实等） | 总成绩（100分） |
|  |  |  |

指导教师签字：

日期：

目 录

[1 实验目的与要求 1](#_Toc500541478)

[2 实验内容 1](#_Toc500541479)

[3 实验过程 2](#_Toc500541480)

[3.1 任务一 2](#_Toc500541481)

[3.1.1 实验步骤 2](#_Toc500541482)

[3.1.2 实验记录与分析 3](#_Toc500541483)

[3.2 任务二 6](#_Toc500541484)

[3.2.1 设计思想及存储单元分配 6](#_Toc500541485)

[3.2.2 流程图 6](#_Toc500541486)

[3.2.3 源程序 8](#_Toc500541487)

[3.2.4 实验步骤 13](#_Toc500541488)

[3.2.5 实验数据记录与分析 13](#_Toc500541489)

[4 总结与体会 14](#_Toc500541490)

[参考文献 16](#_Toc500541491)

# 实验目的与要求

1. 熟悉WIN32程序的设计和调试方法；
2. 熟悉宏汇编语言中INVOKE、结构变量、简化段定义等功能；
3. 进一步理解机器语言、汇编语言、高级语言之间以及实方式、保护方式之间的一些关系。

# 实验内容

**任务1. Win32程序开发奠基**

熟悉Win32程序的汇编、链接和调试；熟悉Win32程序的结构和执行过程。

1. 安装MASM32软件包，观察MASM32软件包目录结构和环境配置
2. 学习win32程序的编译和链接方法。
3. 学习使用 VS2013。
4. 阅读cbox的源程序，了解程序代码的特点和执行流程，体会基于窗口的应用程序所包含的四个部分之间的衔接关系。
5. 程序调试。观察程序模块间的调用关系；观察 Invoke等语句翻译成机器码后的结果、以及执行相应指令时堆栈的变化；观察执行一些指令前后，内存以及寄存器的变化；观察结构变量中数据的存放形式。

**任务2. Win32编程**

编写一个基于窗口的WIN32程序，实现学生成绩表信息的平均值计算及显示功能（借鉴前面实验中的一些做法），具体要求如下描述。

功能一：编写一个基于窗口的WIN32程序的菜单框架，具有以下的下拉菜单项：

File Action Help

Exit Average About

List

点菜单File下的Exit选项时结束程序；点菜单Help下的选项About，弹出一个消息框，显示本人信息，类似图1所示。点菜单Action下的选项Average、List将分别实现计算平均值或显示所有成绩的功能（详见功能二的描述）。



图1 菜单示例

功能二：每个学生的相关信息包括：姓名（结尾含1个以上的数值0，共占10个字节），语文成绩（1个字节），数学成绩（1个字节），英语成绩（一个字节），平均成绩（1个字节），等级（1个字节）。要求采用结构变量存放学生的相关信息。学生人数至少5人。姓名和各科成绩直接在数据段中给定，不必运行时输入。成绩表中最后一个学生必须使用自己的姓名。

（1） 点菜单项Average时，计算平均成绩并给出等级（等级的定义见实验一，但这里不用单独显示等级）。平均成绩的计算仍按照实验一的公式进行。平均成绩和等级保存到上述结构变量的相应字段中。

（2） 点菜单项List时，要求能在窗口中列出所有学生信息，包括姓名、各科成绩、平均成绩、等级等。如图2所示。平均成绩尚未计算时，平均成绩及等级显示为空白（调用API函数：TextOut）。

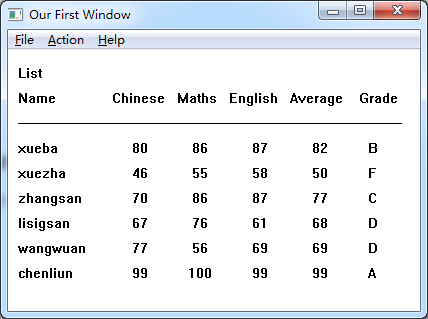


图2 成绩单显示示意图

**选做题.**

1. 增加程序的密码保护功能

在开始运行任务2的程序时，出现对话框，要求输入密码。若输入的密码错误，则给出提示信息，并等待再次输入密码。输入密码正确后，可正常执行原有功能。

1. 密码破解

对上面的程序，在只有执行程序的情况下，找出密码。或者暴力修改执行程序，绕过对密码的检查。

# 实验过程

## 任务一

### 实验步骤

1. 点击Install在D盘安装MASM32软件包，观察MASM32软件包目录结构和环境配置。
2. 打开Cmd窗口，编译链接win32程序cbox.asm。
3. 下载VS2017，调试win32程序。
4. 阅读cbox的源程序，观察程序代码的特点和执行流程，以及基于窗口的应用程序所包含的四个部分之间的衔接关系。
5. 调试程序，观察程序模块间的调用关系；观察 Invoke等语句翻译成机器码后的结果、以及执行相应指令时堆栈的变化；观察执行一些指令前后，内存以及寄存器的变化；观察结构变量中数据的存放形式。

### 实验记录与分析

1. 安装MASM32软件包，软件包目录结构如图3.1.1所示

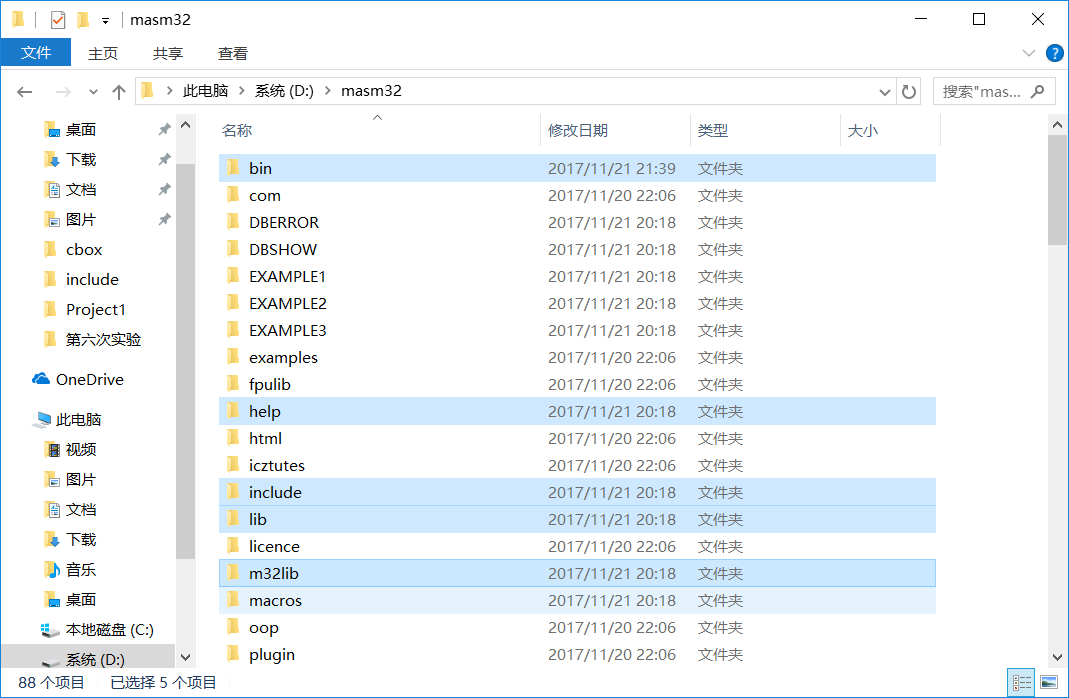


图3.1.1 MASM32软件包目录

软件包环境配置如图3.1.2所示

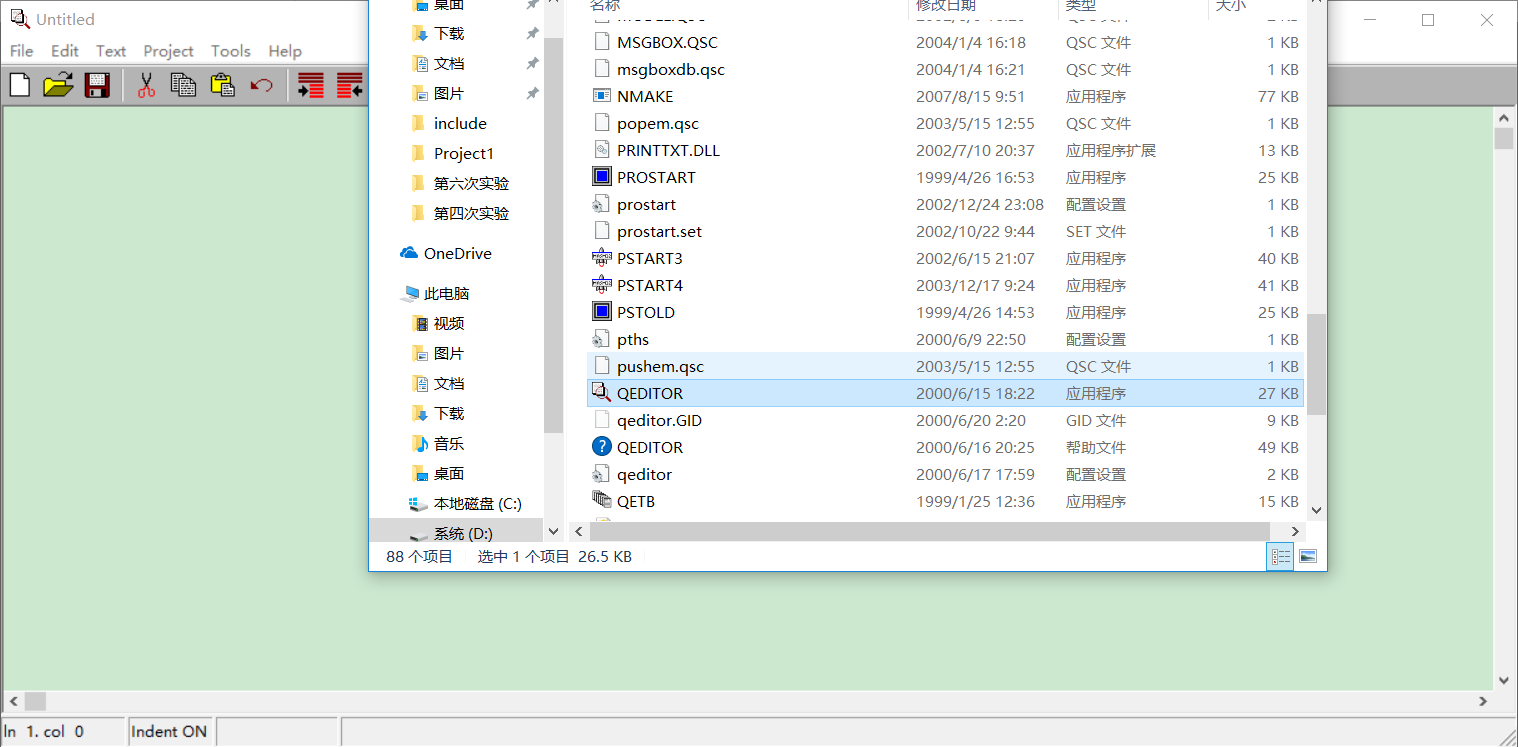


图3.1.2 MASM软件包环境配置

1. 打开CMD，编译连接win32程序，如图3.1.3所示

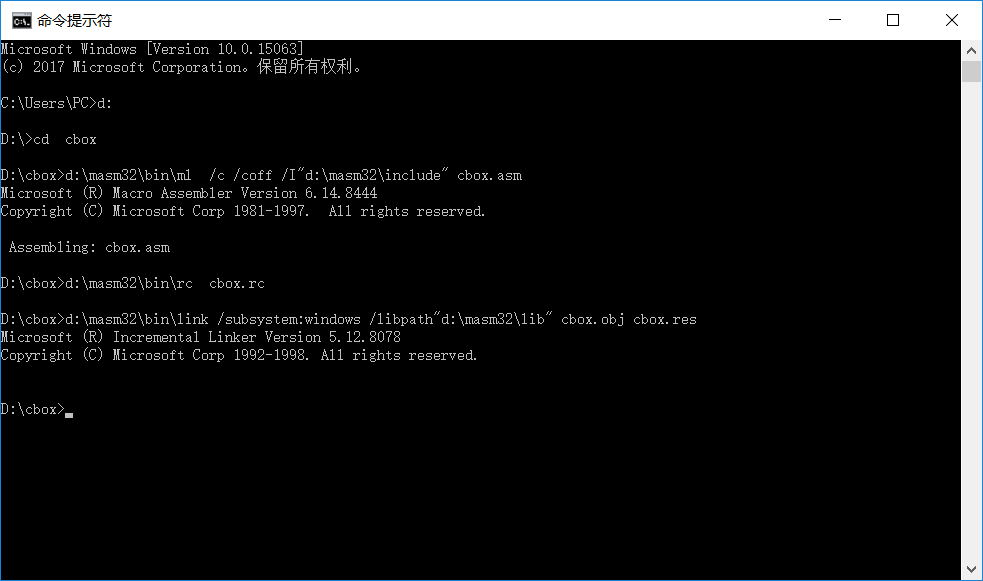


图3.1.3 编译链接win32程序

1. 阅读cbox的源程序，代码中使用了类似C语言的if、elseif等语句，并且使用invoke语句，方便调用子程序和传参。执行流程是获取句柄，调用窗口主程序，在窗口主程序中注册窗口，创建窗口，获取消息，转换消息，分发到消息处理程序，在消息处理程序中根据消息调用相应的用户处理程序，处理完后，再循环获取消息，处理消息。如获取消息是退出，则退出窗口主程序，在主程序中退出。
2. 程序调试，观察到invoke语句翻译的机器码，其中参数从右到左依次入栈，如图3.1.4所示



图3.1.4 invoke语句翻译的机器码

执行lea eax，[msg]后，eax发生变化，如图3.1.5所示



图3.1.5 执行指令后寄存器变化

执行完pop wc.hInstance后，hInstance发生变化如图3.1.6所示

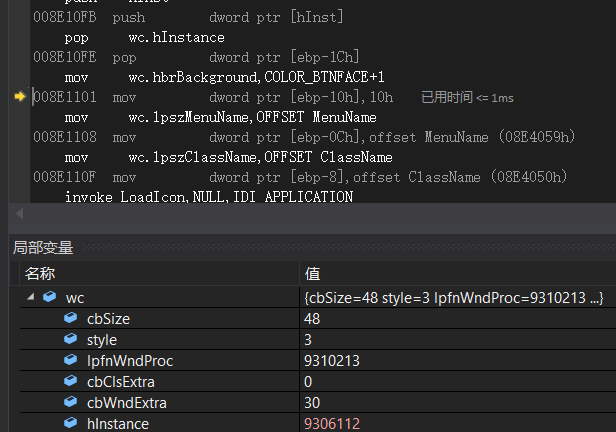


图3.1.6 执行指令后内存变化

结构变量中数据存放如图3.1.7所示

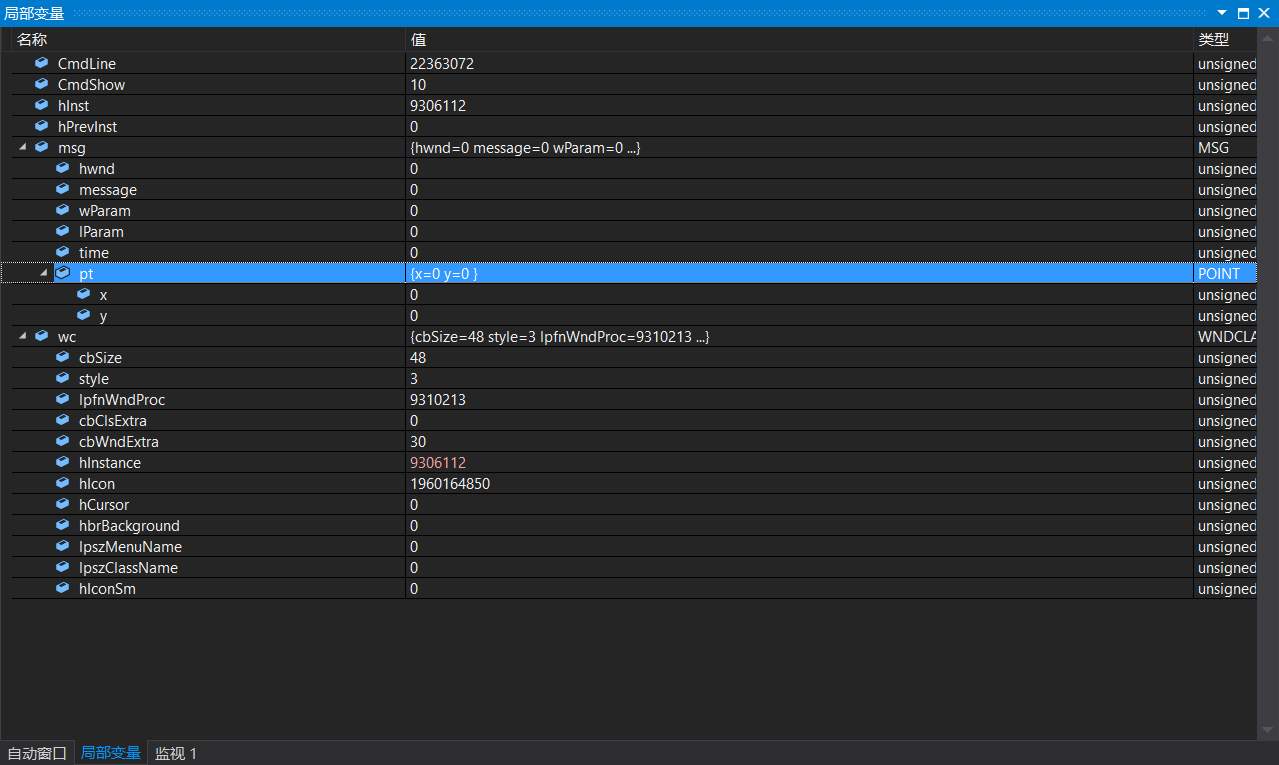


图3.1.7 结构变量中数据存放形式

由于我的VS2017缺少一些文件，无法观察调用堆栈中的信息，但是在调用子程序时，可以看到push指令，在退出子程序时可以看到pop指令以及ret指令，因此，可以想象到堆栈区的变化。

## 任务二

### 设计思想及存储单元分配

1. 主程序

设计思想：首先获取本次向在主存中的句柄、获取命令行参数的地址，再调用窗口主程序，最后退出主程序。

寄存器分配：

EAX：存放句柄、地址、以及返回值。

存储单元分配：

szClassName：字节变量，作为类名

szDisplayName：字节变量，存储窗口标题

AboutMsg：字节变量，存储提示信息

hInstance：双字变量，存储实例句柄

CommandLine：双字变量，存储命令行地址

hWnd：双字变量，存储窗口句柄

buf：结构变量，存储学生信息

msg\_xxx：字节变量，存储字符串，输出时用

flag：字节变量，作为是否计算平均值的标志

1. 窗口主程序

设计思想：首先注册窗口类，再建立窗口，根据窗口获取的消息，分发到窗口消息处理程序，如果是退出消息，则退出，否则无限获取消息。

存储单元分配：

wc：结构变量，作为WNDCLASSEX

msg：结构变量，作为MSG

Wwd：双字变量，存储窗口宽度

Whd：双字变量，存储窗口高度

Wtx：双字变量，存储窗口位置X

Wty：双字变量，存储窗口位置Y

1. 窗口消息处理程序

设计思想：首先判断消息是否是菜单命令产生的消息，如果是就判断是哪个菜单，根据标志码调用相应的处理程序，然后返回；如果不是判断是否是退出命令，如果是就发送退出消息，返回；如果不是就搁置，直接返回。

1. 用户处理程序

设计思想：求平均值程序可以参考之前的子程序，输出成绩程序中需要判断是否以及计算了平均值，因此可以考虑加一个Flag标志，若已计算，则输出平均值和等级，若无，则只输出语数英成绩。

### 流程图

主程序流程图如图3.2.1所示

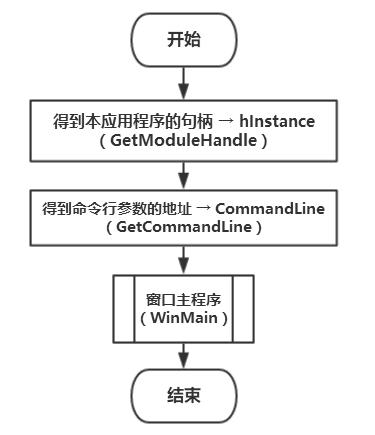


图3.2.1 主程序流程图

窗口主程序流程图如图3.2.2所示

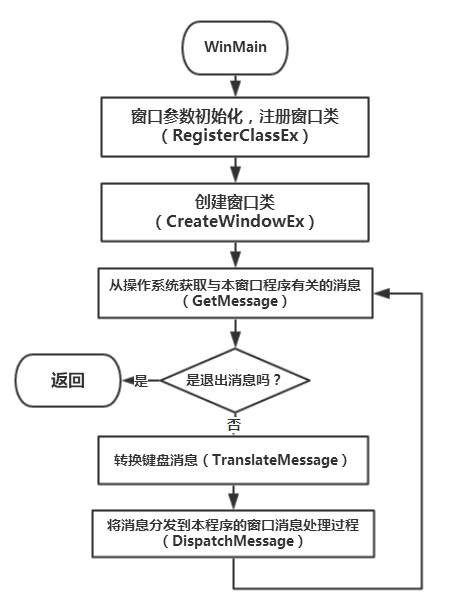


图3.2.2 窗口主程序流程图

窗口处理程序WndProc的流程图如图3.2.1所示

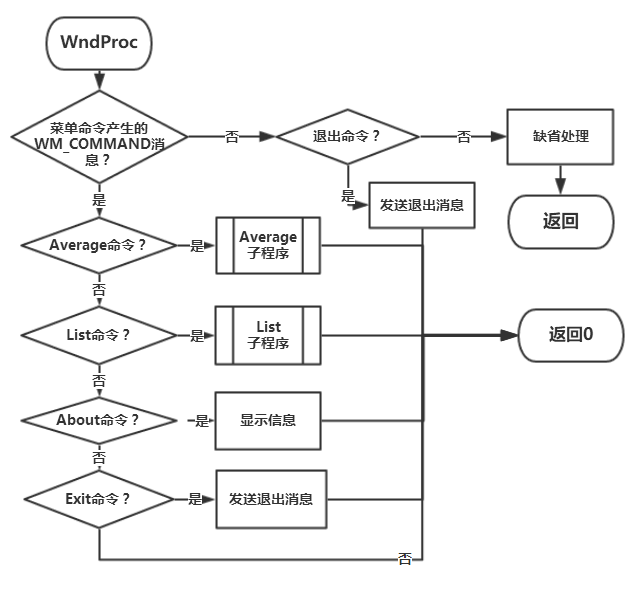


图3.2.1 WndProc程序流程图

### 源程序

.rc文件

#include "resource.h"

#if !defined(AFX\_RESOURCE\_DLL) || defined(AFX\_TARG\_CHS)

LANGUAGE LANG\_CHINESE, SUBLANG\_CHINESE\_SIMPLIFIED

#pragma code\_page(936)

// Menu

600 MENU

BEGIN

POPUP "&File"

BEGIN

MENUITEM "&Exit", 1010

END

POPUP "&Action"

BEGIN

MENUITEM "&Average", 1800

MENUITEM "&List", 1801

END

POPUP "&Help"

BEGIN

MENUITEM "&About", 1900

END

END

.asm文件

.386

.MODEL FLAT, STDCALL

OPTION CASEMAP: NONE

include windows.inc

include user32.inc

include kernel32.inc

include gdi32.inc

include shell32.inc

includelib user32.lib

includelib kernel32.lib

includelib gdi32.lib

includelib shell32.lib

WinMain proto :DWORD,:DWORD,:DWORD,:DWORD

WndProc proto :DWORD,:DWORD,:DWORD,:DWORD

List proto :DWORD

Average proto :DWORD

student struct

myname db 10 dup(0)

chinese db 0

math db 0

english db 0

average db 0

grade db 0

student ends

;------------------------------------------

.data

szClassName db 'MainWndClass',0

szDisplayName db 'My first Window',0 ;窗口名称

MenuName db 'MyMenu',0

DlgName db 'MyDialog',0

AboutMsg db '我是IS1601黄永恒',0 ;提示窗内容

hInstance dd 0

CommandLine dd 0

hWnd dd 0

buf student <'zhaojiu',89,75,88,0,0> ;学生结构集合

student <'zhangsan',60,85,46,0,0>

student <'lisi',95,90,95,0,0>

student <'wangwu',80,65,85,0,0>

student <'yongheng',99,98,97,0,0>

msg\_list db 'List',0

msg\_name db 'Name',0

msg\_chinese db 'Chinese',0

msg\_math db 'Math',0

msg\_english db 'English',0

msg\_average db 'Average',0

msg\_grade db 'Grade',0

msg\_line db '\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_',0

chinese db '89','60','95','80', '99' ;便于输出的学生成绩

math db '75','85','90','65', '98'

english db '88','46','95','85', '97'

average db '00','00','00','00', '00'

flag db 0

N EQU 5

.code

;-----------------------------主程序--------------------------------------

start:

invoke GetModuleHandle ,NULL

mov hInstance, EAX

invoke GetCommandLine

mov CommandLine ,EAX

invoke WinMain,hInstance ,NULL ,CommandLine ,SW\_SHOWDEFAULT

invoke ExitProcess, EAX

;---------------------------窗口主程序-----------------------------------

WinMain proc hInst :DWORD, hPrevInst :DWORD,

CmdLine :DWORD, CmdShow :DWORD

local wc :WNDCLASSEX

local msg :MSG

local Wwd :DWORD

local Wht :DWORD

local Wtx :DWORD

local Wty :DWORD

mov wc.cbSize, sizeof WNDCLASSEX

mov wc.style, CS\_HREDRAW or CS\_VREDRAW

mov wc.lpfnWndProc, offset WndProc

mov wc.cbClsExtra, NULL

mov wc.cbWndExtra, NULL

push hInst

pop wc.hInstance

mov wc.hbrBackground, COLOR\_WINDOW+1

mov wc.lpszMenuName, offset MenuName

mov wc.lpszClassName, offset szClassName

mov wc.hIcon, 0

mov wc.hIconSm, 0

invoke LoadCursor, NULL,IDC\_ARROW

mov wc.hCursor, eax

invoke RegisterClassEx, addr wc ;注册窗口类

mov Wwd, 600

mov Wht, 400

mov Wtx, 100

mov Wty, 100

invoke CreateWindowEx, WS\_EX\_ACCEPTFILES + WS\_EX\_APPWINDOW,

addr szClassName, addr szDisplayName,WS\_OVERLAPPEDWINDOW+ WS\_VISIBLE,Wtx, Wty, Wwd, Wht,

NULL, NULL, hInst, NULL ;建立窗口

mov hWnd,EAX

invoke LoadMenu, hInst,600

invoke SetMenu,hWnd, EAX

StartLoop:

invoke GetMessage,addr msg,NULL,0,0 ;获取消息

cmp EAX,0

je ExitLoop

invoke TranslateMessage,addr msg ;转换消息

invoke DispatchMessage,addr msg ;分发消息

jmp StartLoop

ExitLoop:

mov EAX , msg.wParam ;设置退出码

Ret

WinMain ENDP

;---------------------窗口消息处理程序----------------------------

WndProc proc hWin :DWORD, uMsg :DWORD, wParam:DWORD, lParam:DWORD

.if uMsg == WM\_COMMAND

.if wParam == 1800

invoke Average,N ;调用平均值子程序

.elseif wParam == 1801

invoke List,hWin ;调用展示子程序

.elseif wParam == 1900 ;help-about信息展示

invoke MessageBox, hWin, addr AboutMsg, addr szDisplayName,MB\_OK

.elseif wParam == 1010

invoke SendMessage,hWin,WM\_CLOSE,0,0

.endif

.elseif uMsg == WM\_DESTROY

invoke PostQuitMessage,NULL

.else

invoke DefWindowProc,hWin,uMsg,wParam,lParam

ret

.endif

mov EAX,0

ret

WndProc endp

;-----------------------用户处理程序------------------------------------

;子程序名：Average

;功能：计算加权平均值

;原型：Average PROTO ：DWORD

; buf：结构变量 ;学生信息

Average proc number:DWORD

pusha

mov flag,1

mov ecx, 0

\_next: cmp ecx, number

jge \_exit

xor eax, eax

xor edx, edx

imul ebx, ecx, 15

mov al, buf[ebx].chinese

mov dl, buf[ebx].math

sal eax,2

sal edx,1

add eax,edx

mov edx,0

mov dl, buf[ebx].english

add ax, dx

mov dl,7

div dl

;判断等级

.if al > 90

mov buf[ebx].grade, 'A'

.elseif al > 80

mov buf[ebx].grade, 'B'

.elseif al > 70

mov buf[ebx].grade, 'C'

.elseif al > 60

mov buf[ebx].grade, 'D'

.else

mov buf[ebx].grade, 'F'

.endif

mov buf[ebx].average, al

call F2T10 ;将平均成绩转换为10进制数字串

inc ecx

jmp \_next

\_exit: popa

ret

Average ENDP

;---------------------------------------------------------------------

;子程序名：List

;功能：输出成绩表

;原型：List PROTO ：DWORD

List proc hWin:DWORD

XX equ 10

YY equ 10

XX\_GAP equ 100

YY\_GAP equ 30

local hdc :HDC

invoke GetDC, hWin

mov hdc, eax

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+0\*YY\_GAP,offset msg\_list,4

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_name,4

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_chinese,7

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_math,4

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_english,7

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_average,7

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+1\*YY\_GAP,offset msg\_grade,5

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+2\*YY\_GAP,offsetmsg\_line,68

;输出各个学生的成绩

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+3\*YY\_GAP,offset buf[0\*15].myname,7

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+3\*YY\_GAP,offset chinese,2

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+3\*YY\_GAP,offset math,2

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+3\*YY\_GAP,offset english,2

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+4\*YY\_GAP,offset buf[1\*15].myname,8

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+4\*YY\_GAP,offset chinese+2,2

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+4\*YY\_GAP,offset math+2, 2

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+4\*YY\_GAP,offset english+2,2

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+5\*YY\_GAP,offset buf[2\*15].myname,4

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+5\*YY\_GAP,offset chinese+4,2

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+5\*YY\_GAP,offset math+4, 2

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+5\*YY\_GAP,offset english+4,2

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+6\*YY\_GAP,offset buf[3\*15].myname,6

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+6\*YY\_GAP,offset chinese+6,2

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+6\*YY\_GAP,offset math+6,2

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+6\*YY\_GAP,offset english+6,2

invoke TextOut,hdc,XX+0\*XX\_GAP,YY+7\*YY\_GAP,offset buf[4\*15].myname,8

invoke TextOut,hdc,XX+1\*XX\_GAP,YY+7\*YY\_GAP,offset chinese+8,2

invoke TextOut,hdc,XX+2\*XX\_GAP,YY+7\*YY\_GAP,offset math+8,2

invoke TextOut,hdc,XX+3\*XX\_GAP,YY+7\*YY\_GAP,offset english+8,2

;根据flag判断是否平均值等级是否已计算

cmp flag,0

je exit

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+3\*YY\_GAP,offset average,2

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+3\*YY\_GAP,offset buf[0\*15].grade,1

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+4\*YY\_GAP,offset average+2,2

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+4\*YY\_GAP,offset buf[1\*15].grade,1

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+5\*YY\_GAP,offset average+4,2

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+5\*YY\_GAP,offset buf[2\*15].grade,1

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+6\*YY\_GAP,offset average+6,2

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+6\*YY\_GAP,offset buf[3\*15].grade,1

invoke TextOut,hdc,XX+4\*XX\_GAP,YY+7\*YY\_GAP,offset average+8,2

invoke TextOut,hdc,XX+5\*XX\_GAP,YY+7\*YY\_GAP,offset buf[4\*15].grade,1

exit: ret

List endp

;-----------------------------------------------------------------------

;子程序名：F2T10

;功能：将二进制数转换为十进制数字串

;参数：ax：二进制数

; average：数字串首址

F2T10 proc

pusha

mov ah, 0

mov dx, 10

idiv dl

mov average[ecx\*2+1], ah

add average[ecx\*2+1], '0'

mov ah,0

idiv dl

mov average[ecx\*2], ah

add average[ecx\*2], '0'

popa

ret

F2T10 ENDP

end start

### 实验步骤

* 1. 准备上机环境，编辑、汇编、连接文件。
  2. 运行程序，点击Help-About，观察是否弹出提示信息。
  3. 点击 Action-List，观察是否显示语数英成绩。
  4. 点击Action-Average，计算平均值，再点击Action-List，观察是否输出成绩和等级。
  5. 点击File-Exit，观察是否退出程序。

### 实验数据记录与分析

1. 实验环境条件：i7-5500U 3.0GHz，8G内存；WINDOWS 10下 VS2017
2. 运行该程序，点击Help-About，显示信息，如图3.2.4所示

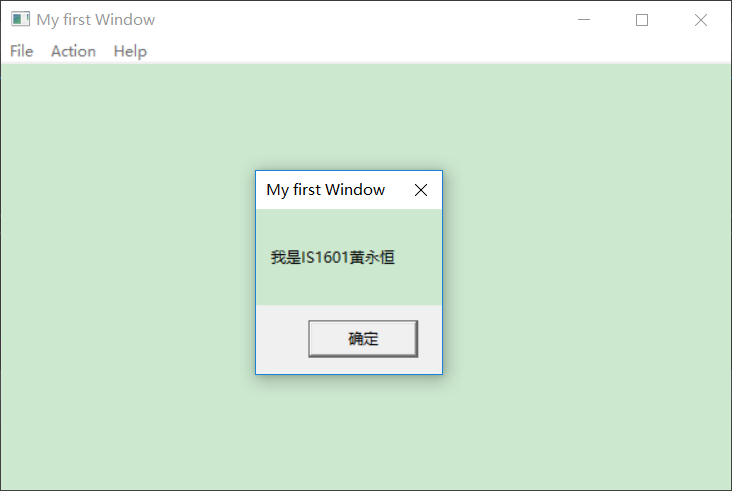


图3.2.4 点击Help-About，显示信息

1. 点击Action-List，如图3.2.5所示

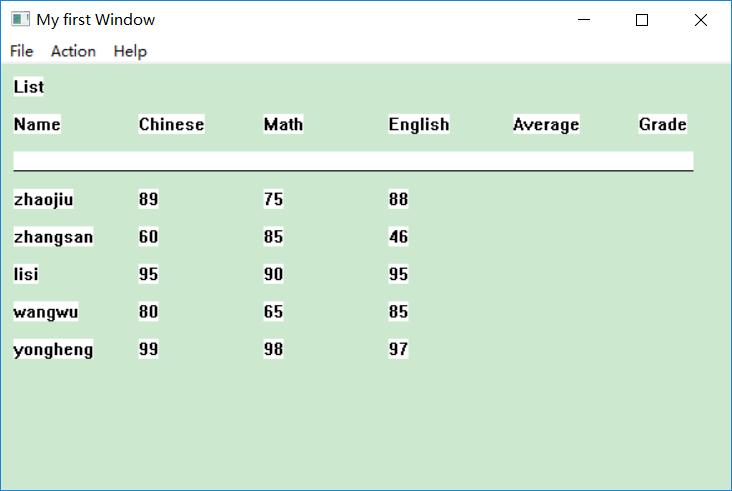


图3.2.5 点击Action-List，显示语数英成绩

1. 点击Action-Average，计算平均值，再点击Action-List，如图3.2.6所示

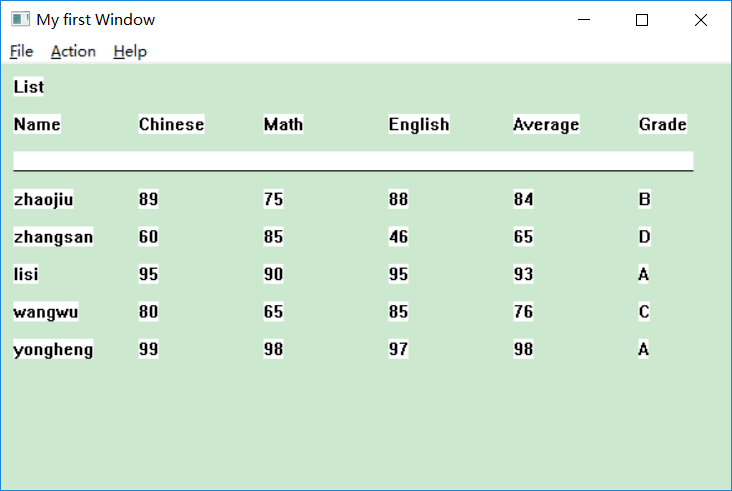


图3.2.6 计算平均值后点击Action-List，显示成绩和等级

1. 点击File-Exit，退出程序。

# 总结与体会

本次实验是汇编最后一次实验，对我来说，也是最难的一个实验，我觉得主要原因是上课根本没有听，而且win32编程与dos编程不太一样，初次接触，学习，我感到很迷惑。

但是不管怎么样，我求助同学、网络，最终还是完成了这次实验。

在任务一中，我知道了如何MASM32软件包的目录结构，这些文件在win32编程中起到很方便的作用，通过老师发的文件，我学会了如何编译连接win32程序，如何使用VS系列调试win32程序，通过自己调试，我体会到invoke指令的方便，了解了它的传参方式，这些让我有了一些win32编程基础。

在任务二中我编写了首个win32程序，刚开始错误百出，然后我在网上一一查找错误解决方法，解决不了的问同学，最终顺利完成。

在附加题中，也是应用中断程序驻留内存相关知识，以及复原中断处理程序、

汇编实验已经结束，然而我发现自己还是不熟练，每次应用新知识的时候都有一定障碍，应该是自己的理解能力比较弱导致的，之后还得训练！

# 参考文献

[1] 许向阳，《80X86汇编语言程序设计上机指南》“第七章 MASM32环境”、“第八章 一个文本编辑器”、“第九章反汇编及调试工具W32Dasm”。华中科技大学出版社。

[2] 汇编语言教学网站（http://115.156.187.251/huibian1/site/index.jsp）-》资料下载-》案例-》win32程序、编译和连接。

[3] 汇编语言教学网站-》资料下载-》书籍-》Win32汇编程序的源码级调试

其中的操作说明，给出了几种编译和链接生成执行程序的方法。

[4] 罗云彬，《Windows环境下32位汇编语言设计》. 电子工业出版社。

[5] MSDN（Microsoft Developer Network）， 有关 Windows API 的帮助。