

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： WIN32编程**

**实验时间： 2018-5-18，18:30-21:50 实验地点： 南一楼804室16号实验台**

**指导教师： 李海波 专业班级：计算机科学与技术201607班**

**学 号： U201614700 姓 名： 王亚宁**

**同组学生： 无 报告日期： 2018年5月18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**

[实验目的与要求 1](#_Toc514606606)

[实验内容 1](#_Toc514606607)

[实验过程 2](#_Toc514606608)

[任务1 2](#_Toc514606609)

[1. 设计思想及存储单元分配 2](#_Toc514606610)

[2. 流程图 2](#_Toc514606611)

[3. 源程序 2](#_Toc514606612)

[4. 实验步骤 2](#_Toc514606613)

[5. 实验记录 3](#_Toc514606614)

[任务2 3](#_Toc514606615)

[1.设计思想及存储单元分配 3](#_Toc514606616)

[2.流程图 3](#_Toc514606617)

[3.源代码 3](#_Toc514606618)

[4.实验步骤 8](#_Toc514606619)

[5.实验记录 8](#_Toc514606620)

[体会 9](#_Toc514606621)

[参考文献 10](#_Toc514606622)

# 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

编写一个基于窗口的WIN32程序，实现**网店商品信息管理程序**的平均利润率计算及商品信息显示的功能（借鉴实验三的一些做法），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5.1所示。点菜单Action下的选项Average、List将分别实现计算平均利润率或显示SHOP1所有商品信息的功能（详见功能二的描述）。



图5.1 菜单示例

功能二：要求采用结构变量存放商品的相关信息。商品数至少定义5种。

点菜单项Average时，按照实验三的方法计算所有商品的平均利润率。用TD32观察计算结果。

点菜单项List时，要求能在窗口中列出SHOP1的所有商品的信息。具体显示格式自行定义，可以参照图5.2的样式（不要求用中文）。



图5.2 商品信息显示示意图

# 实验过程

## 任务1

### 1. 设计思想及存储单元分配

菜单设计不涉及编译和连接的操作，只需要编写资源文件就行了。要做到题目中要求的样式，需要3个弹出菜单，第一个弹出菜单File下面只有一个选项Exit，第二个弹出菜单Action下有两个选项Average和List，第三个弹出菜单Help下有一个选项是About显示程序信息。

### 2. 流程图

无

### 3. 源程序

menu.rc文件的内容如下：

#define File\_Exit 10001

#define Action\_Average 10101

#define Action\_List 10102

#define Help\_About 10201

MyMenu MENU

BEGIN

POPUP "&File"

BEGIN

MENUITEM "E&xit", File\_Exit

END

POPUP "&Action"

BEGIN

MENUITEM "A&verage", Action\_Average

MENUITEM "L&ist", Action\_List

END

POPUP "&Help"

BEGIN

MENUITEM "A&bout",Help\_About

END

END

menuID.inc文件内容如下：

File\_Exit equ 10001

Action\_Average equ 10101

Action\_List equ 10102

Help\_About equ 10201

### 4. 实验步骤

要得到资源文件只需要对文件使用命令行编译即可。

本机系统信息：Windows 10 1803 64bit，编译环境为：

rc.exe: Microsoft (R) Windows (R) Resource Compiler, Version 5.00.1823.1 - Build 1823

ml.exe: Microsoft (R) Macro Assembler Version 6.14.8444

link.exe: Microsoft (R) Incremental Linker Version 5.12.8078

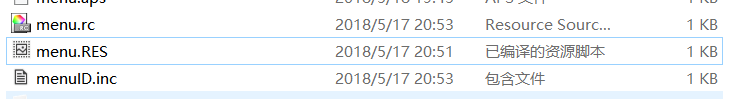
td32.exe: Turbo Debugger 32-bit Version 5.0 Copyright (c) 1988,96 Borland International

### 5. 实验记录

打开文件所在目录，执行命令：

rc ./menu.rc

生成一个.res文件，如图：



## 任务2

### 1.设计思想及存储单元分配

实验不算太难，使用MSDN可以搜索到使用的各函数的接口声明，对调用有很大帮助，实验的代码也是在给出的代码基础上修改的，因此实现的并不算很曲折。环境配置好，使用power shell运行给定文档里的指令也成功的显示出了窗口，根据自己的需求修改好代码和函数就行了。

### 2.流程图

无

### 3.源代码

.386

.model flat,stdcall

option casemap:none

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

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

Display proto :DWORD

include menuID.inc; 自己的ID文件

; 系统inc文件

include windows.inc

include user32.inc

include kernel32.inc

include gdi32.inc

include shell32.inc

; 系统库文件

includelib user32.lib

includelib kernel32.lib

includelib gdi32.lib

includelib shell32.lib

;结构体定义

; 大小为24个字节

goods struct

g\_name db 10 dup(0)

inpre dw 0

outpre dw 0

incnt dw 0

outcnt dw 0

pro dw 0

len dd 0

goods ends

; 数据段定义

.data

ClassName db 'TryWinClass',0

AppName db 'Shop Manage System',0

MenuName db 'MyMenu',0

DlgName db 'MyDialog',0

AboutMsg db 'I am Yaning Wang from CS1607.',0

hInstance dd 0

CommandLine dd 0

all\_goods goods <'Bag', 12, 30, 100, 5, ?, 3>

goods <'Pen', 35, 56, 70, 25, ?, 3>

goods <'Book', 12, 30, 25, 5, ?, 4>

goods <'Cup', 35, 50, 30, 24, ?, 3>

goods <'Eraser', 12, 28, 20, 15, ?, 6>

msg\_name db 'Name',0

msg\_inpre db 'In Price',0

msg\_outpre db 'Out Price',0

msg\_incnt db 'In Count',0

msg\_outcnt db 'Out Count',0

msg\_pro db 'Profile(%)',0

sum db 0

;tostr

num\_str db 10 dup(0)

len dd 0

ten dw 10

; display

xgap dd 0

ygap dd 0

g\_idx dd 0

; 代码段

.code

Start: invoke GetModuleHandle,NULL

mov hInstance,eax

invoke GetCommandLine

mov CommandLine,eax

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

invoke ExitProcess, eax

;;

WinMain proc hInst:DWORD, hPrevInst:DWORD, CmdLine:DWORD, CmdShow:DWORD

LOCAL wc:WNDCLASSEX

LOCAL msg:MSG

LOCAL hWnd:HWND

invoke RtlZeroMemory, addr wc, sizeof wc

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 ClassName

invoke LoadIcon, NULL,IDI\_APPLICATION

mov wc.hIcon, eax

mov wc.hIconSm, 0

invoke LoadCursor, NULL, IDC\_ARROW

mov wc.hCursor, eax

invoke RegisterClassEx, addr wc

invoke CreateWindowEx, NULL,addr ClassName,addr AppName,\

WS\_OVERLAPPEDWINDOW,CW\_USEDEFAULT,\

CW\_USEDEFAULT,CW\_USEDEFAULT,CW\_USEDEFAULT,NULL,NULL,\

hInst,NULL

mov hWnd, eax

invoke ShowWindow, hWnd, SW\_SHOWNORMAL

invoke UpdateWindow, hWnd

;主循环

MsgLoop:

invoke GetMessage, addr msg, NULL, 0, 0

cmp eax, 0

je ExitLoop

invoke TranslateMessage, addr msg

invoke DispatchMessage, addr msg

jmp MsgLoop

ExitLoop:

; 结束程序

mov eax,msg.wParam

ret

WinMain endp

; 计算所有商品的利润率

Average proc hWnd:dword

mov sum, 5

push eax

push ebx

push ecx

push edx

push edi

mov ebx, 0

LOOP\_AVE:

movsx eax, all\_goods[ebx].inpre

movsx ecx, all\_goods[ebx].incnt

imul ecx

mov edi, eax

movsx eax, all\_goods[ebx].outpre

movsx ecx, all\_goods[ebx].outcnt

imul ecx

sub eax, edi

imul eax, 100

cdq

idiv edi

mov all\_goods[ebx].pro, ax

add ebx, 24

dec sum

cmp sum, 0

jg LOOP\_AVE

pop edi

pop edx

pop ecx

pop ebx

pop eax

;invoke Display, hWnd

ret

Average endp

; 函数功能：将参数数字转换为字符串存储在num\_str字符串中，len 存储

; 返回eax作为字符串长

ToStr proc num:dword

push eax

push ebx

push ecx

push edx

mov ecx, 0

mov len, 0

mov ax, word ptr num

cmp ax, 0

mov ebx, offset num\_str

jge POSITIVE

neg ax

mov byte ptr [ebx], '-'

inc ebx

inc len

POSITIVE:

inc len

cwd

idiv ten

push dx

inc ecx

cmp ax, 0

jne POSITIVE

POP\_LOOP:

pop dx

add dl, '0'

mov [ebx], dl

inc ebx

loop POP\_LOOP

pop edx

pop ecx

pop ebx

pop eax

ret

ToStr endp

; 专门处理的程序

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

LOCAL hdc:HDC

.IF uMsg == WM\_DESTROY

invoke PostQuitMessage,NULL

.ELSEIF uMsg == WM\_COMMAND

; 按了退出

.IF wParam == File\_Exit

invoke SendMessage,hWnd,WM\_CLOSE,0,0

.ELSEIF wParam == Action\_Average

invoke Average, hWnd

.ELSEIF wParam == Action\_List

; 显示商品信息

invoke Display, hWnd

.ELSEIF wParam == Help\_About

; 显示作者信息

invoke MessageBox, hWnd, addr AboutMsg, addr AppName, MB\_OK

.ENDIF

.ELSE

invoke DefWindowProc,hWnd,uMsg,wParam,lParam

ret

.ENDIF

xor eax, eax

ret

WndProc endp

Display proc hWnd:DWORD

XX equ 10

YY equ 10

XX\_GAP equ 100

YY\_GAP equ 30

LOCAL hdc:HDC

push eax

push ebx

push ecx

push edx

push edi

xor eax, eax

xor ebx, ebx

xor ecx, ecx

xor edx, edi

invoke GetDC, hWnd

mov hdc, eax

mov ygap, YY

; 输出最上面的信息栏

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

invoke TextOut,hdc,XX+1\*XX\_GAP,ygap,offset msg\_inpre, 8

invoke TextOut,hdc,XX+2\*XX\_GAP,ygap,offset msg\_outpre, 9

invoke TextOut,hdc,XX+3\*XX\_GAP,ygap,offset msg\_incnt, 8

invoke TextOut,hdc,XX+4\*XX\_GAP,ygap,offset msg\_outcnt, 9

invoke TextOut,hdc,XX+5\*XX\_GAP,ygap,offset msg\_pro, 10

; 一共5件商品

mov sum, 5

; ebx放置偏移

mov ebx, 0

PRINT\_LOOP:

;push ecx

mov xgap, XX

add ygap, YY\_GAP

invoke TextOut,hdc,xgap,ygap,addr all\_goods[ebx].g\_name, all\_goods[ebx].len;all\_goods[ebx].len

; 下面的需要先转换为str，还要有长度

; 调用ToStr函数

add xgap, XX\_GAP

invoke ToStr, all\_goods[ebx].inpre

invoke TextOut,hdc,xgap,ygap,offset num\_str, len

add xgap, XX\_GAP

invoke ToStr, all\_goods[ebx].outpre

invoke TextOut,hdc,xgap,ygap,offset num\_str, len

add xgap, XX\_GAP

invoke ToStr, all\_goods[ebx].incnt

invoke TextOut,hdc,xgap,ygap,offset num\_str, len

add xgap, XX\_GAP

invoke ToStr, all\_goods[ebx].outcnt

invoke TextOut,hdc,xgap,ygap,offset num\_str, len

add xgap, XX\_GAP

invoke ToStr, all\_goods[ebx].pro

invoke TextOut,hdc,xgap,ygap,offset num\_str, len

add ebx, 24

dec sum

cmp sum, 0

jg PRINT\_LOOP

; 结束，恢复环境

pop edi

pop edx

pop ecx

pop ebx

pop eax

ret

Display endp

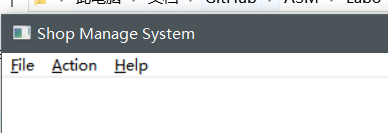
end Start

### 4.实验步骤

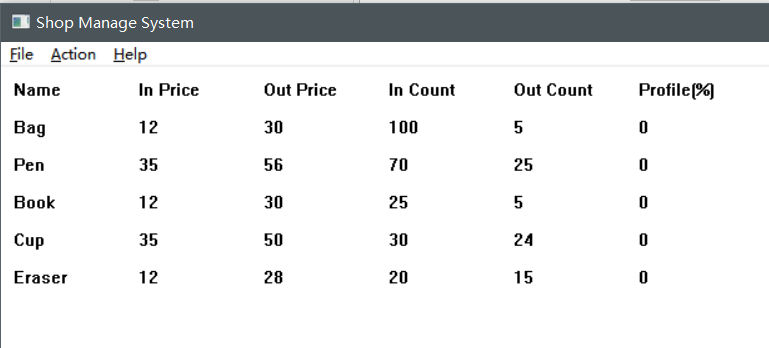
编译代码文件，连接文件，运行即可。

### 5.实验记录

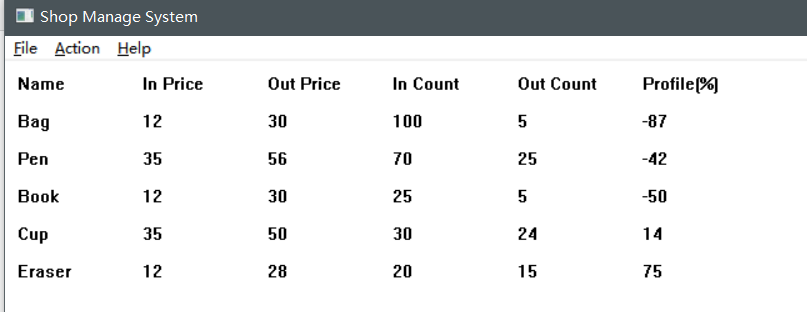
运行如图：



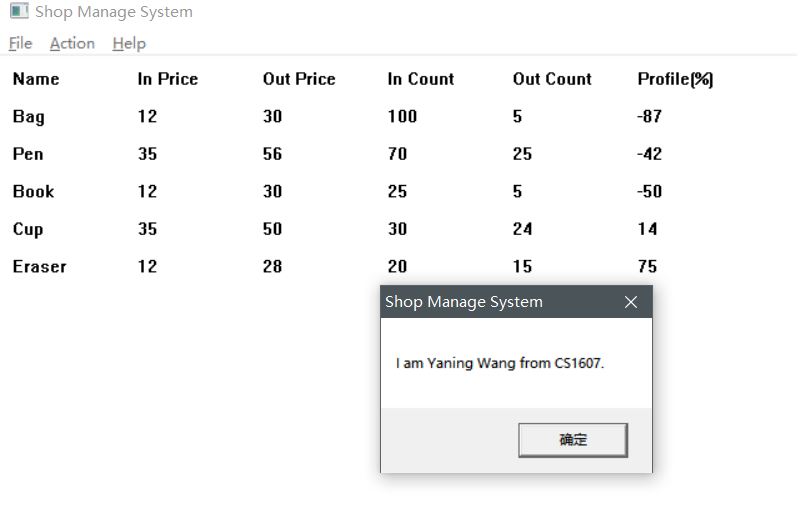
显示数据：



计算利润率后：



弹窗示例：



# 体会

本次实验的主要难点我觉得是实验的环境搭建和函数的调用。我在实验代码编写时遇到了一些问题，第一次就是调用TextOut函数，我其中的几个参数是使用寄存器传入的，而编译时报错了，上MSDN查找后知道是当使用寄存器传参时由于函数调用有可能会更改寄存器的值，要求我将其替换为其他变量，我就替换成了自定义的变量就顺利通过编译了。

还有一个地方就是结构体的使用，以前在C语言中使用过结构题，现在在汇编中使用时发现了其中的不同之处，C语言使用结构体数组时给定的是结构体的偏移量除以一个结构体的大小，而汇编里是直接给定偏移量，比C语言里要麻烦不少。

这次实验中我还遇到了addr和offset两个指令，以前都是用offset，在WIN32汇编里多了个addr，上网搜索后知道了addr是可以在运行时获取元素地址的，而offset只有是编译时获取地址。

# 参考文献

[1] 王元珍等.80x86汇编语言程序设计.版本(第1版)

[2] 王晓虹等.汇编语言程序设计教程.版本(第1版)