

汇编语言程序设计实验报告

目录

1	实验目的与要求.....	1
2	实验内容.....	1
3	实验过程.....	2
3.1	任务 1.....	2
3.1.1	设计思想及存储单元分配.....	2
3.1.2	流程图.....	4
3.1.3	源程序.....	7
3.1.4	实验步骤.....	16
3.1.5	实验记录与分析.....	16
3.2	任务 2.....	21
3.2.1	设计思想及存储单元分配.....	21
3.2.2	流程图.....	22
3.2.3	源程序.....	22
3.2.4	实验步骤.....	28
3.2.5	实验记录与分析.....	28
4	总结与体会.....	41
	参考文献.....	43

汇编语言程序设计实验报告

1 实验目的与要求

- (1) 掌握子程序设计的方法与技巧，熟悉子程序的参数传递方法和调用原理；
- (2) 掌握宏指令、模块化程序的设计方法；
- (3) 掌握较大规模程序的合作开发与调试方法；
- (4) 掌握汇编语言程序与 C 语言程序混合编程的方法；
- (5) 熟悉 C 编译器的基本优化方法；
- (6) 了解 C 语言编译器的命名方法，主、子程序之间参数传递的机制。

2 实验内容

任务 1 宏与子程序设计

进一步修改与增强实验一任务四的网店商品信息管理程序的功能，主要调整功能三。

1. 调整后的功能三的描述

(1) 首先显示一个功能菜单（格式自行定义。若是未登录状态，只显示菜单“1”和“6”）：

1=查询商品信息，2=修改商品信息，3=计算平均利润率，

4=计算利润率排名，5=输出全部商品信息，6=程序退出。

输入 1-6 的数字进入对应的功能。

(2) 查询商品信息

提示用户输入要查询的商品名称。若未能在第一个网店中找到该商品，重新提示输入商品名称。若只输入回车，则回到功能三（1）。

找到该商品之后，按照：“SHOP1，商品名称，销售价，进货总数，已售数量”顺序显示该商品的信息，同时还要将“SHOP2”中该商品的信息也显示出来。显示之后回到功能三（1）。

(3) 修改商品信息

提示用户输入要修改信息的商品名称（先指定网店名称）。[若把接下来的处理步骤写成子程序，则网店名称和商品名称（或其偏移地址）就是子程序的入口参数，是否找到、是否是回车或者修改成功的信息是出口参数]。若未能在指定网店中找到该商品，重新提示输入网店名称和商品名称。若只输入回车，则回到功能三（1）。

找到该商品之后，按照：进货价，销售价，进货总数的次序，逐一先显示原来的数值，然后输入新的数值（若输入有错，则重新对该项信息进行显示与修改。若直接回车，则不修改该项信息）。

如：进货价：25》24 //符号“》”仅作为分隔符，也可以选择其他分隔符号

销售价：46》5A6 //输入了非法数值，下一行重新显示和输入

销售价：46》56

进货总数：30》 //直接回车时，对这项信息不做修改

当对三项信息都处理完毕后，回到功能三（1）。

(4) 计算平均利润率

首先计算 SHOP1 中第一个商品的利润率 PR1，然后在 SHOP2 网店中寻找该商品，也计算

汇编语言程序设计实验报告

其利润率 PR2。最后求出该商品的平均利润率 $APR=(PR1+PR2)/2$ ，并保存到 SHOP1 的利润率字段中。重复上述步骤，依次将每个商品的平均利润率计算出来。回到功能三（1）。

（5）计算利润率排名

对 SHOP2 中的每个商品按照平均利润率的大小排名，排名信息存放到 SHOP2 中商品的利润率字段中。回到功能三（1）。

（6）输出全部商品信息

将 SHOP1 和 SHOP2 中的所有商品信息显示到屏幕上，包括平均利润率和排名（替代了商品原有的利润率字段）。具体的显示格式自行定义（可以分网店显示，也可以按照商品排名显示，等等，显示方式可以作为子程序的入口参数）。回到功能三（1）。

2.其他要求

（1）两人一组，一人负责包括菜单显示、程序退出在内的主程序，以及菜单中的功能（1）和（2）；另一人负责菜单中的功能（3）、（4）和（5）。各自汇编自己的模块，设计测试方法，测试通过；然后把自己的模块交给对方，各自把对方的程序整合到自己的程序中，连接生成一个程序，再进行整体调试。

（2）排名的基本要求是按照平均利润率从高到低计算名次，也可以考虑按照指定字段（比如已售数量等）排名。相同平均利润率时排名相同，下一个相邻平均利润率的名次应该是排名在前的所有商品种类“和”的下一个数值。

（3）将 9 号和 10 号 DOS 系统功能调用定义成宏指令并调用。功能（1）-（5）应尽量采用子程序方式实现。需要借鉴书上的进制转换程序：十进制转二进制的子程序 F10T2 和二进制转十进制的子程序 F2T10。

任务 2：在 C 语言程序中调用 汇编语言实现的函数

对于任务 1 的程序进行改造，主控程序、以及输入输出较多的某一个功能（如功能（1）、（2）、（5）中的某一个）用 C 语言实现，其他功能用独立的汇编语言子程序的方式实现；在 C 语言程序中调用汇编语言子程序。

3 实验过程

3.1 任务 1

3.1.1 设计思想及存储单元分配

设计思想：本次任务设计思想与之前任务没有太大区别，首先菜单功能直接利用 dos9 号调用输出即可。查询商品的信息也和之前做过的功能类似，这次新增的修改商品信息的功能其实也是在查询功能上的改进，只需要查找到商品再直接修改其中的数据就行了。

存储单元分配：

数据段用 menu1, menu2, menu3 作为菜单输出，以\$结尾，便于输出。还利用了 shop1, shop2, gname, price 等等变量名作为字符串的变量名，以便输出各种提示信息。之后商店名，商品名，用

汇编语言程序设计实验报告

户名，密码等信息的存储和之前的实验一任务 5 相同。

本次任务将 dos9 号，10 号调用定义成了宏指令，增强了代码易读性。

寄存器使用：

在登录阶段使用 bl，dl 寄存器进行输入信息与已有信息的比较。

在子程序 query_1 中，寄存器 cx，si 用来计数，bp 存放找到商品的偏移地址。dl 用来比较。

在子程序 query_2 中，用到了相同的寄存器。

在子程序 modify_item 中，al 用于接受输入操作指令，ax 用来显示之前商品的信息，si 用来指向输入数据的偏移地址，cx 用于存放输入数据的长度。

汇编语言程序设计实验报告

3.1.2 流程图

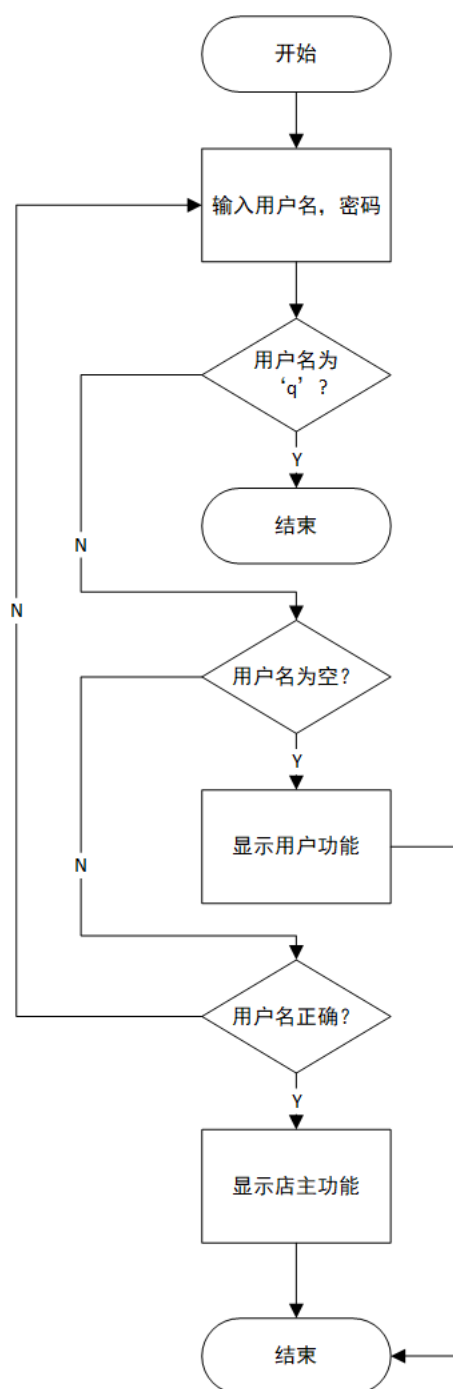
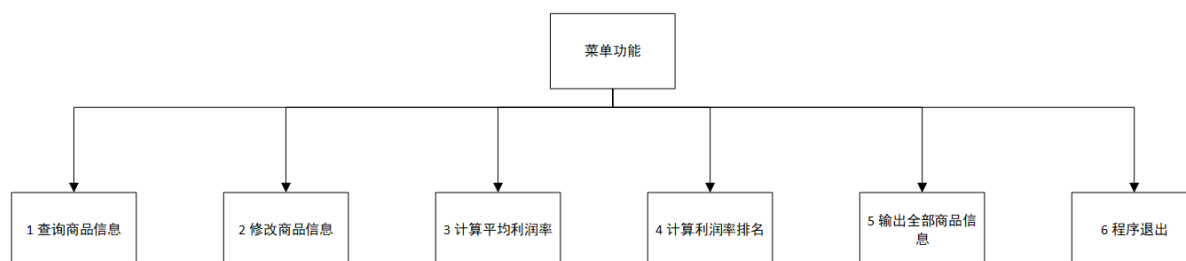


图 3.1.1 菜单功能流程图



汇编语言程序设计实验报告

图 3.1.2 模块功能图

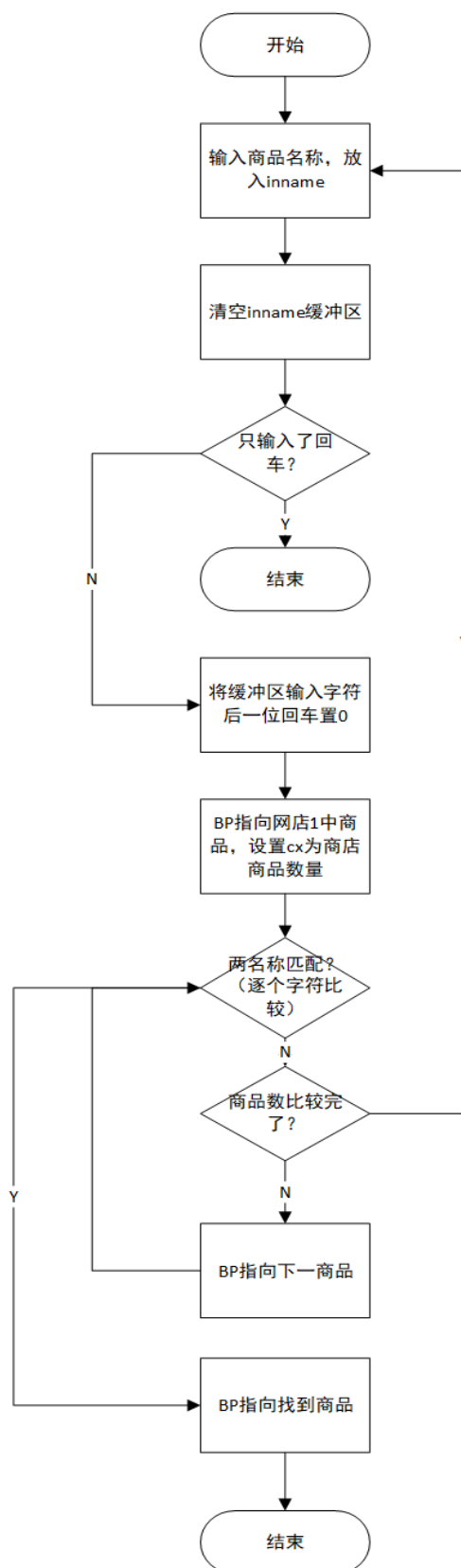
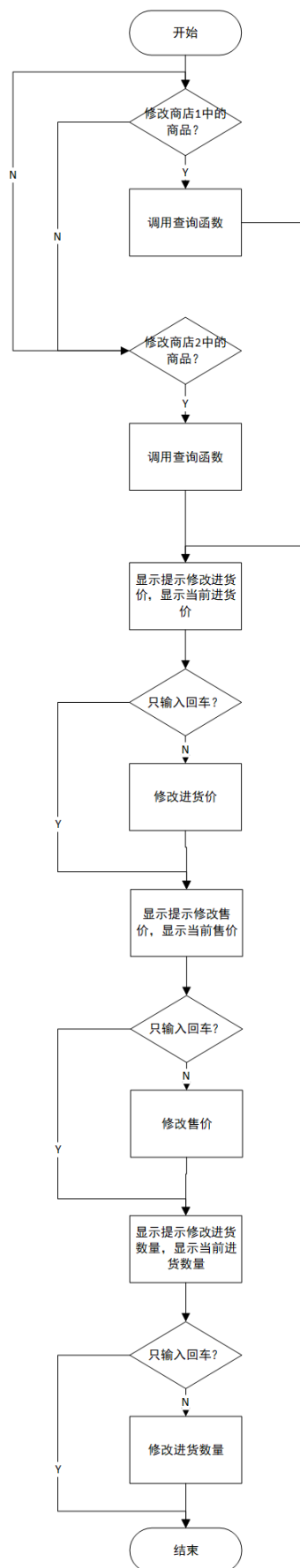


图 3.1.3 查询商品流程图 (1,2 网店相同)

汇编语言程序设计实验报告



汇编语言程序设计实验报告

图 3.1.4 修改商品信息流程图

3.1.3 源程序

```
NAME    wan12
;程序功能：实现菜单显示，查询商品信息，修改商品信息功能
;无特别需要注意的地方

;本模块代码提供者：王超凡
;同组成员：蒋苡杭，邓永煜

public  gbl,prol,find,gal,prorank,cost,prank,total,price,f2t10,gal,shop2,gbl,shop1,sold,
profit,gname,query_1,query_2,modify_item
extrn fresh:far,rank:far,_outall:far
.386
assume  cs:code,ds:data,ss:stack
stack  segment      use16      para      stack      'stack'
        db 200 dup(0)
stack  ends
data    segment      use16      para      public      'data'
        menu1 db 0dh,0ah,'-----menu-----',
                db 0dh,0ah,' 1 = inquire articles information'
                db 0dh,0ah,' 2 = modify articles information'
                db 0dh,0ah,' 3 = calculate the average rate of profit'
                db 0dh,0ah,' 4 = calculate the rank of profit'
                db 0dh,0ah,' 5 = output all articles information'
                db 0dh,0ah,' 6 = exit'
                db 0dh,0ah,'-----',0dh,0ah,'$'
        menu2 db 0dh,0ah,'-----menu-----',
                db 0dh,0ah,' 1 = inquire articles information'
                db 0dh,0ah,' 2 = exit'
                db 0dh,0ah,'-----',0dh,0ah,'$'
        menu3 db '1.modify shop1',0dh,0ah
                db '2.modify shop2',0dh,0ah,'$'
        shop1 db 'shop1:',0dh,0ah,'$'
        shop2 db 'shop2:',0dh,0ah,'$'
        gname db 'name:$'
        price db 'price:$'
        total db 'total:$'
        sold db 'sold:$'
        cost db 'cost:$'
        profit db 'profit:$'
        prank db 'rank:$'
        nameinput db 0ah,0dh,'Please input your name: ','$'
        pwdinput db 0ah,0dh,'Please input your code: ','$'
        iteminput db 0ah,0dh,'Please input the article name you want to inquire:',0dh,0ah,'$'
        faillog db 0ah,0dh,'Failed to log in!','$'
        changev db '-->$'

        bname db 'chaofan',0,0,0 ;用户名
        bpass db 'test',0,0 ;密码
        n equ 3 ;商品个数
        s1 db 'shop1',0
        gal db 'pen',7 dup(0) ;商品名称
        dw 35,56,70,25,? ;进价、售价、总数、已售
```


汇编语言程序设计实验报告

```
ga2      db  'book',6 dup(0)
          dw  12,30,25,5,?
ga3      db  'kindle',4 dup(0)
          dw  58,170,20,11,?
s2       db  'shop2',0
gb1      db  'pen',7 dup(0)
          dw  35,50,30,24,?
gb2      db  'book',6 dup(0)
          dw  12,28,20,15,?
gb3      db  'kindle',4 dup(0)
          dw  90,280,30,12,?
auth     db  0
sign     db  ?
prol     dd  0
prorank  dw  n dup(0)
buf      db  12 dup(?)
inname   db  10
          db  0
          db  10 dup(0)
inpwd    db  6
          db  0
          db  6 dup(0)
initem   db  10
          db  0
          db  10 dup(0)
innum    db  5
          db  0
          db  5 dup(0)
data     ends
;dos10
read macro a
    push ax
    push dx
    lea dx,a
    mov ah,10
    int 21h
    pop dx
    pop ax
endm
;dos9
write macro a
    push ax
    push dx
    lea dx,a
    mov ah,9
    int 21h
    pop dx
    pop ax
endm
;换行
crlf macro
    mov dl,0ah
    mov ah,2h
    int 21h
endm
```

汇编语言程序设计实验报告

```
code    segment    use16    para    public    'code'
start:  mov ax,data
        mov ds,ax
        jmp login
failtologin:
        mov auth,0
        write faillog
        crlf
login:  write nameinput ;提示输入姓名
        crlf
        read inname
        crlf
        cmp byte ptr [inname+1],0 ;仅输入回车
        je user ;普通用户
        cmp byte ptr[inname+2],'q' ;若输入为q
        je continuejudge ;继续判断
        cmp byte ptr[inname+1],7 ;长度不一致
        jne failtologin
password:
        write pwdinput ;提示输入密码
        crlf
        read inpwd
        crlf
        cmp byte ptr[inpwd+1],4 ;长度不一致
        jnz failtologin
        mov si,0 ;计数
        mov cl,[inname+1]
        mov ch,0
cmpa:   mov dl,[bname+si] ;判断用户名是否正确
        mov bl,[inname+si+2]
        cmp bl,dl
        jnz failtologin
        inc si
        loop cmpa
        mov si,0
        mov cl,[inpwd+1]
        mov ch,0
cmpp:   mov dl,[bpass+si] ;判断密码是否正确
        mov bl,[inpwd+si+2]
        cmp bl,dl
        jnz failtologin
        inc si
        loop cmpp
        mov auth,1 ;状态改变
manager:
        write menu1
        mov ah,1
        int 21h
        crlf
        cmp al,'1'
        je f3_1
        cmp al,'2'
        je f3_2
        cmp al,'3'
        je f3_3
        cmp al,'4'
        je f3_4
```

汇编语言程序设计实验报告

```
    cmp al,'5'
    je f3_5
    cmp al,'6'
    je f3_6
    jmp manager
user: write menu2
      mov ah,1
      int 21h
      crlf
      cmp al,'1'
      je f3_1
      cmp al,'2'
      je f3_6
      jmp user

f3_1: call query_1 ;商店1 找
      write shop1
      write gname
      mov bl,[initem+1]
      mov bh,0
      mov byte ptr [bx+initem+2],'$'
      write initem+2
      crlf
      write price
      mov ax,ds:[bp+12]
      call f2t10
      crlf
      write total
      mov ax,ds:[bp+14]
      call f2t10
      crlf
      write sold
      mov ax,ds:[bp+16]
      call f2t10
      crlf

      call query_2 ;商店2 找
      write shop2
      write gname
      mov bl,[initem+1]
      mov bh,0
      mov byte ptr [bx+initem+2],'$'
      write initem+2
      crlf
      write price
      mov ax,ds:[bp+12]
      call f2t10
      crlf
      write total
      mov ax,ds:[bp+14]
      call f2t10
      crlf
      write sold
      mov ax,ds:[bp+16]
      call f2t10
      crlf
      cmp auth,1
```

汇编语言程序设计实验报告

```
        jne user
        jmp manager
f3_2:   call modify_item
        jmp manager
f3_3:   mov cx,n
        lea si,gal
lp:     call fresh
        add si,20
        loop lp
        jmp manager
f3_4:   call rank
        jmp manager
f3_5:   call _outall
        jmp manager
f3_6:   mov ah,4ch
        int 21h
;shop1 查找商品, 商品偏移地址存在 bp
query_1 proc near
        push cx
        push bx
        push si
        push dx
notfind:
        write iteminput ;提示输入商品名
        crlf
        mov si,0
        mov cx,10
init:   mov [si+initem+2],0 ;清空缓冲区
        inc si
        loop init
        read initem
        crlf
        cmp byte ptr [initem+1],0 ;只输入回车
        je empty
        mov bl,[initem+1]
        mov bh,0
        mov byte ptr [initem+bx+2],0 ;最后字节置0
        lea bp,gal
        mov cx,n
cmpl:   mov si,0
        push cx
        mov cx,10
cmpitem:
        mov dl,[initem+si+2] ;逐个字节比较
        cmp dl,ds:[bp+si]
        jnz nextitem
        inc si ;下个字节
        loop cmpitem
        jmp break
nextitem:
        add bp,20
        pop cx
        loop cmpl
        jmp notfind
break:  pop cx
        pop dx
        pop si
```

汇编语言程序设计实验报告

```
        pop bx
        pop cx
        ret
empty:   pop dx
        pop si
        pop bx
        pop cx
        cmp auth, 1
        jne user
        jmp manager
query_1 endp

;shop2 查找商品, 商品偏移地址存在 bp
query_2 proc near
        push cx
        push bx
        push si
        push dx

        mov bl, [initem+1]
        mov bh, 0
        mov byte ptr [initem+bx+2], 0 ;最后字节置 0
        lea bp, gbl
        mov cx, n
cmp2:    mov si, 0
        push cx
        mov cx, 10
cmpitem2:
        mov dl, [initem+si+2]      ;逐个字节比较
        cmp dl, ds:[bp+si]
        jnz nextitem2
        inc si ;下个字节
        loop cmpitem2
        jmp break2
nextitem2:
        add bp, 20
        pop cx
        loop cmp2
break2:  pop cx
        pop dx
        pop si
        pop bx
        pop cx
        ret
query_2 endp
;修改商品信息子程序
modify_item proc near
        push dx
        push ax
        push bp
        push si

        write menu3
        mov dx, 16
op:      mov ah, 1
        int 21h
        crlf
```

汇编语言程序设计实验报告

```
        cmp al,'1'
        je sh1
        cmp al,'2'
        je sh2
        jmp op
sh1:    call query_1
        call modify_op
        jmp breakmodify
sh2:    call query_1
        call query_2
        call modify_op
        jmp breakmodify
breakmodify:
        pop si
        pop bp
        pop ax
        pop dx
        ret
modify_item endp
;修改操作
modify_op proc
        mov ch,0 ;计数
jinhuo: write cost ;修改进货价
        mov ax,word ptr ds:[bp+10]
        call f2t10
        write changev
        read innum
        crlf
        lea si,innum+2
        mov cl,[innum+1]
        cmp cx,0
        je xiaoshou
        call f10t2
        cmp si,-1
        je jinhuo
        mov ds:[bp+10],ax
xiaoshou:
        write price ;修改售价
        mov ax,word ptr ds:[bp+12]
        call f2t10
        write changev
        read innum
        crlf
        lea si,innum+2
        mov cl,[innum+1]
        cmp cx,0
        je zongshu
        call f10t2
        cmp si,-1
        je xiaoshou
        mov ds:[bp+12],ax
zongshu: write total ;修改进货数量
        mov ax,word ptr ds:[bp+14]
        call f2t10
        write changev
        read innum
        crlf
```

汇编语言程序设计实验报告

```
        lea si, innum+2
        mov cl, [innum+1]
        cmp cx, 0
        je breakmop
        call f10t2
        cmp si, -1
        je zongshu
        mov ds: [bp+14], ax
breakmop:
        ret
modify_op endp

find    proc
        push    bx
        push    ax
        push    bp
        mov di, offset gbl
        mov bx, 0    ;先用 bx 算出需要比较的字符数
        mov bp, 0    ;bp 用于带出字符串长度
_f1:    inc bx
        mov bp, bx
        cmp BYTE PTR[si+bx], 0
        jnz _f1
_f2:    mov al, BYTE PTR[di+bx]
        mov ah, BYTE PTR[si+bx]
        cmp al, ah
        jnz _f3
        dec bx
        cmp bx, 0
        jnl _f2
        pop bp
        pop ax
        pop bx
        ret
_f3:    add di, 20
        mov bx, bp
        jmp _f2
find    endp

;输出存放在 ax 中的十进制数
f2t10 proc near
        push ebx
        push si
        lea si, buf
        cmp dx, 32
        jne b
        movsx eax, ax
b:       or eax, eax
        jns plus
        neg eax
        mov byte ptr [si], '-'
        inc si
plus:    mov ebx, 10
        call radix
        mov byte ptr [si], '$'
        write buf
        pop si
```

汇编语言程序设计实验报告

```
        pop ebx
        ret
f2t10   endp
radix   proc
        push cx
        push edx
        xor cx, cx
lop1:   xor edx, edx
        div ebx
        push dx
        inc cx
        or  eax, eax
        jnz lop1
lop2:   pop ax
        cmp al, 10
        jb  l1
        add al, 7
l1:     add al, 30h
        mov [si], al
        inc si
        loop lop2
        pop edx
        pop cx
        ret
radix   endp
;输入十进制数转二进制
;入口参数:si 十进制串首址, cx 十进制串长度, dx 转换为 16 位或 32 位标志
;出口参数:si 为-1 时转换失败, eax 存放转换后的数
f10t2   proc near
        push ebx
        mov eax, 0
        mov sign, 0
        mov bl, [si]
        cmp bl, '+'
        je  f10
        cmp bl, '-'
        jne next2
        mov sign, 1
f10:    dec cx
        jz  err
next1:  inc si
        mov bl, [si]
next2:  cmp bl, '0'
        jb  err
        cmp bl, '9'
        ja  err
        sub bl, 30h
        movzx ebx, bl
        imul eax, 10
        jo  err
        add eax, ebx
        jo  err
        js  err
        jc  err
        dec cx
        jnz next1
        cmp dx, 16
```


汇编语言程序设计实验报告

```
        jne pp0
        cmp eax,7fffh
        ja err
pp0:    cmp sign,1
        jne ppl
        neg eax
ppl:    pop ebx
        ret
err:    mov si,-1
        jmp ppl
f10t2  endp
continuejudge:
        cmp byte ptr[inname+1],1
        jnz password
        mov ah,4ch
        int 21h
code ends
end start
```

3.1.4 实验步骤

- 1.准备上机实验环境，编辑，汇编，连接文件。
- 2.首先测试未登录状态下显示的功能，逐个测试功能正确性。
- 2.输入老板姓名，密码，进入登录状态下的菜单。
- 3.测试登录状态下的菜单所有功能是否能正确执行。
- 5.完成思考题

3.1.5 实验记录与分析

1. 实验环境条件：P3 1GHz，256M 内存；WINDOWS 10 下 DOSBox0.73；EDIT.EXE 2.0；MASM.EXE 6.0；LINK.EXE 5.2；TD.EXE 5.0。

2. 汇编源程序时，汇编程序没有报错，进入连接。

3. 连接过程发生异常。

（1）当时没有截图，再次说明错误情况：数据段不同名，无法连接。在修改数据段名称后，发现可以正常运行。

4. 程序中的逻辑错误如下：（由于当时没有截图，现只在下面说明情况）

（1）在调用完在网店 1 中查询商品的功能后，由于没有考虑清除，在子程序里将商品名称后面的\$给清掉了便于比较是否字符串相同，后来在调用在网店 2 中查询商品功能后，直接输入要查询的商品名称，会出现一堆乱码，原因是在调用完后没有在商品名称最后加上\$导致 dos9 号调用一直输出。

（2）在做网店 2 查询功能时，以为只是简单改一下在网店 1 中的查询功能就可以了，结果发现不行，它最后只会输出一个网店的商品，当你再次使用查询功能时，会显示另一个网店的商品。原因是在网店 2 查询功能中也加入了输入商品名称的功能，没有删去，导致程序出现错误。

5. 未登录状态下测试菜单功能：

（1）测试查询商品信息功能

汇 编 语 言 程 序 设 计 实 验 报 告

```
Please input your name:

-----menu-----
 1 = inquire articles information
 2 = exit
-----
1
Please input the article name you want to inquire:
```

图 3.1.5 未登录状态下菜单

输入要查询的商品名称 kindle，显示两个网店商品的名称，售价，进货数量，销售量。

```
Please input the article name you want to inquire:
kindle
shop1:
  name:kindle
  price:170
  total:20
  sold:11
shop2:
  name:kindle
  price:280
  total:30
  sold:12
-----menu-----
 1 = inquire articles information
 2 = exit
-----
```

图 3.1.6 查询商品信息功能

输入回车，回到菜单界面。

```
Please input the article name you want to inquire:

-----menu-----
 1 = inquire articles information
 2 = exit
-----
```

图 3.1.7 查询商品信息功能，只输入回车

输入不存在的商品名称，重新提示输入商品名称。

```
Please input the article name you want to inquire:
ok
Please input the article name you want to inquire:
```

图 3.1.8 查询商品信息功能，输入不存在商品情况

(2) 输入操作 2，程序退出。

```
-----menu-----
 1 = inquire articles information
 2 = exit
-----
2
C:\N>
```

图 3.1.9 程序退出

6. 登录状态下测试菜单功能

(1) 输入用户名，密码（登录验证用户名，密码的正确性在之前任务中已经完成过，此处不

汇编语言程序设计实验报告

展示)

```
Please input your name:
chaofan

Please input your code:
test

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.10 登录状态下菜单栏

(2) 测试修改商品信息功能

在商店 1 中修改商品 pen。其中只修改进货价，售价，进货总数不变。

```
2
1.modify shop1
2.modify shop2
1

Please input the article name you want to inquire:
pen
cost:35-->30
price:56-->57
total:70-->

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.11 修改商品信息功能

在商店 2 中修改商品 pen。只修改进货价，其他不变。

```
2
1.modify shop1
2.modify shop2
2

Please input the article name you want to inquire:
pen
cost:35-->36
price:50-->
total:30-->

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.12 修改商品信息功能

查询商品 pen 信息，确保修改成功。

shop1:	name:pen	cost:30	price:57	total:70
shop2:	name:pen	cost:36	price:50	total:30

图 3.1.13 确认修改成功

输入错误商品名称，重新提示输入商品名称，输入回车退出。

汇编语言程序设计实验报告

```
2
1.modify shop1
2.modify shop2
1

Please input the article name you want to inquire:
ok
Please input the article name you want to inquire:
```

图 3.1.14 输入错误商品名称

```
Please input the article name you want to inquire:

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.15 输入回车

输入非法数值，重新提示输入。

```
book
cost:12-->1ra
cost:12-->
price:30-->
total:25-->
```

图 3.1.16 输入非法数值

(3) 计算平均利润率功能测试

```
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
3

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.17 平均利润率功能测试

(4) 计算利润率排名功能测试

汇编语言程序设计实验报告

```
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
4
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.18 利润率排名功能测试

(5) 输出全部商品信息功能测试

利润率能正确计算，负数也能显示，排名正确。

```
5
shop1:
name:pen           cost:30      price:57      total:70
sold:25            profit:-11
name:book          cost:12      price:30      total:25
sold:5             profit:12
name:kindle        cost:58      price:170     total:20
sold:11            profit:42
shop2:
name:pen           cost:36      price:50      total:30
sold:24            rank:3
name:book          cost:12      price:28      total:20
sold:15            rank:2
name:kindle        cost:90      price:280     total:30
sold:12            rank:1
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.1.19 输出全部商品信息功能测试

7. 思考题

(1) 在 TD 中跟踪到子程序内部有几种方法？观察子程序调用和返回堆栈的变化。

至少有两种方法，在子程序中设置断点或者利用 F7 单步调试进入子程序。子程序调用时，如果是短程，则 IP 进栈，如果是远程，则 CS, IP 依次进栈。返回时，如果是短程，则 IP 出栈，如果是远程，则 IP, CS 依次出栈。

(2) 注意观察 FAR、NEAR 类型子程序的 RET 指令的机器码有何不同？观察 FAR 类型子程序被调用时堆栈的变化情况。

这种情况在上一个思考题中已经给出了解答。

(3) 通过把一个模块拆成多个模块或反之，体会子程序和模块化程序设计的方法，体会模块调用关系图、子程序功能说明、输入/输出说明在程序设计中的作用。

模块化设计更有利于一个团队合作完成某个项目，功能等说明有利于模块的正常运行。

(4) 观察不同模块的可合并段合并后变量偏移地址的变化情况。观察不同段在内存里的放置次序。体会模块间段的定义及其对应的装配方法。

汇编语言程序设计实验报告

可合并段合并后，变量偏移地址按照连接顺序合并后的段来计算。不同段在内存中放置顺序不是固定的，按照第一个连接模块来决定段的放置顺序。

(5) 在编程中使用不同的子程序参数传递方法来编写子程序。

在源程序中已经给出。

(6) 观察模块间的参数的传递方法，包括公共符号的定义和外部符号的引用，若符号名不一致或类型不一致会有什么现象发生？

在编译时不会出错，但是在 link 时会出现错误，找不到名称外部扩展。

(7) 通过 TD 观察宏指令在执行程序中的替换和扩展，解释宏和子程序的调用有何不同。

宏是在编译时已经将其翻译了一遍，直接在代码中。子程序调用是在代码进行时进行调用。

(8) 如何使菜单等显示信息显示得更漂亮一点？

可以加一些边界，还可以排一下版，如上面图所示。

(9) EXTRN 说明语句放在 .386 之前或者之后有什么区别？

在目前看来没有区别。

(10) EXTRN 说明的变量的段与段寄存器的关联关系 (ASSUME 伪指令所表达的信息) 是否能带入到本模块中？如果不能带入，是否可以通过加段前缀的方法来解决？

不能带入。可以通过加段前缀的方法来解决。

(11) 如何利用宏功能使汇编语言的程序变得更加直观易读？

将一些重复的程序，运用较多的程序进行宏定义，这样可以减少主程序中的代码量，同时使程序变得直观易读。

3.2 任务 2

3.2.1 设计思想及存储单元分配

本次实验由 C 语言部分和汇编程序构成，开发环境为 visual studio 2017 community。实现方法在下文中给出。

在 C 语言里，定义了结构体 Item, Shop。Item 结构体包含物品的名称，类型为字符数组，成本价，售价，进货量，销售量，利润，类型为 short 类型。Shop 结构体包含商店名称，类型为字符数组，商品结构体。

关于汇编程序部分则做了部分修改以适应 win32 编程。

汇编语言程序设计实验报告

3.2.2 流程图

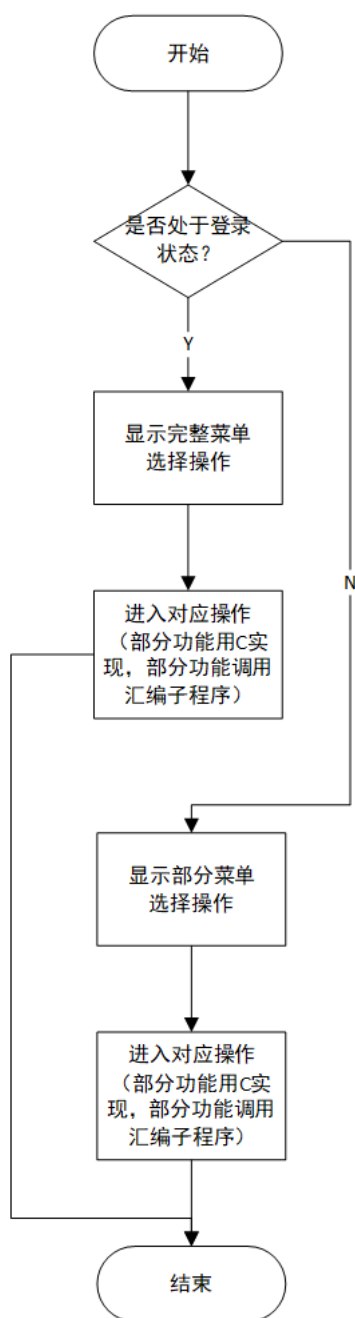


图 3.2.1 修改后程序流程图

3.2.3 源程序

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

typedef struct Item
{
```

汇编语言程序设计实验报告

```
    char item_name[10];
    short cost, price, total, sold, profit;
}Item;

typedef struct Shop
{
    char name[6];
    Item G1, G2, G3;
}Shop;

void modify_info(Item *gal);
void avg_rate(Item *gal, Item *gbl);
void rank(Shop *s1, Shop *s2);
void out_all_info(Shop s1, Shop s2);

extern "C" Item* query_articles(Item *GAl, char *name);//调用声明

int main()
{
    char admin_name[8] = "chaofan";
    char password[5] = "test";
    char input_user_name[11];
    char in_pwd[7];
    Shop s1,s2;
    strcpy(s1.name, "Shop1");
    strcpy(s2.name, "Shop2");
    s1.G1 = { "pen", 35, 56, 70, 25, 0 };
    s1.G2 = { "book", 12, 30, 25, 5, 0 };
    s1.G3 = { "kindle", 58, 170, 20, 11, 0 };
    s2.G1 = { "pen", 35, 50, 30, 24, 0 };
    s2.G2 = { "book", 12, 28, 20, 15, 0 };
    s2.G3 = { "kindle", 90, 280, 30, 12, 0 };
    while (1)
    {
        printf("Please input your name('q' to quit) :");
        fgets(input_user_name, 8, stdin);
        if (input_user_name[0] == 'q' && input_user_name[1] == '\n') break;//只输入q退出
        else if (!strcmp(input_user_name, admin_name))//输入管理者用户名密码
        {
            printf("Please input your code :");
            scanf("%s", in_pwd);
            getchar();
            if (strcmp(in_pwd, password))//密码错误
            {
                printf("Failed to log in!\n");
                continue;
            }
            int op;
            while (1)
            {
                system("cls");
                printf("\t\t-----menu-----\n");
                printf("\t\t 1 = inquire articles information\n");
                printf("\t\t 2 = modify articles information\n");
                printf("\t\t 3 = calculate the average rate of profit\n");
                printf("\t\t 4 = calculate the rank of profit\n");
                printf("\t\t 5 = output all articles information\n");
```


汇编语言程序设计实验报告

```
printf("\t\t 6 = exit\n");
printf("\t\t-----\n");
scanf("%d", &op);
getchar();
char in_item[10] = { '\0' };//初始化
if (op == 1)//调用汇编程序实现
{
    printf("Please input the article name you want to inquire:");
    scanf("%s", in_item);
    Item *i1 = query_articles(&s1.G1, in_item);//调用汇编程序
    if (i1)
    {
        printf("Shop1:\n");
        printf("name:%s\tprice:%hd\ttotal:%hd\tsold:%hd\n", i1->item_name,
i1->price, i1->total, i1->sold);
    }
    else//没找到
    {
        printf("Article doesn't exist!\n");
        system("pause");
        continue;
    }
    Item *i2 = query_articles(&s2.G1, in_item);//调用汇编程序
    printf("Shop2:\n");
    printf("name:%s\tprice:%hd\ttotal:%hd\tsold:%hd\n", i1->item_name,
i2->price, i2->total, i2->sold);
    system("pause");
}
else if (op == 2)
{
    printf("1. Shop1\n2. Shop2\n");
    int choice;
    scanf("%d", &choice);
    getchar();
    printf("Please input the article name you want to modify:");
    scanf("%s", in_item);
    if (choice == 1)
    {
        Item *i1 = query_articles(&s1.G1, in_item);//调用汇编程序
        if (i1)
        {
            modify_info(i1);//修改
        }
        else
        {
            printf("Article doesn't exist!\n");
            system("pause");
            continue;
        }
    }
    else if (choice == 2)
    {
        Item *i2 = query_articles(&s2.G1, in_item);//调用汇编程序
        if (i2)
        {
            modify_info(i2);//修改
        }
    }
}
```

汇编语言程序设计实验报告

```
        else
        {
            printf("Article doesn't exit!\n");
            system("pause");
            continue;
        }
    }
    else
    {
        continue;
    }
    system("pause");
}
else if (op == 3)
{
    avg_rate(&s1.G1, &s2.G1);
    avg_rate(&s1.G2, &s2.G2);
    avg_rate(&s1.G3, &s2.G3);
    system("pause");
}
else if (op == 4)
{
    rank(&s1, &s2);
    system("pause");
}
else if (op == 5)
{
    out_all_info(s1, s2);
    system("pause");
}
else if (op == 6)
{
    break;
}
else
{
    op = 0;
    continue;
}
}

}

else if (input_user_name[0] == '\n')//未登录
{
    int op;
    while (1)
    {
        system("cls");
        printf("\t\t-----menu-----\n");
        printf("\t\t 1 = inquire articles information\n");
        printf("\t\t 2 = exit\n");
        printf("\t\t-----\n");
        scanf("%d", &op);
        getchar();
        char in_item[10] = { '\0' };//初始化
        if (op == 1)
        {
```

汇编语言程序设计实验报告

```
        printf("Please input the article name you want to inquire:");
        scanf("%s", in_item);
        Item *i1 = query_articles(&s1.G1, in_item); //调用汇编程序
        if (i1)
        {
            printf("Shop1:\n");
            printf("name:%s\tprice:%hd\ttotal:%hd\tsold:%hd\n", i1->item_name,
i1->price, i1->total, i1->sold);
        }
        else //没找到
        {
            printf("Article doesn't exist!\n");
            system("pause");
            continue;
        }
        Item *i2 = query_articles(&s2.G1, in_item); //调用汇编程序
        printf("Shop2:\n");
        printf("name:%s\tprice:%hd\ttotal:%hd\tsold:%hd\n", i1->item_name,
i2->price, i2->total, i2->sold);
        system("pause");
    }
    else if (op == 2)
    {
        break;
    }
    else continue;
}
}
else
{
    printf("Failed to log in!\n");
    continue; //继续输入用户名密码
}
}
system("pause");
return 0;
}

void modify_info(Item *gal)
{
    short new_cost, new_price, new_total;
    char c; //清空缓冲区
    while (1)
    {
        printf("cost:%hd-->", gal->cost);
        int x = scanf("%hd", &new_cost);
        if (x) //判断输入合法性
        {
            gal->cost = new_cost;
            break;
        }
        if (x == 0)
        {
            while ((c = getchar()) != EOF && c != '\n'); //清空缓冲区
            continue;
        }
    }
}
```

汇编语言程序设计实验报告

```
}
while (1)
{
    printf("price:%hd-->", gal->price);
    int x = scanf("%hd", &new_price);
    if (x)//判断输入合法性
    {
        gal->price = new_price;
        break;
    }
    if (x == 0)
    {
        while ((c = getchar()) != EOF && c != '\n');
        continue;
    }
}
while (1)
{
    printf("total:%hd-->", gal->total);
    int x = scanf("%hd", &new_total);
    if (x)//判断输入合法性
    {
        gal->total = new_total;
        break;
    }
    if (x == 0)
    {
        while ((c = getchar()) != EOF && c != '\n');
        continue;
    }
}

}

void avg_rate(Item *gal, Item *gbl)
{
    gal->profit = (gal->price*gal->sold - gal->cost*gal->total)*100 / (2 *
gal->cost*gal->total)
    +(gbl->price*gbl->sold - gbl->cost*gbl->total)*100 / (2 * gbl->cost*gbl->total);
}

void rank(Shop *s1, Shop *s2)
{
    if (s1->G1.profit <= s1->G2.profit&& s1->G1.profit<=s1->G3.profit) s2->G1.profit = 3;
    if (s1->G2.profit <= s1->G1.profit&& s1->G2.profit <= s1->G3.profit) s2->G2.profit = 3;
    if (s1->G3.profit <= s1->G2.profit&& s1->G3.profit <= s1->G1.profit) s2->G3.profit = 3;
    if (s1->G1.profit >= s1->G2.profit&& s1->G1.profit >= s1->G3.profit) s2->G1.profit = 1;
    if (s1->G2.profit >= s1->G1.profit&& s1->G2.profit >= s1->G3.profit) s2->G2.profit = 1;
    if (s1->G3.profit >= s1->G2.profit&& s1->G3.profit >= s1->G1.profit) s2->G3.profit = 1;
    if (s2->G1.profit == 0)s2->G1.profit = 2;
    if (s2->G2.profit == 0)s2->G2.profit = 2;
    if (s2->G3.profit == 0)s2->G3.profit = 2;
}

void out_all_info(Shop s1, Shop s2)
```

汇编语言程序设计实验报告

```
{
    printf("Shop1:\n");
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \tprofit:%hd\n",
        s1.G1.item_name, s1.G1.cost, s1.G1.price, s1.G1.total, s1.G1.sold, s1.G1.profit);
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \tprofit:%hd\n",
        s1.G2.item_name, s1.G2.cost, s1.G2.price, s1.G2.total, s1.G2.sold, s1.G2.profit);
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \tprofit:%hd\n",
        s1.G3.item_name, s1.G3.cost, s1.G3.price, s1.G3.total, s1.G3.sold, s1.G3.profit);
    printf("Shop2:\n");
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \trank:%hd\n",
        s2.G1.item_name, s2.G1.cost, s2.G1.price, s2.G1.total, s2.G1.sold, s2.G1.profit);
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \trank:%hd\n",
        s2.G2.item_name, s2.G2.cost, s2.G2.price, s2.G2.total, s2.G2.sold, s2.G2.profit);
    printf("name:%s \tcost:%hd \tprice:%hd \ttotal:%hd \tsold:%hd \trank:%hd\n",
        s2.G3.item_name, s2.G3.cost, s2.G3.price, s2.G3.total, s2.G3.sold, s2.G3.profit);
}
```

3.2.4 实验步骤

- 1.修改汇编子程序，使其适合 win32 编程；编写 C 语言程序。
- 2.查找资料，研究在 visual studio 下如何进行 C 与汇编的混编。
- 3.准备上机实验环境，编辑，汇编，连接 c 语言与汇编文件。
- 4.在登录状态下测试函数是否调用汇编子程序成功。
- 5.测试其他部分功能是否能正确执行。
- 6.完成思考题

3.2.5 实验记录与分析

- 1.实验环境条件：P3 1GHz，256M 内存；win10 下 visual studio 2017 community
- 2.在 C 程序里调用汇编子程序具体实现方法：
 - (1) 创建一个空项目

汇编语言程序设计实验报告

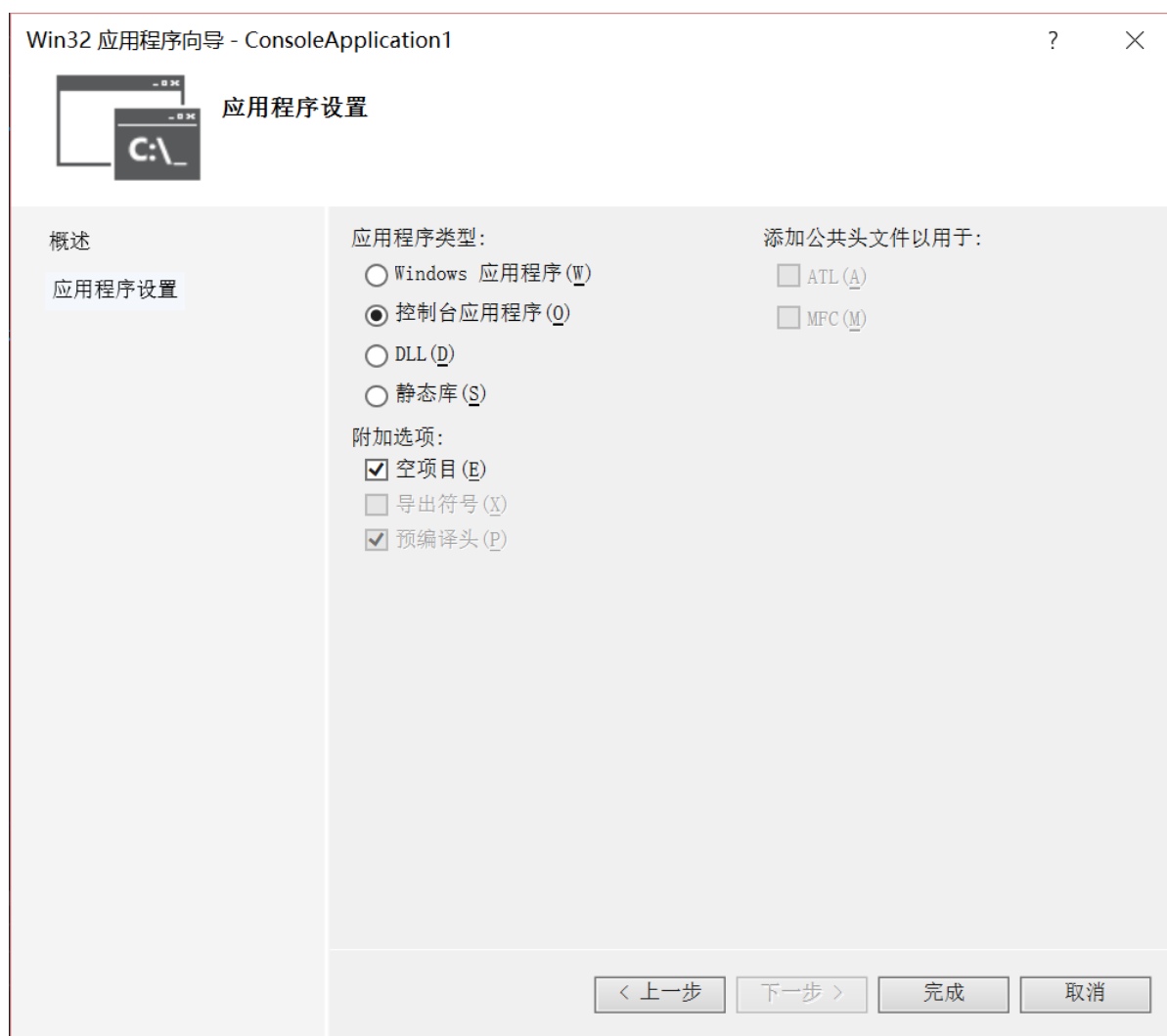


图 3.2.2 创建空项目

(2) 添加 cpp 文件以及 asm 文件



图 3.2.3 添加文件

(3) 生成依赖项

汇编语言程序设计实验报告

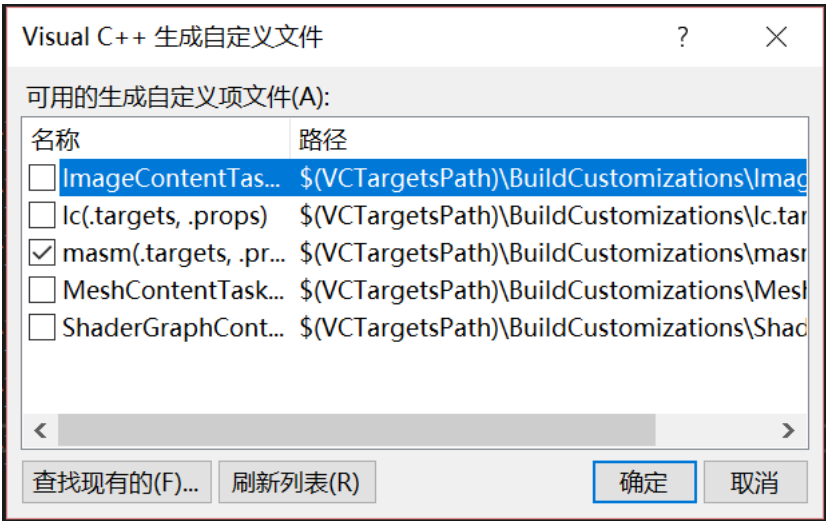


图 3.2.4 生成依赖项

(4) 设置环境变量（将 vs2017 编译命令所在目录放进 Path 中）

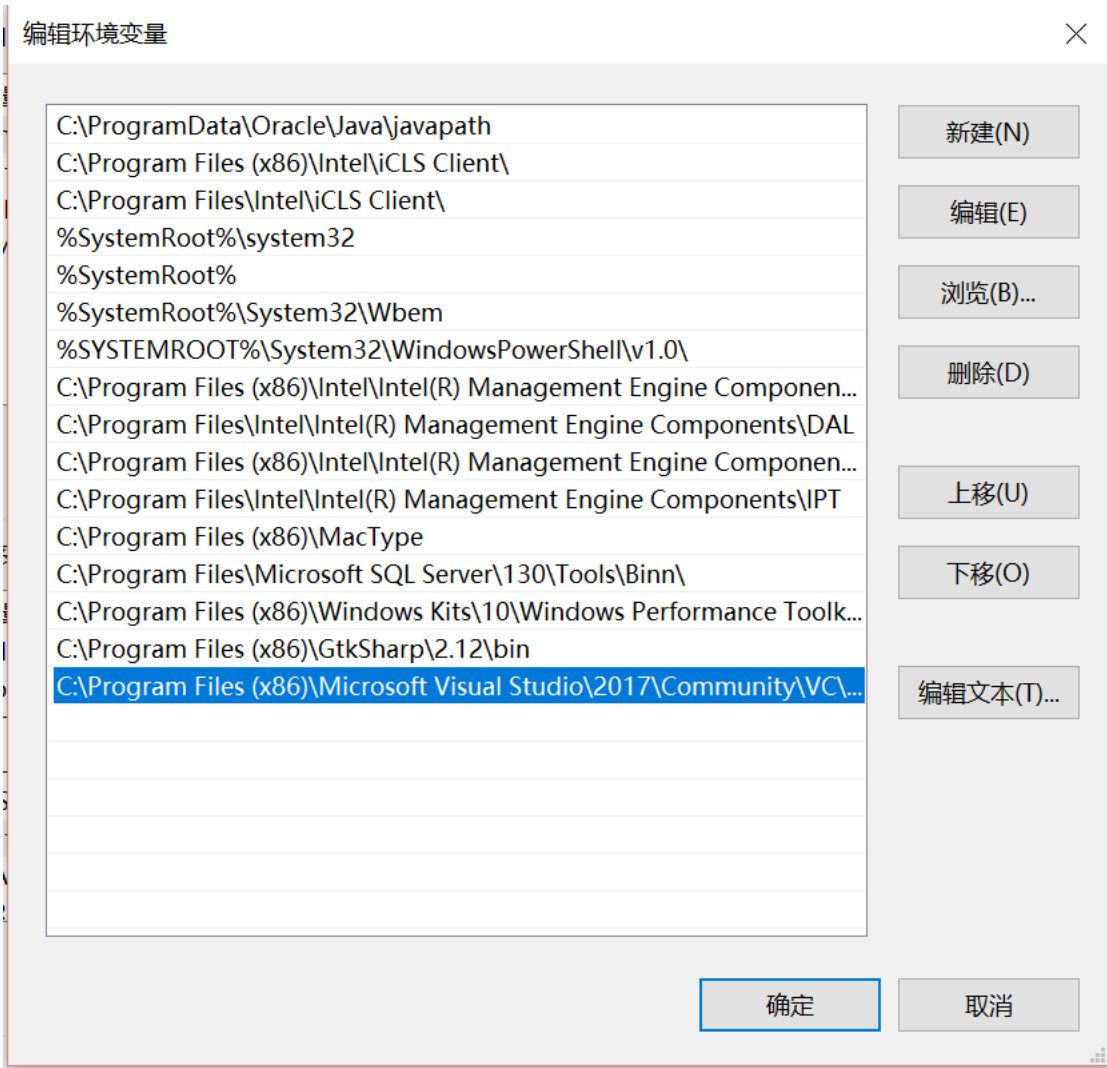


图 3.2.5 设置环境变量

(5) 设置 asm 文件属性

汇编语言程序设计实验报告

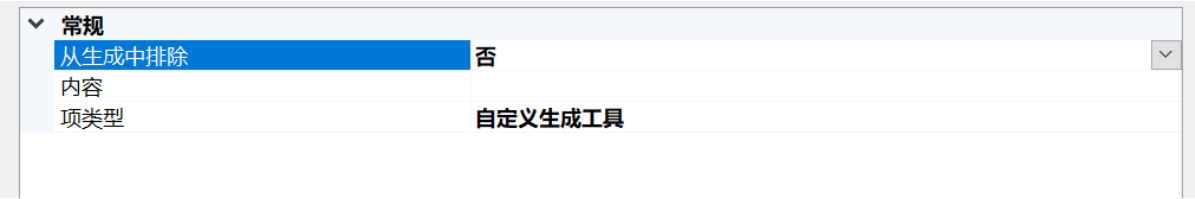


图 3.2.6 设置属性

命令行	ml /c /coff %(fileName).asm
说明	Performing Custom Build Tools
输出	%(fileName).obj;%(OutPuts)
附加依赖项	
链接对象	是
将输出视为内容	否
在以下操作之后执行	
在以下操作之前执行	

图 3.2.7 设置属性

(6) 对 asm 文件以及 cpp 文件进行编译，连接。

3. 编译过程发生异常。

(1) 在编译过程中发现无法对 asm 文件进行编译。

解决办法：查阅资料后发现是由于自己在之前没有设置环境变量，导致 vs2017 无法编译 asm 文件。

(2) 在编译 cpp 文件时发生错误如下：（文字说明）

fatal error LNK1190: 找到无效的链接地址信息,请键入 0x0001

解决办法：这个在当时查找了资料，发现还是无法解决问题。后来重新将上述步骤执行了一遍，发现这个错误消失了。猜测当时的汇编程序编译产生的 obj 文件出现问题。

4. 调试过程中发生异常。

(1) 调试调用汇编子程序时，发生无法访问的错误。

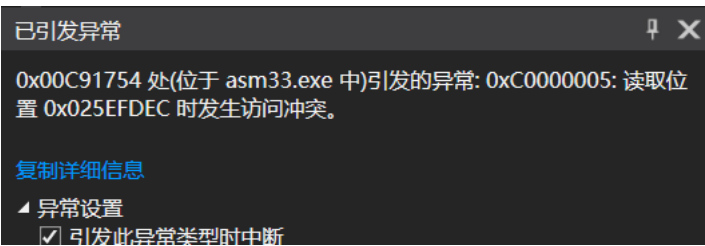


图 3.2.8 访问异常

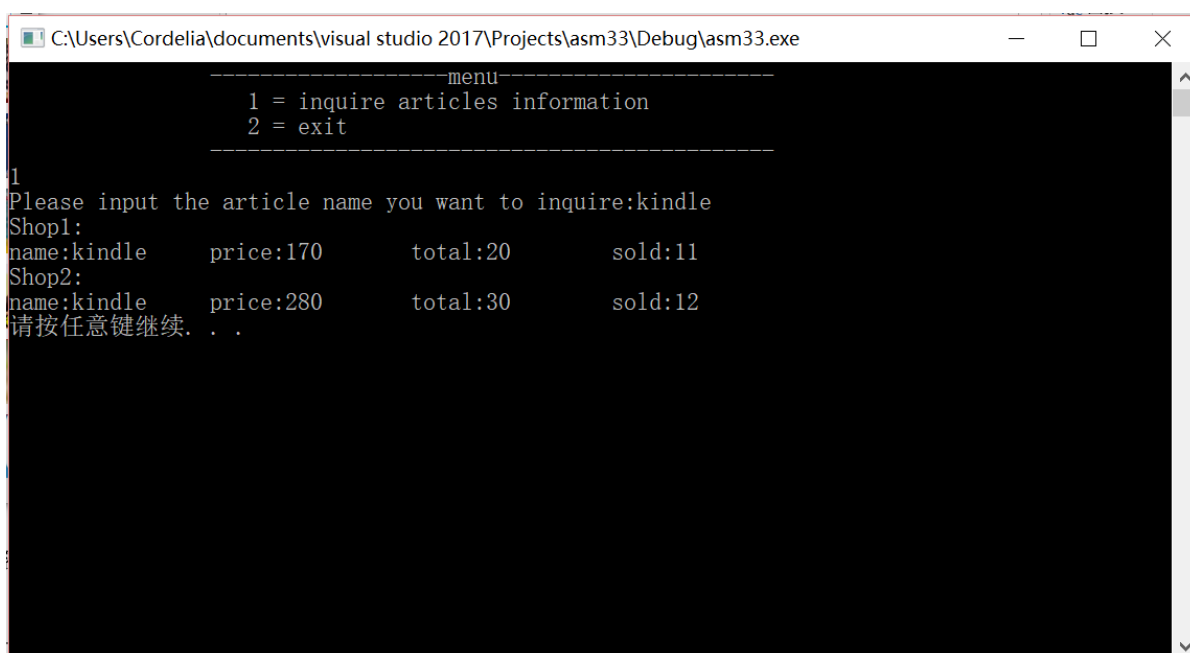
解决方法：经过自己判断，发现由于 asm 文件中寄存器部分没有改为 32 为寄存器，这使得一些汇编指令对寄存器操作时会出现错误，比如 mov 指令。

5. 未登录状态下测试菜单功能：

(1) 测试查询商品信息功能（调用汇编子程序实现）

输入要查询的商品名称 kindle，显示两个网店商品的名称，售价，进货数量，销售量。

汇 编 语 言 程 序 设 计 实 验 报 告



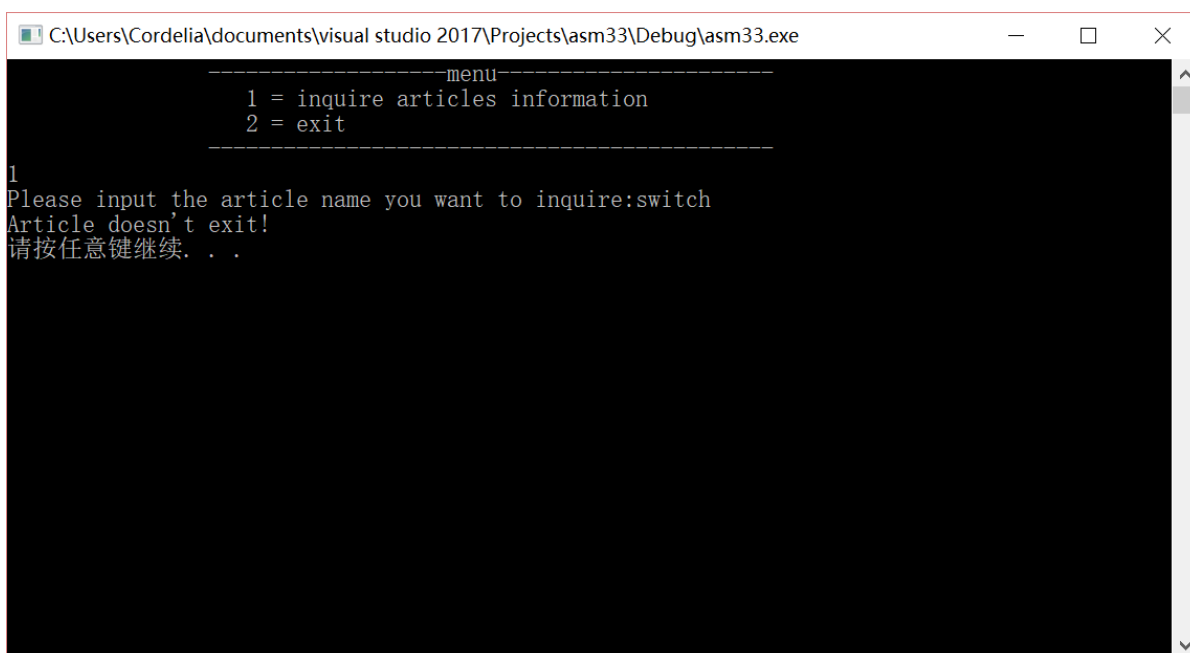
```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = exit
-----

1
Please input the article name you want to inquire:kindle
Shop1:
name:kindle      price:170      total:20      sold:11
Shop2:
name:kindle      price:280      total:30      sold:12
请按任意键继续. . .
```

图 3.2.9 查询商品信息功能

输入不存在的商品名称，提示商品不存在。



```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

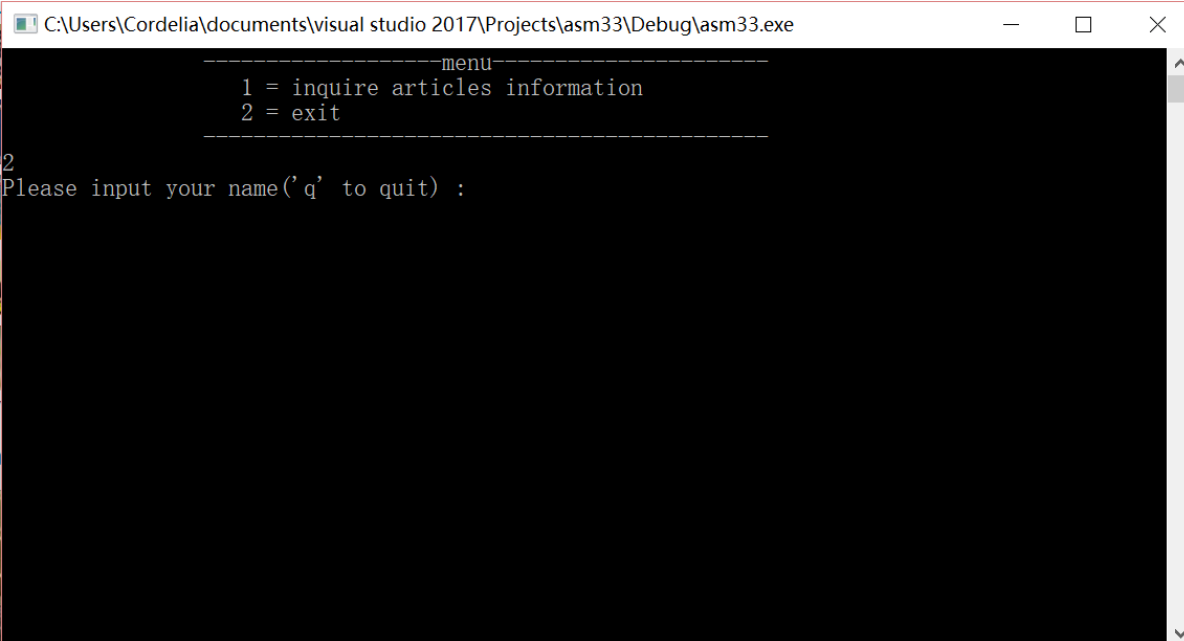
-----menu-----
1 = inquire articles information
2 = exit
-----

1
Please input the article name you want to inquire:switch
Article doesn't exist!
请按任意键继续. . .
```

图 3.2.10 查询商品信息功能，输入不存在商品情况

(2) 输入操作 2，程序退出。

汇编语言程序设计实验报告



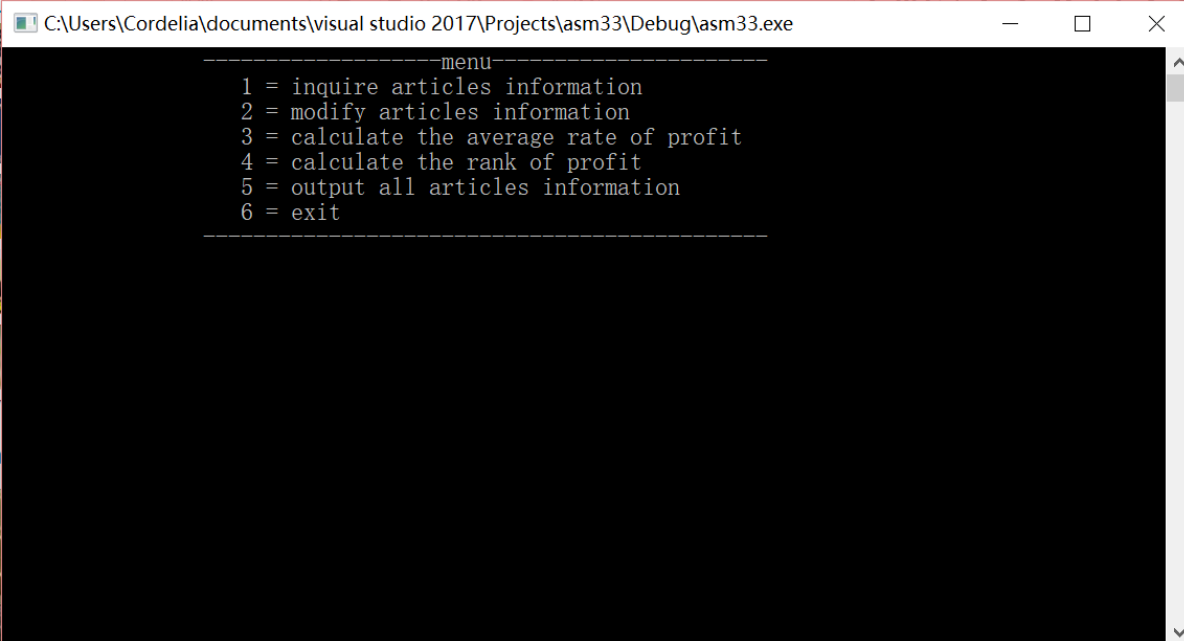
```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = exit
-----
2
Please input your name('q' to quit) :
```

图 3.2.11 程序退出

6. 登录状态下测试菜单功能

(1) 输入用户名，密码（登录验证用户名，密码的正确性在之前任务中已经完成过，此处不展示）



```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

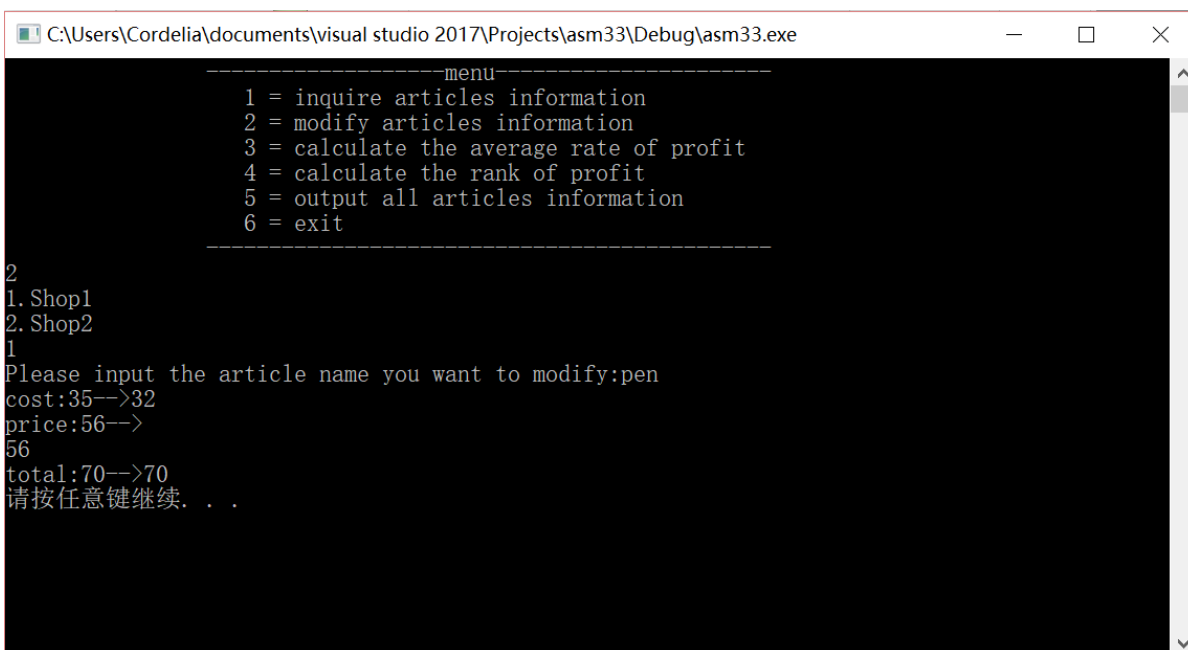
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
```

图 3.2.12 登录状态下菜单栏

(2) 测试修改商品信息功能（调用汇编程序查找）

在商店 1 中修改商品 pen。其中只修改进货价；售价，进货总数不变。

汇编语言程序设计实验报告



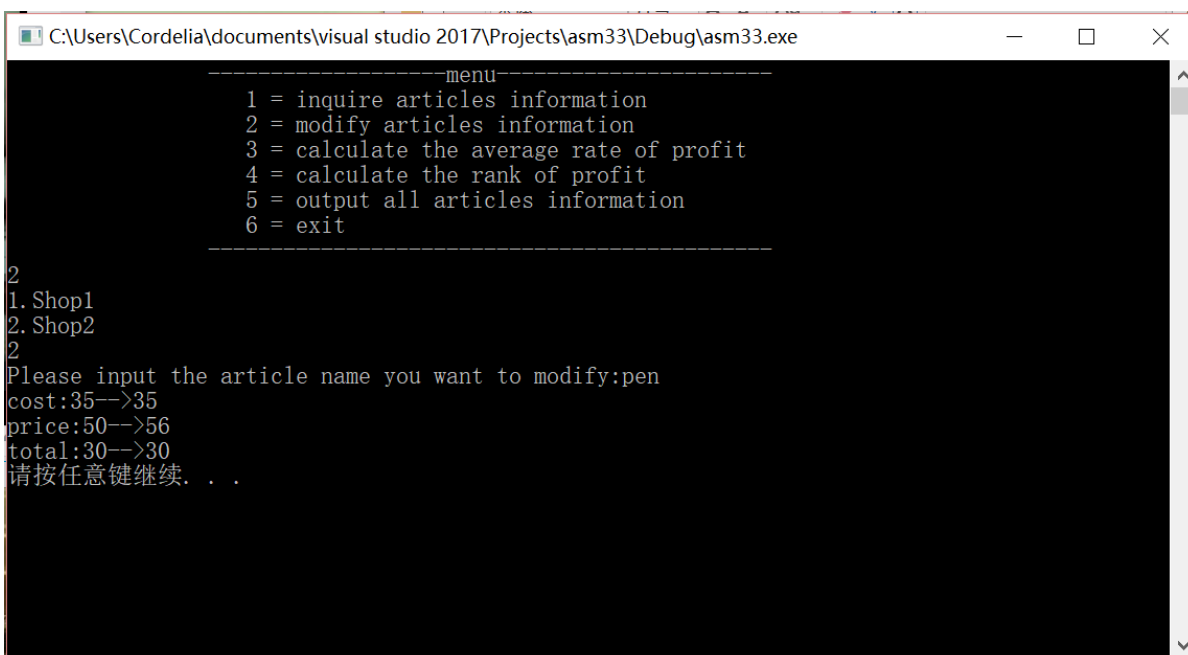
```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----

2
1. Shop1
2. Shop2
1
Please input the article name you want to modify:pen
cost:35-->32
price:56-->
56
total:70-->70
请按任意键继续. . .
```

图 3.2.13 修改商品信息功能

在商店 2 中修改商品 pen。只修改售价，其他不变。



```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

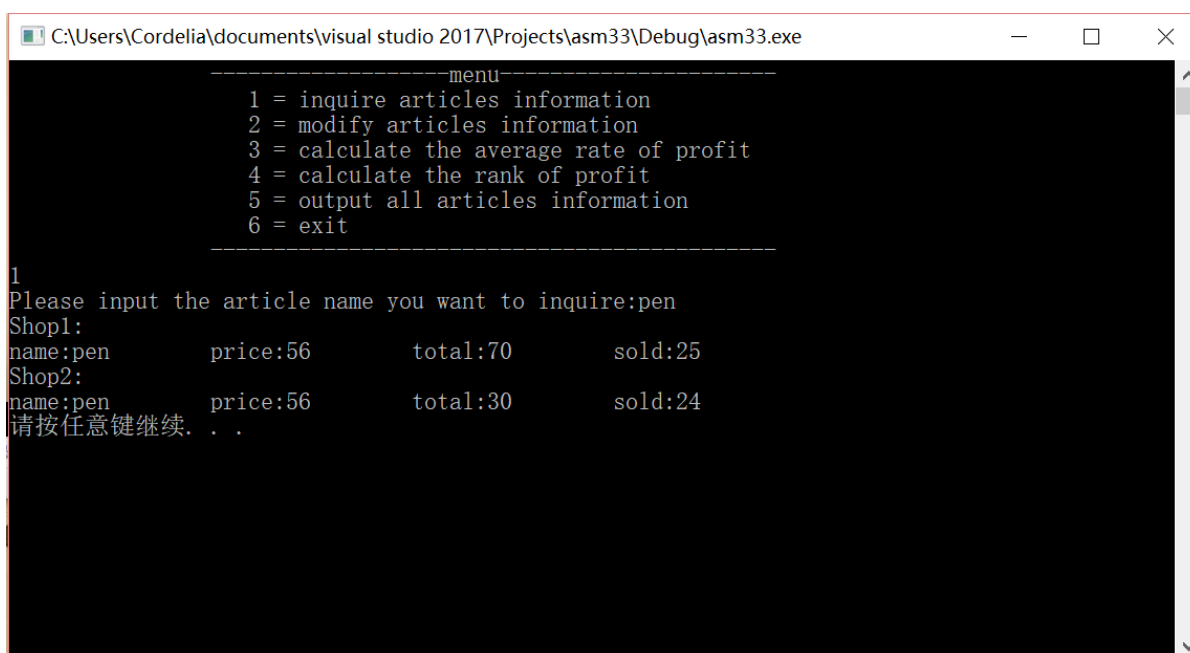
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----

2
1. Shop1
2. Shop2
2
Please input the article name you want to modify:pen
cost:35-->35
price:50-->56
total:30-->30
请按任意键继续. . .
```

图 3.2.14 修改商品信息功能

查询商品 pen 信息，确保修改成功。

汇编语言程序设计实验报告



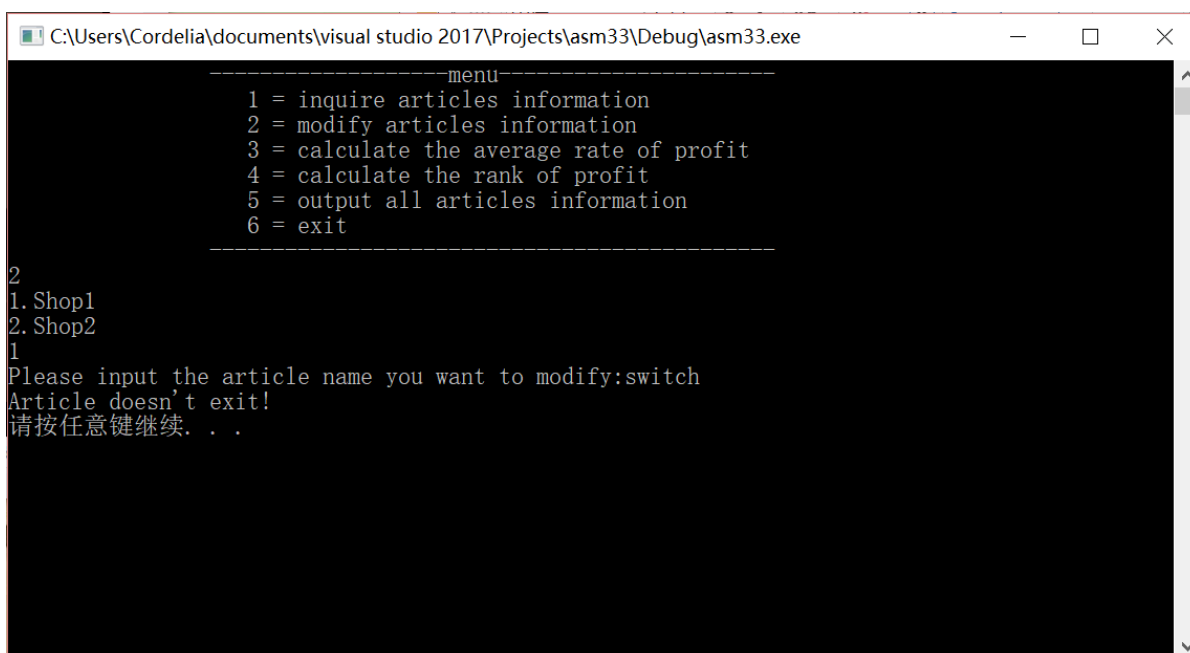
```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----

1
Please input the article name you want to inquire:pen
Shop1:
name:pen      price:56      total:70      sold:25
Shop2:
name:pen      price:56      total:30      sold:24
请按任意键继续. . .
```

图 3.2.15 确认修改成功

输入错误商品名称，提示商品不存在。



```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

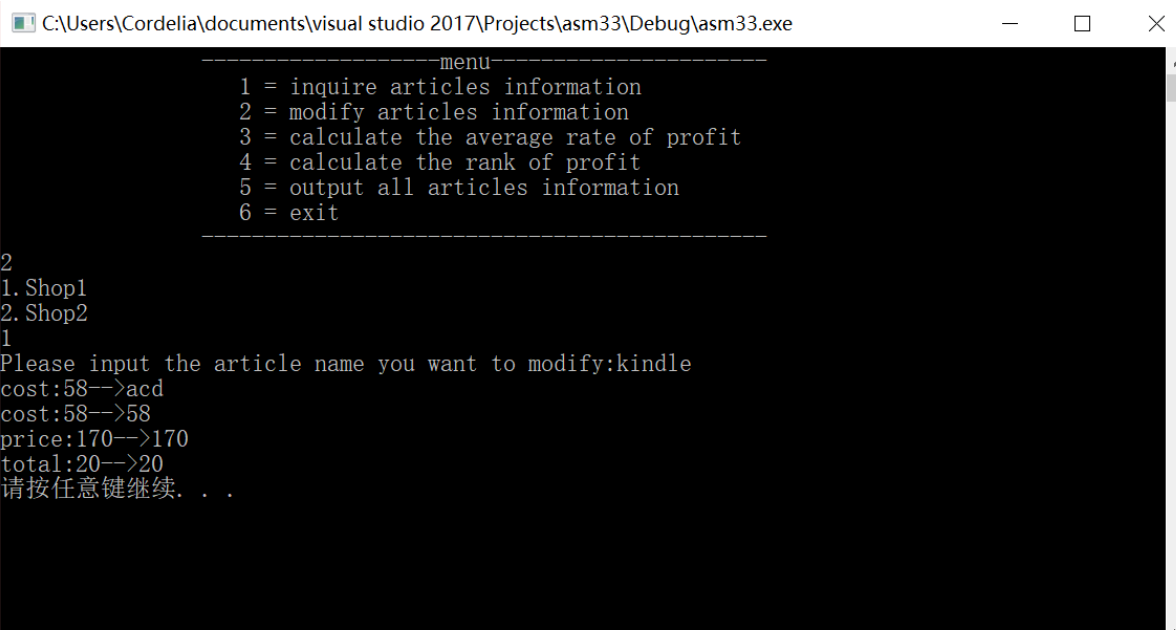
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----

2
1. Shop1
2. Shop2
1
Please input the article name you want to modify:switch
Article doesn't exist!
请按任意键继续. . .
```

图 3.2.16 输入错误商品名称

输入非法数值，重新提示输入。

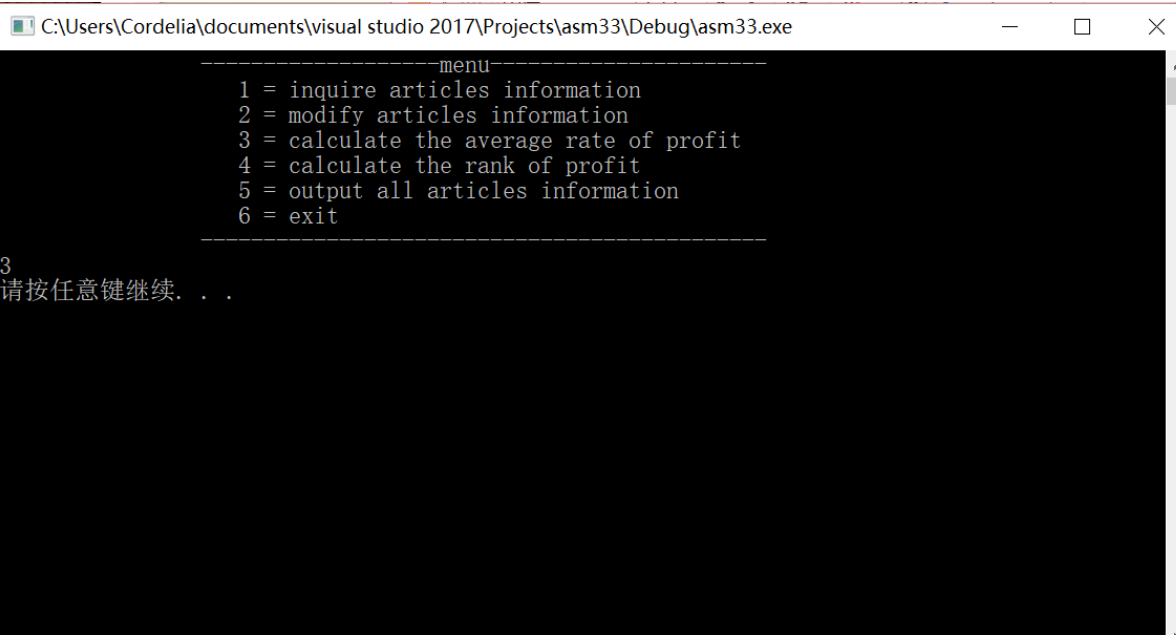
汇编语言程序设计实验报告



```
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
2
1. Shop1
2. Shop2
1
Please input the article name you want to modify:kindle
cost:58-->acd
cost:58-->58
price:170-->170
total:20-->20
请按任意键继续. . .
```

图 3.2.17 输入非法数值

(3) 计算平均利润率功能测试（由于此模块不由本人提供，在试图修改别人写的模块时无法顺利调用实现功能，改用 C 语言实现）

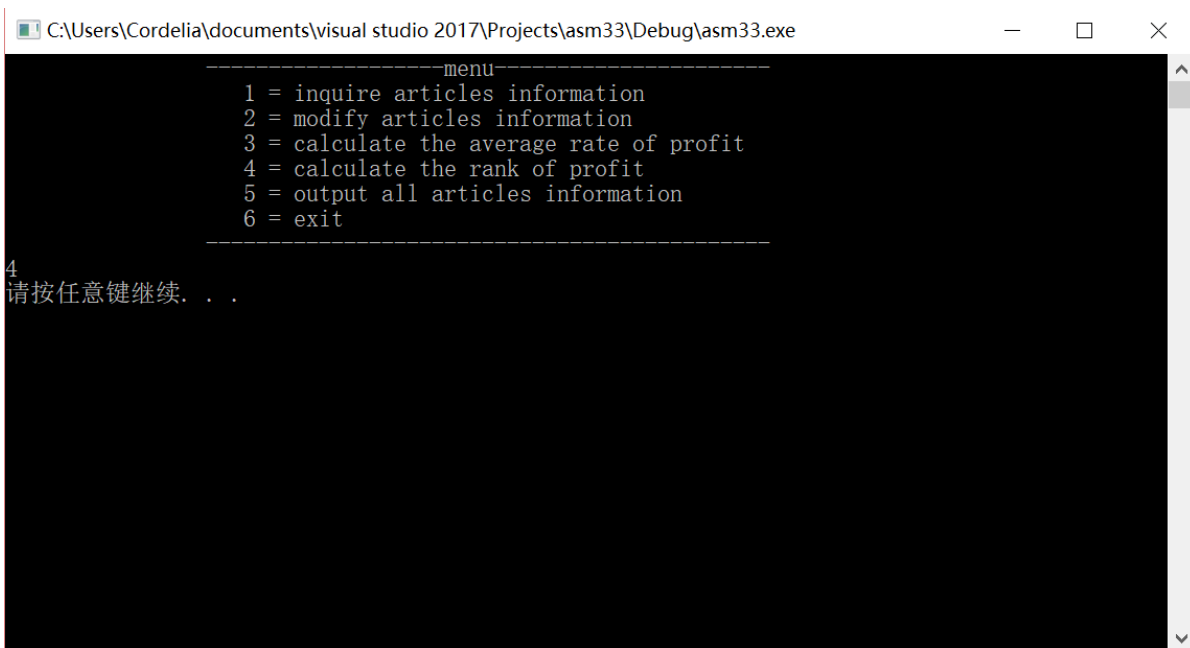


```
-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
3
请按任意键继续. . .
```

图 3.2.18 平均利润率功能测试

(4) 计算利润率排名功能测试（同上）

汇编语言程序设计实验报告



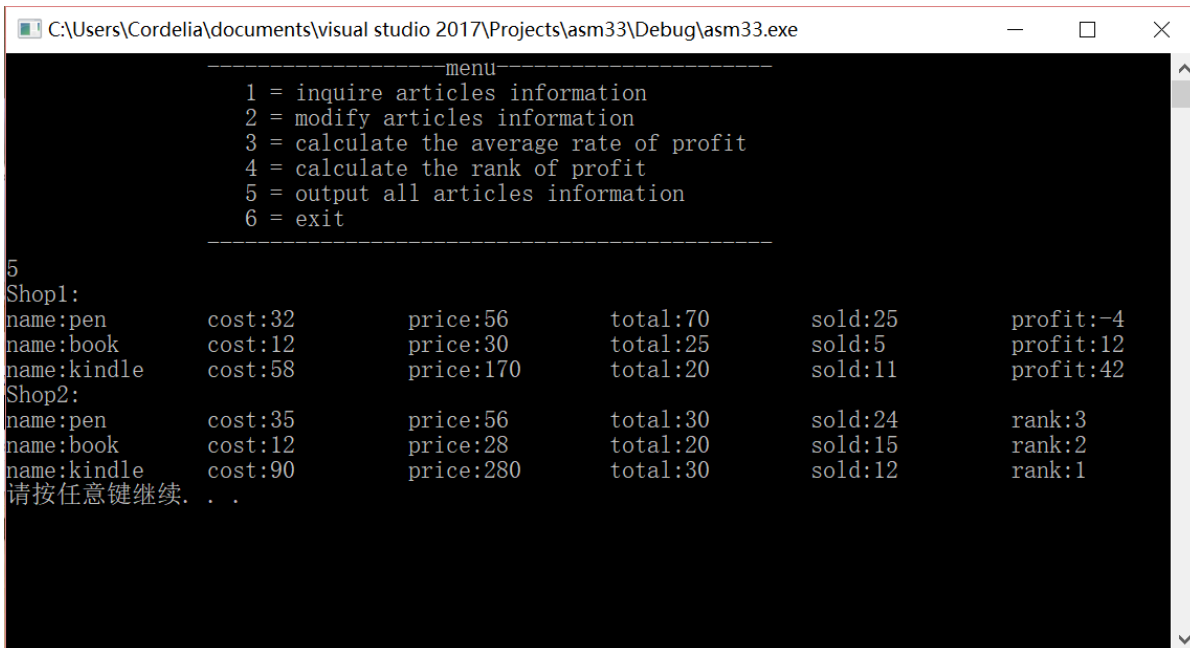
```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
4
请按任意键继续. . .
```

图 3.2.19 利润率排名功能测试

(5) 输出全部商品信息功能测试（用 C 语言实现）

利润率能正确计算，负数也能显示，排名正确。



```
C:\Users\Cordelia\documents\visual studio 2017\Projects\asm33\Debug\asm33.exe

-----menu-----
1 = inquire articles information
2 = modify articles information
3 = calculate the average rate of profit
4 = calculate the rank of profit
5 = output all articles information
6 = exit
-----
5
Shop1:
name:pen      cost:32      price:56      total:70      sold:25      profit:-4
name:book     cost:12      price:30      total:25      sold:5       profit:12
name:kindle   cost:58      price:170     total:20      sold:11      profit:42
Shop2:
name:pen      cost:35      price:56      total:30      sold:24      rank:3
name:book     cost:12      price:28      total:20      sold:15      rank:2
name:kindle   cost:90      price:280     total:30      sold:12      rank:1
请按任意键继续. . .
```

图 3.2.20 输出全部商品信息功能测试

7. 思考题

(1) 观察 C 语言编译器中对各种符号的命名规则（指编译器内部可以识别的命名规则，比如，符号名前面是否加下划线“_”，等），主、子程序之间参数传递的机制，通过堆栈传递参数后堆栈空间回收的方法（要设计一个有多个参数需要传递的 C 函数）。

Int main 函数反汇编出现的语句中有一条加了__双下划线

汇编语言程序设计实验报告

```
int main()
{
00962030 push     ebp
00962031 mov     ebp, esp
00962033 sub     esp, 39Ch
00962039 push     ebx
0096203A push     esi
0096203B push     edi
0096203C lea     edi, [ebp-39Ch]
00962042 mov     ecx, 0E7h
00962047 mov     eax, 0CCCCCCCCh
0096204C rep stos dword ptr es:[edi]
0096204E mov     eax, dword ptr [__security_cookie (096B004h)]
00962053 xor     eax, ebp
}
```

图 3.2.21 观察反汇编语句

调用 strcpy 函数时，反汇编语句前 call 指令后加了_单下划线。用 offset 指令后加了 string，猜测应该是类型指明。

```
strcpy(s1.name, "Shop1");
0096207A push     offset string "Shop1" (0968B44h)
0096207F lea     eax, [s1]
00962085 push     eax
00962086 call    _strcpy (09610E6h)
0096208B add     esp, 8
```

图 3.2.22 观察反汇编语句

调用自己写的 C 语言函数，发现并没有被加上下划线。

```
00962921 movs    word ptr es:[edi], word ptr [esi]
00962923 call    out_all_info (09612A8h)
```

图 3.2.23 观察反汇编语句

参数传递机制：

发现是通过栈来实现主程序，子程序之间参数传递。从后面位置的参数传递起，将地址送入 eax，依次入栈。然后调用子程序。最后 esp 指针加 8，推测是栈指针增加。

堆栈空间回收是由子程序完成的，主程序不完成此项功能。

```
rank(&s1, &s2);
009628C3 lea     eax, [s2]
009628C9 push     eax
009628CA lea     ecx, [s1]
009628D0 push     ecx
009628D1 call    rank (096130Ch)
009628D6 add     esp, 8
```

图 3.2.24 观察反汇编语句

(2) 对混合编程形成的执行程序，用调试工具观察由 C 语言形成的程序代码与由汇编语言形成的程序代码之间的相互关系，包括段、偏移的值，汇编指令访问 C 的变量时是如何翻译的等。

偏移值：（发现使用 offset string，后面括号里还加了很长一串 16 进制码，推测是新指令）

汇编语言程序设计实验报告

```
strcpy(s1.name, "Shop1");  
0096207A push      offset string "Shop1" (0968B44h)  
0096207F lea       eax, [s1]  
00962085 push      eax  
00962086 call     _strcpy (09610E6h)  
0096208B add       esp, 8
```

图 3.2.25 观察反汇编语句

段：发现多使用基址寻址，多用 ebx，ebp 等，其默认段分别是 ds，ss。

访问变量：

发现 C 访问变量运用了堆栈，ebp 作为基址寄存器，默认段为 ss，其寻址方式为变址寻址。将其扩展为定义时的类型。

```
if (op == 1)//调用汇编程序实现  
009625CC cmp      dword ptr [ebp-0E8h], 1
```

图 3.2.26 观察反汇编语句

(3) 请尝试在 C 语言源程序中不合理地嵌入汇编语言的指令语句，达到破坏 C 语言程序的正确性的目的。比如，在连续几条 C 语言语句中间加入一条修改 AX 寄存器（或 DS 等其他寄存器）的汇编指令语句，而 AX 的内容在此处本不该被修改，这样就可观察到破坏 C 语言程序正确性的效果。

经过多次尝试无果，但是通过看反汇编语句，发现一条 C 语言语句会反汇编成很多条汇编指令，若在这之间改变了一些寄存器的值，就会发生意想不到的错误。可以通过推理轻松得出。在 C 语言程序中，若不考虑上下语句翻译成怎样的机器码而随意嵌入汇编指令语句时，有可能存在出错的风险。

(4) 观察 C 编译器的优化策略对代码的影响。通过实际观察与分析，记录本实验中汇编语言程序的效率会优于 C 语言程序的实例。

在 C 语言反汇编成汇编语句时，发现 C 语言不会使用之前学到的优化技巧，比如 sar，sal 指令，这样会使 C 语言在效率上不如人为优化的汇编程序。

```
0096181D mov      edx, dword ptr [gb1]  
00961820 movsx   eax, word ptr [edx+0Ch]  
00961824 mov      edx, dword ptr [gb1]  
00961827 movsx   edx, word ptr [edx+10h]  
0096182B imul    eax, edx  
0096182E mov      edx, dword ptr [gb1]  
00961831 movsx   edx, word ptr [edx+0Ah]  
00961835 mov      esi, dword ptr [gb1]  
00961838 movsx   esi, word ptr [esi+0Eh]  
0096183C imul    edx, esi  
0096183F sub      eax, edx  
00961841 imul    eax, eax, 64h  
00961844 mov      edx, dword ptr [gb1]
```

图 3.2.27 观察反汇编语句

汇 编 语 言 程 序 设 计 实 验 报 告

```
00961847 movsx     esi,word ptr [edx+0Ah]
0096184B shl       esi,1
0096184D mov       edx,dword ptr [gb1]
00961850 movsx     edx,word ptr [edx+0Eh]
00961854 imul      esi,edx
00961857 cdq
00961858 idiv       eax,esi
0096185A add       ecx,eax
0096185C mov       eax,dword ptr [ga1]
0096185F mov       word ptr [eax+12h],cx
```

图 3. 2. 28 观察反汇编语句

汇编语言程序设计实验报告

4 总结与体会

在本次实验任务 1 中,最初的感觉就是在原来任务的基础上做一些修改,增加一些新功能就好了,但这次任务真正的重点是在模块化程序设计上面。是的,如果单就难度上来说,这次任务的功能实现并不是很难,但是如果把多人写作一起写的东西连接到一起的话就很容易出现问题。因为每个人的思维不一样,有时候你很可能没有考虑到这个模块的独立性,或许这个模块有很多限制,是基于你自己的数据来设计的,然后交给另一个人用的时候错误出在哪里都不知道。

在这次任务中,首先学到了宏指令的设计方法,在这次任务中将 dos9 号,10 号功能定义成宏指令,有利于代码简洁,不仅仅是书写方便,看起来也比较清楚。同时写养成了写子程序的习惯,子程序写起来可能会比直接在主程序中实现麻烦一点,但同时写子程序是一种优良的变成习惯,有利于将其变成模块,供其他程序使用。子程序的调用原理在实验记录里已经阐述过了。

在实现这次任务的功能时,我感受到了有时候两个功能描述一样的函数,在实现的时候绝对不能仅仅是简单地复制一下,而是要考虑它们在整体中的功能,比如这次的查询商品信息的功能。看起来都是一样的,但是如果不注意很可能就会出现这个问题,这在上面已经描述过了,在此不再赘述。在实现修改商品信息的功能时,遇到了要将二进制数转换成十进制数的要求,在参考了书上的代码实现之后,终于是完成了这个功能。同时在找到了商品位置后,怎样显示原有的商品信息也要好好地考虑一下。

在本次任务 2 中,最初感觉就是把原来写好的汇编程序改一下,然后就用 C 语言调用就好了,然而在自己实际操作过程中却不是一回事。

在本次任务 2 开始的时候,首先就将原来的汇编程序改成了 win32 下适用的程序。然后写好了 C 语言程序。由于自己使用的 IDE 是 visual studio 2017 community,于是就上网查找相关资料,研究如何在自己的开发环境下调用汇编程序,经过查找了大量的资料,还是发现自己出现了一些错误,具体细节可以在实验步骤里看到,最后终于是成功了。然后在调试过程中发现自己修改的别人的模块出了问题,于是就改用 C 语言实现功能。

在本次试验,最大的收获就是知道了在 C 语言内可以嵌入汇编语言,可能汇编写起来不是很方便,但在 C 语言程序的一些核心步骤上使用汇编代替,可以极大提高 C 语言的效率,这也是优化代码的很好方法。仅仅是简单的替换,就可以使代码运行速度加快,这在之前是没有想到的。让我意识到了汇编其实是有实际应用的。而在混编的调试下,其操作比纯粹汇编调试要复杂一些,这也是不好的地方。

在观察 C 语言编译器反汇编后的语句时,可以发现有些库函数调用时,其前面都加了_单下划线,调用自己写的函数时,并没有加下划线。而且也学到了 C 语言主程序与子程序之间参数传递其实使用的是堆栈,自己仔细想了下,发现这样的参数传递机制是很合理的。自己写的 C 语言程序经过反汇编,最后都有保护寄存器的功能,这与平时我们写汇编时的要求是一样的。

在 C 语言程序中,访问变量也使用了堆栈,而且多使用变址寻址方式。如果在连续的 C 语言语句中插入一些汇编语句,要考虑会不会影响原来的 C 语言语句的运行。而这是很难做到的。

本次上机主要是首次感受模块化程序设计,体验一下多人合作共同完成一个项目。在这次任务中,重点是要培养合作的精神,同时在合作时,各自要做好自己的部分,本着对自己工作负责的原则,一定要将自己写的模块进行测试,然后才能交到别人手上,这样团队合作才会效率高。同时在

汇 编 语 言 程 序 设 计 实 验 报 告

团队合作的时候有利于培养自己多角度的观察，要站在不知情的人的角度，别人是如何来使用你这个程序的，要说明一下模块的基本功能以及使用注意事项。

在任务 2 上机主要是让我们明白不同的编程语言是可以协同解决一个问题的，而且可以利用不同语言的特点来更好地解决问题，利用汇编语言的知识，能够更好地理解高级语言的内部处理原理与策略，为编写更好的 C 语言程序、用好 C 编译器提供支持。比如 C 语言方便，好写，可以在一些输入输出函数上使用 C 语言来写，而汇编的优势在于块，可以在一些核心步骤上使用汇编语言编写，比如乘除功能。这对代码的优化有着巨大的帮助。而 C 语言语句经过反汇编，可以帮助我们理解高级语言的处理策略，为之后我们编写更好地 C 语言程序，用好 C 编译器提供了基础。在本次上机主要是拓展我们的视野，让我们知道还有很多方法是可以运用在编程中的。

汇 编 语 言 程 序 设 计 实 验 报 告

参考文献

- [1] 王元珍、韩宗芬、曹忠升.《80X86 汇编语言程序设计》. 华中科技大学出版社:2005 年 04 月
- [2] 许向阳,《80X86 汇编语言程序设计上机指南》“第十一章 汇编语言程序与 C 程序的连接”。
- [3] C 程序调用汇编语言函数操作说明.doc (教学网站实验指导)