

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验五 WIN32编程**

**实验时间： 2018-5-14，14：00-17：30 实验地点： 南一楼**

**指导教师： 朱虹**

**专业班级： 计算机科学与技术1601班**

**学 号： U201614531 姓 名： 刘本嵩**

**同组学生： 无 报告日期： 2018年 5 月 14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](#_Toc1348863323)

[2 实验内容 2](#_Toc1735444070)

[3 实验过程 4](#_Toc618894386)

[3.1 任务1 4](#_Toc1931082360)

[3.1.1 实验步骤 4](#_Toc115390079)

[3.1.2 源程序 4](#_Toc2135420692)

[3.1.3 实验记录与分析 28](#_Toc530814034)

[4 总结与体会 32](#_Toc581862124)

[5 参考文献 33](#_Toc2133922976)

# 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）。

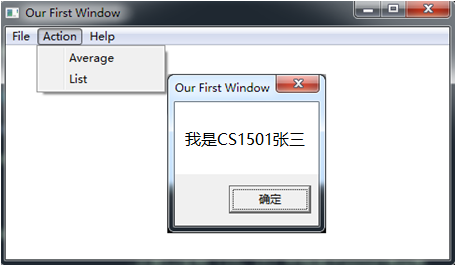


图5.1 菜单示例

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

### 实验步骤

1. 配置开发环境。

2. 调用clang编译汇编源文件impl.s, main.s。具体的命令为clang -c impl.s && clang -c main.s。窗口布局直接写死在main.s，无需资源文件，简化构建。

3. 使用ld连接生成的目标文件以及资源文件，具体的命令为 clang -o exp5.exe main.o impl.o

4. 确保以上过程正确的情况下运行exp5.exe。并且测试该程序功能正常，没有错误。

### 源程序

;impl.s;大量复用先前代码，快速实现主要功能。

.file "impl.rasm"

.intel\_syntax noprefix

.globl \_calc\_done

.bss

.align 4

\_calc\_done:

.space 4

.globl \_products

.data

.align 32

\_products:

.ascii "scala\0"

.space 26

.long 30

.long 70

.byte 63

.space 3

.ascii "java\0"

.space 27

.long 500

.long 500

.byte 63

.space 3

.ascii "golang\0"

.space 25

.long 60

.long 50

.byte 63

.space 3

.ascii "C++1z\0"

.space 26

.long 1000

.long 1100

.byte 63

.space 3

.ascii "Python\0"

.space 25

.long 10

.long 1000

.byte 63

.space 3

.ascii "Assembly\0"

.space 23

.long 100000

.long 1

.byte 63

.space 3

.text

.globl \_profit\_mark\_of

.def \_profit\_mark\_of; .scl 2; .type 32; .endef

\_profit\_mark\_of:

LFB11:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 16

fld DWORD PTR [ebp+8]

fdiv DWORD PTR [ebp+12]

fld1

fsubp st(1), st

fstp DWORD PTR [ebp-4]

fldz

fld DWORD PTR [ebp-4]

fxch st(1)

fucompp

fnstsw ax

sahf

jbe L15

mov eax, 70

jmp L4

L15:

fld DWORD PTR [ebp-4]

fld QWORD PTR LC2

fxch st(1)

fucompp

fnstsw ax

sahf

jbe L16

mov eax, 65

jmp L4

L16:

fld DWORD PTR [ebp-4]

fld DWORD PTR LC3

fxch st(1)

fucompp

fnstsw ax

sahf

jbe L17

mov eax, 66

jmp L4

L17:

fld DWORD PTR [ebp-4]

fld QWORD PTR LC4

fxch st(1)

fucompp

fnstsw ax

sahf

jbe L18

mov eax, 67

jmp L4

L18:

mov eax, 68

L4:

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE11:

.section .rdata,"dr"

LC6:

.ascii "Calculation done.\0"

.text

.globl \_calc\_average

.def \_calc\_average; .scl 2; .type 32; .endef

\_calc\_average:

LFB12:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 56

mov eax, DWORD PTR \_hShowWin

mov DWORD PTR [esp+4], OFFSET FLAT:LC6

mov DWORD PTR [esp], eax

call \_SetWindowTextA@8

sub esp, 8

mov DWORD PTR \_calc\_done, 1

mov DWORD PTR [ebp-12], 0

jmp L20

L21:

mov eax, DWORD PTR [ebp-12]

imul eax, eax, 44

add eax, OFFSET FLAT:\_products+32

mov eax, DWORD PTR [eax]

mov DWORD PTR [ebp-28], eax

fild DWORD PTR [ebp-28]

mov eax, DWORD PTR [ebp-12]

imul eax, eax, 44

add eax, OFFSET FLAT:\_products+36

mov eax, DWORD PTR [eax]

mov DWORD PTR [ebp-28], eax

fild DWORD PTR [ebp-28]

fxch st(1)

fstp DWORD PTR [esp+4]

fstp DWORD PTR [esp]

call \_profit\_mark\_of

mov edx, eax

mov eax, DWORD PTR [ebp-12]

imul eax, eax, 44

add eax, OFFSET FLAT:\_products+40

mov BYTE PTR [eax], dl

add DWORD PTR [ebp-12], 1

L20:

cmp DWORD PTR [ebp-12], 5

jle L21

nop

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE12:

.globl \_r\_memcpy

.def \_r\_memcpy; .scl 2; .type 32; .endef

\_r\_memcpy:

LFB13:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 16

mov DWORD PTR [ebp-4], 0

jmp L23

L24:

mov edx, DWORD PTR [ebp-4]

mov eax, DWORD PTR [ebp+8]

add edx, eax

mov ecx, DWORD PTR [ebp-4]

mov eax, DWORD PTR [ebp+12]

add eax, ecx

movzx eax, BYTE PTR [eax]

mov BYTE PTR [edx], al

add DWORD PTR [ebp-4], 1

L23:

mov eax, DWORD PTR [ebp-4]

cmp eax, DWORD PTR [ebp+16]

jl L24

nop

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE13:

.globl \_r\_strncpy\_fillspace

.def \_r\_strncpy\_fillspace; .scl 2; .type 32; .endef

\_r\_strncpy\_fillspace:

LFB14:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 16

mov DWORD PTR [ebp-4], 0

mov DWORD PTR [ebp-8], 0

jmp L26

L30:

cmp DWORD PTR [ebp-4], 0

je L27

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+8]

add eax, edx

mov BYTE PTR [eax], 32

jmp L28

L27:

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+12]

add eax, edx

movzx eax, BYTE PTR [eax]

test al, al

jne L29

mov DWORD PTR [ebp-4], 1

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+8]

add eax, edx

mov BYTE PTR [eax], 32

jmp L28

L29:

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+8]

add edx, eax

mov ecx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+12]

add eax, ecx

movzx eax, BYTE PTR [eax]

mov BYTE PTR [edx], al

L28:

add DWORD PTR [ebp-8], 1

L26:

mov eax, DWORD PTR [ebp-8]

cmp eax, DWORD PTR [ebp+16]

jl L30

nop

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE14:

.globl \_r\_long8\_to\_mem

.def \_r\_long8\_to\_mem; .scl 2; .type 32; .endef

\_r\_long8\_to\_mem:

LFB15:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

push ebx

sub esp, 16

.cfi\_offset 3, -12

mov ecx, DWORD PTR [ebp+12]

mov edx, 1801439851

mov eax, ecx

imul edx

sar edx, 22

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov edx, eax

mov eax, DWORD PTR [ebp+8]

mov BYTE PTR [eax], dl

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+1]

mov ecx, DWORD PTR [ebp+12]

mov edx, 1125899907

mov eax, ecx

imul edx

sar edx, 18

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+2]

mov ecx, DWORD PTR [ebp+12]

mov edx, 351843721

mov eax, ecx

imul edx

sar edx, 13

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+3]

mov ecx, DWORD PTR [ebp+12]

mov edx, 1759218605

mov eax, ecx

imul edx

sar edx, 12

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+4]

mov ecx, DWORD PTR [ebp+12]

mov edx, 274877907

mov eax, ecx

imul edx

sar edx, 6

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+5]

mov ecx, DWORD PTR [ebp+12]

mov edx, 1374389535

mov eax, ecx

imul edx

sar edx, 5

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+6]

mov ecx, DWORD PTR [ebp+12]

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

mov ecx, edx

sub ecx, eax

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov eax, DWORD PTR [ebp+8]

lea ebx, [eax+7]

mov ecx, DWORD PTR [ebp+12]

mov edx, 1717986919

mov eax, ecx

imul edx

sar edx, 2

mov eax, ecx

sar eax, 31

sub edx, eax

mov eax, edx

sal eax, 2

add eax, edx

add eax, eax

sub ecx, eax

mov edx, ecx

mov eax, edx

add eax, 48

mov BYTE PTR [ebx], al

mov DWORD PTR [ebp-8], 0

jmp L32

L36:

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+8]

add eax, edx

movzx eax, BYTE PTR [eax]

cmp al, 48

jne L37

mov edx, DWORD PTR [ebp-8]

mov eax, DWORD PTR [ebp+8]

add eax, edx

mov BYTE PTR [eax], 32

add DWORD PTR [ebp-8], 1

L32:

cmp DWORD PTR [ebp-8], 7

jle L36

jmp L35

L37:

nop

L35:

nop

add esp, 16

pop ebx

.cfi\_restore 3

pop ebp

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE15:

.globl \_r\_serialize\_record

.def \_r\_serialize\_record; .scl 2; .type 32; .endef

\_r\_serialize\_record:

LFB16:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 12

mov eax, DWORD PTR [ebp+12]

mov DWORD PTR [esp+8], 8

mov DWORD PTR [esp+4], eax

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_r\_strncpy\_fillspace

mov eax, DWORD PTR [ebp+12]

mov eax, DWORD PTR [eax+32]

mov edx, DWORD PTR [ebp+8]

add edx, 8

mov DWORD PTR [esp+4], eax

mov DWORD PTR [esp], edx

call \_r\_long8\_to\_mem

mov eax, DWORD PTR [ebp+12]

mov eax, DWORD PTR [eax+36]

mov edx, DWORD PTR [ebp+8]

add edx, 16

mov DWORD PTR [esp+4], eax

mov DWORD PTR [esp], edx

call \_r\_long8\_to\_mem

mov eax, DWORD PTR [ebp+8]

add eax, 24

mov BYTE PTR [eax], 32

mov eax, DWORD PTR [ebp+8]

lea edx, [eax+25]

mov eax, DWORD PTR [ebp+12]

movzx eax, BYTE PTR [eax+40]

mov BYTE PTR [edx], al

mov eax, DWORD PTR [ebp+8]

add eax, 26

mov BYTE PTR [eax], 13

mov eax, DWORD PTR [ebp+8]

add eax, 27

mov BYTE PTR [eax], 10

nop

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE16:

.globl \_show\_list

.def \_show\_list; .scl 2; .type 32; .endef

\_show\_list:

LFB17:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

push edi

push ebx

sub esp, 240

.cfi\_offset 7, -12

.cfi\_offset 3, -16

mov DWORD PTR [ebp-219], 1701667182

mov DWORD PTR [ebp-215], 538976288

mov DWORD PTR [ebp-211], 1885302377

mov DWORD PTR [ebp-207], 1701013874

mov DWORD PTR [ebp-203], 1953853216

mov DWORD PTR [ebp-199], 1769107551

mov DWORD PTR [ebp-195], 1730176355

mov DWORD PTR [ebp-191], 1701077362

mov DWORD PTR [ebp-187], 2573

lea eax, [ebp-183]

mov ecx, 167

mov ebx, 0

mov DWORD PTR [eax], ebx

mov DWORD PTR [eax-4+ecx], ebx

lea edx, [eax+4]

and edx, -4

sub eax, edx

add ecx, eax

and ecx, -4

shr ecx, 2

mov edi, edx

mov eax, ebx

rep stosd

mov DWORD PTR [ebp-12], 34

mov DWORD PTR [ebp-16], 0

jmp L40

L41:

mov eax, DWORD PTR [ebp-16]

imul eax, eax, 44

lea edx, \_products[eax]

mov eax, DWORD PTR [ebp-12]

lea ecx, [ebp-219]

add eax, ecx

mov DWORD PTR [esp+4], edx

mov DWORD PTR [esp], eax

call \_r\_serialize\_record

add DWORD PTR [ebp-16], 1

add DWORD PTR [ebp-12], 28

L40:

cmp DWORD PTR [ebp-16], 5

jle L41

mov BYTE PTR [ebp-16], 0

mov eax, DWORD PTR \_hShowWin

lea edx, [ebp-219]

mov DWORD PTR [esp+4], edx

mov DWORD PTR [esp], eax

call \_SetWindowTextA@8

sub esp, 8

nop

lea esp, [ebp-8]

pop ebx

.cfi\_restore 3

pop edi

.cfi\_restore 7

pop ebp

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret

.cfi\_endproc

LFE17:

.section .rdata,"dr"

.align 8

LC2:

.long -858993459

.long 1072483532

.align 4

LC3:

.long 1056964608

.align 8

LC4:

.long -1717986918

.long 1070176665

.ident "RecolicLangC bata-1.0.1.6"

.def \_SetWindowTextA@8; .scl 2; .type 32; .endef

;main.s

.file "main.rasm"

.intel\_syntax noprefix

.globl \_hShowWin

.bss

.align 4

\_hShowWin:

.space 4

.section .rdata,"dr"

LC0:

.ascii "Hi\0"

.align 4

LC1:

.ascii "By Bensong Liu <root@recolic.net>, Licensed under Modified Mozilla Public License 2.0.\0"

.text

.globl \_mainWinProc@16

.def \_mainWinProc@16; .scl 2; .type 32; .endef

\_mainWinProc@16:

LFB11:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

sub esp, 120

mov eax, DWORD PTR [ebp+12]

cmp eax, 5

je L3

cmp eax, 5

ja L4

cmp eax, 1

je L5

cmp eax, 2

je L6

jmp L2

L4:

cmp eax, 15

je L7

cmp eax, 273

je L8

jmp L2

L5:

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_DrawMenuBar@4

sub esp, 4

jmp L9

L7:

lea eax, [ebp-76]

mov DWORD PTR [esp+4], eax

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_BeginPaint@8

sub esp, 8

mov DWORD PTR [ebp-12], eax

mov DWORD PTR [esp+8], 6

lea eax, [ebp-76]

add eax, 8

mov DWORD PTR [esp+4], eax

mov eax, DWORD PTR [ebp-12]

mov DWORD PTR [esp], eax

call \_FillRect@12

sub esp, 12

lea eax, [ebp-76]

mov DWORD PTR [esp+4], eax

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_EndPaint@8

sub esp, 8

nop

jmp L9

L3:

mov eax, DWORD PTR [ebp+20]

shr eax, 16

movzx eax, ax

lea ecx, [eax-20]

mov eax, DWORD PTR [ebp+20]

movzx eax, ax

lea edx, [eax-20]

mov eax, DWORD PTR \_hShowWin

mov DWORD PTR [esp+24], 0

mov DWORD PTR [esp+20], ecx

mov DWORD PTR [esp+16], edx

mov DWORD PTR [esp+12], 10

mov DWORD PTR [esp+8], 10

mov DWORD PTR [esp+4], 0

mov DWORD PTR [esp], eax

call \_SetWindowPos@28

sub esp, 28

jmp L9

L8:

mov eax, DWORD PTR [ebp+16]

movzx eax, ax

cmp eax, 2

je L11

cmp eax, 2

jg L12

cmp eax, 1

je L13

jmp L9

L12:

cmp eax, 3

je L14

cmp eax, 4

je L15

jmp L9

L13:

mov DWORD PTR [esp+12], 0

mov DWORD PTR [esp+8], 0

mov DWORD PTR [esp+4], 16

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_PostMessageA@16

sub esp, 16

jmp L10

L11:

call \_calc\_average

jmp L10

L14:

call \_show\_list

jmp L10

L15:

mov DWORD PTR [esp+12], 0

mov DWORD PTR [esp+8], OFFSET FLAT:LC0

mov DWORD PTR [esp+4], OFFSET FLAT:LC1

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_MessageBoxA@16

sub esp, 16

nop

L10:

jmp L9

L6:

mov DWORD PTR [esp], 0

call \_PostQuitMessage@4

sub esp, 4

jmp L9

L2:

mov eax, DWORD PTR [ebp+20]

mov DWORD PTR [esp+12], eax

mov eax, DWORD PTR [ebp+16]

mov DWORD PTR [esp+8], eax

mov eax, DWORD PTR [ebp+12]

mov DWORD PTR [esp+4], eax

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp], eax

call \_DefWindowProcA@16

sub esp, 16

jmp L16

L9:

mov eax, 0

L16:

leave

.cfi\_restore 5

.cfi\_def\_cfa 4, 4

ret 16

.cfi\_endproc

LFE11:

.section .rdata,"dr"

LC2:

.ascii "File\0"

LC3:

.ascii "Action\0"

LC4:

.ascii "Help\0"

LC5:

.ascii "Exit\0"

LC6:

.ascii "Average\0"

LC7:

.ascii "List\0"

LC8:

.ascii "About\0"

LC9:

.ascii "Recolic's Homework\0"

LC10:

.ascii "Ready.\0"

LC11:

.ascii "EDIT\0"

.text

.globl \_WinMain@16

.def \_WinMain@16; .scl 2; .type 32; .endef

\_WinMain@16:

LFB12:

.cfi\_startproc

push ebp

.cfi\_def\_cfa\_offset 8

.cfi\_offset 5, -8

mov ebp, esp

.cfi\_def\_cfa\_register 5

push edi

sub esp, 164

.cfi\_offset 7, -12

mov DWORD PTR [ebp-43], 1852405618

mov DWORD PTR [ebp-39], 1634493229

mov DWORD PTR [ebp-35], 761492339

mov WORD PTR [ebp-31], 12592

mov BYTE PTR [ebp-29], 0

lea edx, [ebp-84]

mov eax, 0

mov ecx, 10

mov edi, edx

rep stosd

mov DWORD PTR [ebp-80], OFFSET FLAT:\_mainWinProc@16

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [ebp-68], eax

lea eax, [ebp-43]

mov DWORD PTR [ebp-48], eax

lea eax, [ebp-84]

mov DWORD PTR [esp], eax

call \_RegisterClassA@4

sub esp, 4

call \_CreateMenu@0

mov DWORD PTR [ebp-12], eax

call \_CreatePopupMenu@0

mov DWORD PTR [ebp-16], eax

call \_CreatePopupMenu@0

mov DWORD PTR [ebp-20], eax

call \_CreatePopupMenu@0

mov DWORD PTR [ebp-24], eax

mov eax, DWORD PTR [ebp-16]

mov DWORD PTR [esp+12], OFFSET FLAT:LC2

mov DWORD PTR [esp+8], eax

mov DWORD PTR [esp+4], 16

mov eax, DWORD PTR [ebp-12]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov eax, DWORD PTR [ebp-20]

mov DWORD PTR [esp+12], OFFSET FLAT:LC3

mov DWORD PTR [esp+8], eax

mov DWORD PTR [esp+4], 16

mov eax, DWORD PTR [ebp-12]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov eax, DWORD PTR [ebp-24]

mov DWORD PTR [esp+12], OFFSET FLAT:LC4

mov DWORD PTR [esp+8], eax

mov DWORD PTR [esp+4], 16

mov eax, DWORD PTR [ebp-12]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov DWORD PTR [esp+12], OFFSET FLAT:LC5

mov DWORD PTR [esp+8], 1

mov DWORD PTR [esp+4], 0

mov eax, DWORD PTR [ebp-16]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov DWORD PTR [esp+12], OFFSET FLAT:LC6

mov DWORD PTR [esp+8], 2

mov DWORD PTR [esp+4], 0

mov eax, DWORD PTR [ebp-20]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov DWORD PTR [esp+12], OFFSET FLAT:LC7

mov DWORD PTR [esp+8], 3

mov DWORD PTR [esp+4], 0

mov eax, DWORD PTR [ebp-20]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov DWORD PTR [esp+12], OFFSET FLAT:LC8

mov DWORD PTR [esp+8], 4

mov DWORD PTR [esp+4], 0

mov eax, DWORD PTR [ebp-24]

mov DWORD PTR [esp], eax

call \_AppendMenuA@16

sub esp, 16

mov DWORD PTR [esp+44], 0

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp+40], eax

mov eax, DWORD PTR [ebp-12]

mov DWORD PTR [esp+36], eax

mov DWORD PTR [esp+32], 0

mov DWORD PTR [esp+28], -2147483648

mov DWORD PTR [esp+24], -2147483648

mov DWORD PTR [esp+20], -2147483648

mov DWORD PTR [esp+16], -2147483648

mov DWORD PTR [esp+12], 13565952

mov DWORD PTR [esp+8], OFFSET FLAT:LC9

lea eax, [ebp-43]

mov DWORD PTR [esp+4], eax

mov DWORD PTR [esp], 0

call \_CreateWindowExA@48

sub esp, 48

mov DWORD PTR [ebp-28], eax

mov DWORD PTR [esp+44], 0

mov eax, DWORD PTR [ebp+8]

mov DWORD PTR [esp+40], eax

mov DWORD PTR [esp+36], 0

mov eax, DWORD PTR [ebp-28]

mov DWORD PTR [esp+32], eax

mov DWORD PTR [esp+28], 1000

mov DWORD PTR [esp+24], 1000

mov DWORD PTR [esp+20], 10

mov DWORD PTR [esp+16], 10

mov DWORD PTR [esp+12], 1342177476

mov DWORD PTR [esp+8], OFFSET FLAT:LC10

mov DWORD PTR [esp+4], OFFSET FLAT:LC11

mov DWORD PTR [esp], 0

call \_CreateWindowExA@48

sub esp, 48

mov DWORD PTR \_hShowWin, eax

mov eax, DWORD PTR [ebp-28]

mov DWORD PTR [esp], eax

call \_DrawMenuBar@4

sub esp, 4

mov DWORD PTR [esp+4], 5

mov eax, DWORD PTR [ebp-28]

mov DWORD PTR [esp], eax

call \_ShowWindow@8

sub esp, 8

mov eax, DWORD PTR \_hShowWin

mov DWORD PTR [esp+4], 5

mov DWORD PTR [esp], eax

call \_ShowWindow@8

sub esp, 8

mov eax, DWORD PTR [ebp-28]

mov DWORD PTR [esp], eax

call \_UpdateWindow@4

sub esp, 4

mov eax, DWORD PTR \_hShowWin

mov DWORD PTR [esp], eax

call \_UpdateWindow@4

sub esp, 4

jmp L18

L19:

lea eax, [ebp-112]

mov DWORD PTR [esp], eax

call \_TranslateMessage@4

sub esp, 4

lea eax, [ebp-112]

mov DWORD PTR [esp], eax

call \_DispatchMessageA@4

sub esp, 4

L18:

mov DWORD PTR [esp+12], 0

mov DWORD PTR [esp+8], 0

mov DWORD PTR [esp+4], 0

lea eax, [ebp-112]

mov DWORD PTR [esp], eax

call \_GetMessageA@16

sub esp, 16

test eax, eax

jne L19

mov eax, 0

mov edi, DWORD PTR [ebp-4]

leave

.cfi\_restore 5

.cfi\_restore 7

.cfi\_def\_cfa 4, 4

ret 16

.cfi\_endproc

LFE12:

.ident "RecolicLangC bata-1.0.1.6"

.def \_DrawMenuBar@4; .scl 2; .type 32; .endef

.def \_BeginPaint@8; .scl 2; .type 32; .endef

.def \_FillRect@12; .scl 2; .type 32; .endef

.def \_EndPaint@8; .scl 2; .type 32; .endef

.def \_SetWindowPos@28; .scl 2; .type 32; .endef

.def \_PostMessageA@16; .scl 2; .type 32; .endef

.def \_calc\_average; .scl 2; .type 32; .endef

.def \_show\_list; .scl 2; .type 32; .endef

.def \_MessageBoxA@16; .scl 2; .type 32; .endef

.def \_PostQuitMessage@4; .scl 2; .type 32; .endef

.def \_DefWindowProcA@16; .scl 2; .type 32; .endef

.def \_RegisterClassA@4; .scl 2; .type 32; .endef

.def \_CreateMenu@0; .scl 2; .type 32; .endef

.def \_CreatePopupMenu@0; .scl 2; .type 32; .endef

.def \_AppendMenuA@16; .scl 2; .type 32; .endef

.def \_CreateWindowExA@48; .scl 2; .type 32; .endef

.def \_ShowWindow@8; .scl 2; .type 32; .endef

.def \_UpdateWindow@4; .scl 2; .type 32; .endef

.def \_TranslateMessage@4; .scl 2; .type 32; .endef

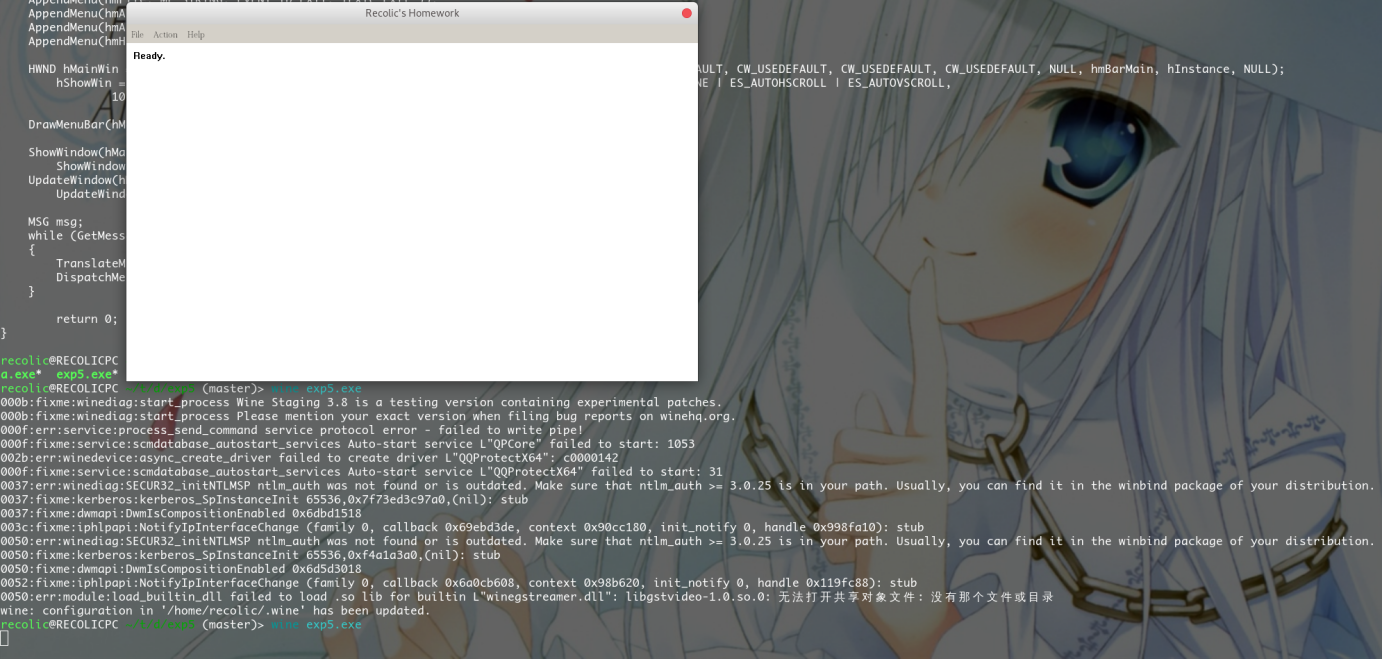
.def \_DispatchMessageA@4; .scl 2; .type 32; .endef

.def \_GetMessageA@16; .scl 2; .type 32; .endef

### 实验记录与分析

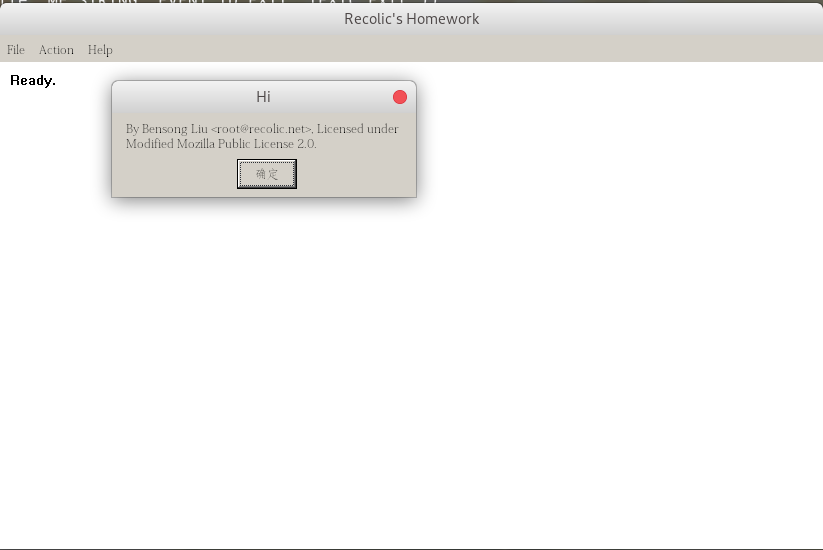
1. 根据参考资料提供的相关示例以及书本上的相关学习资料，我们可以比较清晰的了解Win32 与GUI有关的API。它维护一个基于消息机制的，事件驱动的消息队列，并且在消息处理回调函数中检查消息内容(uMsg, lParam, wParam)，并调用适当的过程。窗口由一个hMainWnd，一个hMenuBar，一个hEditWnd组成。程序员应当响应WM\_PAINT(用brush和dc手动重绘界面)，WM\_SIZE(调整大小，相应调整内部subWindow的大小)，WM\_COMMAND(按钮事件)，WM\_DESTROY(PostQuitMessage)。除此之外的消息类型交由DefWindowProc处理。

2. build target，然后直接打开exp5.exe，如下图所示。注意，由于开发环境限制，在此使用wine运行windows程序。

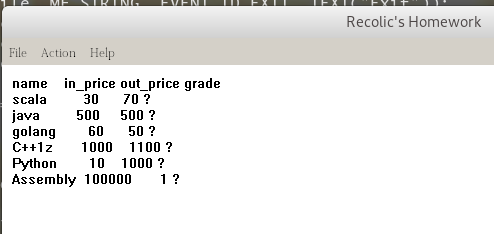


程序运行截图

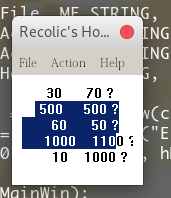
3. 依次选择Help->About，弹出的对话框如下图所示。

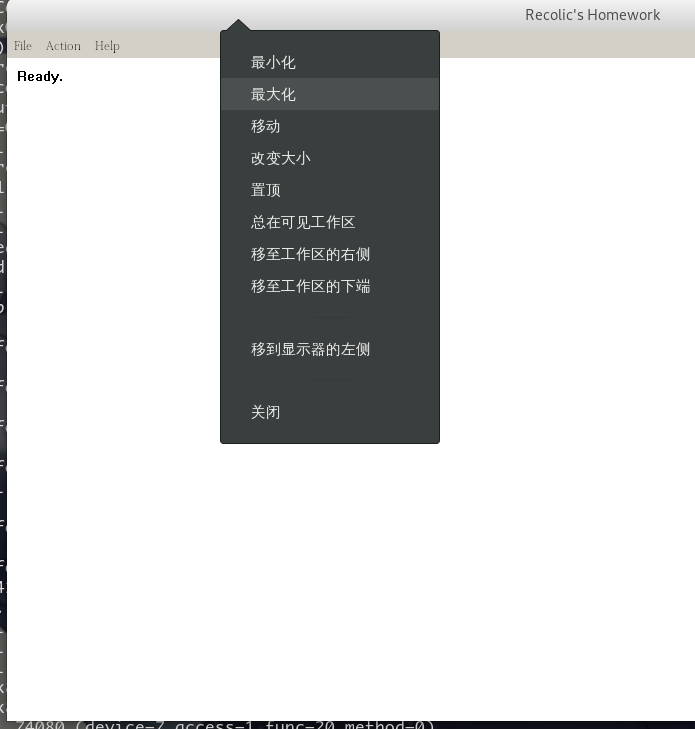


4. 然后点击Action菜单下的List，如下图所示(尚未计算Average)。

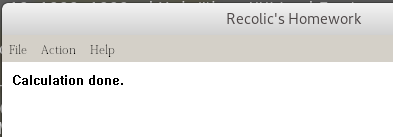


1. 然后拖动程序的窗口，调整窗口大小，观察到成绩表并没有消失，证明我们的重绘是成功的。观察systemMenu，最大化最小化等功能，他们都由windows或wine正常实现了。

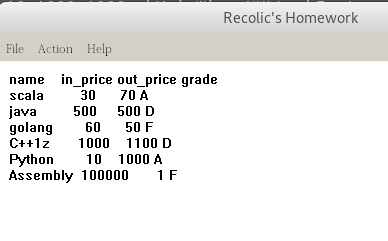




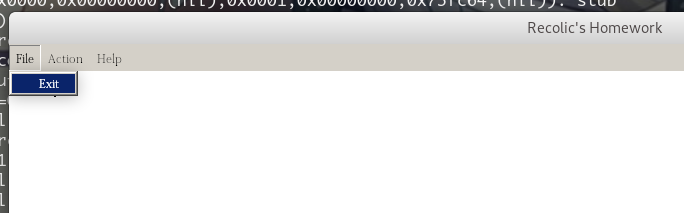
6. 然后点击Action菜单下的Average，如下图所示。



1. 再次List，如下图所示。



1. 最后点击File菜单下的Exit，程序退出。



可以注意到，在此过程中，wine没有出现任何错误提示信息，因为此程序只使用了最常用的WindowsAPI。

# 总结与体会

本实验是所有汇编实验中最后一次实验，同之前实验不同的是，这次实验建立在Win32编程的基础之上，而非我们之前学习的16位汇编。32位汇编和16位汇编的不同是这次实验核心的部分。在本次实验中，32位汇编和16位汇编的不同也导致一些问题。通过这次实验，我也熟悉了Win32相关的API，了解了有关32位编程的特性。

在本次实验中，我也遇到了不少问题。主要就是Win32编程一些独特的特性比如局部变量，以及函数原型申明等等。在不熟悉的情况下很容易造成各种汇编不通过的情况。比如忘了添加函数声明，以及局部变量的操作和寄存器的操作弄混了等等。本次实验主要是使用了前面实验中的代码，并且将之与Win32编程的框架相结合。

# 参考文献

[1] 万元珍 曹忠升 韩宗分.《80x86汇编语言程序设计》.华中科技大学出版社,

[2] 王爽.《汇编语言》.第3版.北京.清华大学出版社

[3] 汇编语言教学网站（https://recolic.net/go/asm）-》资料下载-》案例-》win32程序、编译和连接

[4] 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

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

[6] GDB: The GNU Project Debugger (https://www.gnu.org/software/gdb/)