

重 庆 大 学

学 生 实 验 报 告

实验课程名称 汇编语言程序设计

开课实验室 DS1502 机房

学 院 软件学院 年 级 软件工程 专 业 班 01

学 生 姓 名 _____ 学 号 _____

开 课 时 间 2023 至 2024 学 年 第 1 学 期

总 成 绩	
教师签名	陈蜀宇

大数据与软件工程学院 制

《汇编语言程序设计》实验报告

开课实验室： DS1502 机房

2023 年 10 月 12 日

学院	大数据与软件学院	年级、专业、班	23 级软 01 班	姓名		成绩	
课程名称	汇编语言程序设计	实验项目名称	实验三：伪操作，表达式及程序框架 (涉及章节：第四章)		指导教师		陈蜀宇
教师评语	<div>教师签名：陈蜀宇</div> <div>2023 年 月 日</div>						

一【实验目的】

1. 掌握指令，伪操作，表达式和程序框架。

二【实验环境】

- PC微机；DOS操作系统或 Windows 操作系统；
- MASM.EXE, LINK.EXE, DEBUG.COM或宏汇编集成环境。
- DOSBOX.EXE（64 位 Windows 操作系统需要）。

DOSBOX.EXE 使用方法

1.安装；

2.运行 DOSBOX.EXE 程序；

3.在输入框状态下 Z:\> 输入： mount C D:\masm

说明：“Z:\>”这是提示符 “C”作为虚拟 C 盘，“D:\masm”是要虚拟的文件夹位置；简洁的讲，把要虚拟的文件夹位置换掉上面的 D:\masm；

4.在刚才的提示符下输入 C:，这样就切换到虚拟的 c 盘，也就是你要的目录；

5.按照 DOS 操作系统环境要求运行 C: 目录下的 MASM.EXE, LINK.EXE, DEBUG.COM 等软件。如：C:\>MASM 123.ASM、C:\>LINK 123.OBJ、C:\>debug 123.exe 等。

三【实验要求】

1. 阅读数据传送指令、算术运算指令、逻辑指令、数据串传送程序和数据串传送指令 MOVS、STOS 及重复前缀 REP 的内容、串操作、逻辑指令、控制转移指令等。
2. 理解伪操作，表达式和程序框架。

四【预备知识】

汇编语言程序格式，指令，伪操作，表达式，程序框架。

五【实验内容】

1. 通过调试下列程序段并记录它将实现的功能，注意观察伪代码的应用：

1) 程序一（无堆栈段）：

```
MYDATA    SEGMENT
            GRAY DB  18H, 34H, 05H, 06H, 09H
            DB  0AH, 0CH, 11H, 12H, 14H
MYDATA    ENDS

MYCODE     SEGMENT
ASSUME     CS: MYCODE, DS: MYDATA
GO:        MOV     AX, MYDATA
            MOV     DS, AX
            MOV     BX, OFFSET GRAY
            MOV     CX, 2
CYCLE:     IN  AL, 21H    ;从21H号端口输入8位数据到AL寄存器；
            XLAT     GRAY
            AND     AL, 7FH    ; 输入数据的最高位置0；
            OUT     20H, AL    ; 将置0后的数据从21H号端口输出
            LOOP    CYCLE
```

```
RET
MYCODE    ENDS
END        GO
```

2) 程序二（有堆栈段）：

```
data segment
    dw  0123h,0456h,0789h,0abch,0defh,0fedh,0cbah,0987h
data ends

stack segment
    dw  0,0,0,0,0,0,0,0,0,0,0
stack ends

code segment
assume  cs:code,ds:data,ss:stack
start:  mov ax,stack
        mov ss,ax
        mov sp,20h ;设置栈顶SS:SP指向stack:20
        mov ax,data
        mov ds,ax ;ds指向data段
        mov bx,0 ;ds:bx指向data段中的第一个单元
        mov cx,8
s:      push [bx]
        add bx,2
        loop s ;以上将data段中的0~15单元中的8个字型数据依次入栈
        mov bx,0
        mov cx,8
s0:     pop [bx]
        add bx,2
        loop s0 ;以上依次出栈8个字型数据到data段中的0~15单元中
        mov ax,4c00h
        int 21h
        code ends
        end start
```

2. 按照第一题的格式要求，编写 $99+97+\dots+3+1$ 的累加和程序并调试通过，并记录程序执行结束后CX的值是多少？

3. 按照第一题的格式要求，编写如下程序并调试通过。

1) 把从LIST 到LIST+316中的字符串传送到BLK到BLK+316 中去；

2) 在BLK到BLK+100中查找字符“cqu”字符串，并记下“CQU”字符串出现的起始位置到BX中。

说明，LIST 到LIST+100的内容（from: <http://study.cqu.edu.cn/IndexOfEnglish/Index>）为：

Chongqing University (CQU) is a key national university and a member of the “Excellence League”, located in Chongqing, Southwest China. It is also one of the “211

Project" and "985 Project" universities with full support in construction and development from the central government and Chongqing Municipal Government.

五【实验结果及分析】

1.

(1)

21 号端口返回的是 f8，也就是 248，此时已经超过了 gray 数组的索引，所以不能正确返回 gray 的值；但是也可以将 al 赋值为 1，此时就可以 al 返回第二个数据 34h

<pre>AX=0768 BX=0000 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 S=076A ES=075A SS=0769 CS=076B IP=0005 NU UP EI PL 76B:0005 BB0000 MOV BX,0000 -t</pre>	<pre>文件 编辑 查看 mydata segment gray db 18h,34h,05h,06h,09h db 0ah,0ch,11h,12h,14h mydata ends mycode segment assume cs:mycode,ds:mydata go: mov ax,mydata mov ds,ax mov bx,offset gray mov cx,2 cycle: in al,21h xlat and al,7fh out 20h,al loop cycle ret mycode ends end go</pre>
<pre>AX=076A BX=0000 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 S=076A ES=075A SS=0769 CS=076B IP=0008 NU UP EI PL 76B:0008 B90200 MOV CX,0002 -t</pre>	
<pre>AX=076A BX=0000 CX=0002 DX=0000 SP=0000 BP=0000 SI=0000 S=076A ES=075A SS=0769 CS=076B IP=000B NU UP EI PL 76B:000B E421 IN AL,21 -t</pre>	
<pre>AX=07F8 BX=0000 CX=0002 DX=0000 SP=0000 BP=0000 SI=0000 S=076A ES=075A SS=0769 CS=076B IP=000D NU UP EI PL 76B:000D D7 XLAT -t</pre>	
<pre>AX=0702 BX=0000 CX=0002 DX=0000 SP=0000 BP=0000 SI=0000 S=076A ES=075A SS=0769 CS=076B IP=000E NU UP EI PL 76B:000E 247F AND AL,7F -t</pre>	
<pre>AX=0734 BX=0000 CX=0002 DX=0000 SP=0000 BP=0000 SI=0000 DS=076A ES=075A SS=0769 CS=076B IP=0010 NU UP EI PL 076B:0010 E2F9 LOOP 000B -t</pre>	
<pre>AX=0734 BX=0000 CX=0001 DX=0000 SP=0000 BP=0000 SI=0000 DS=076A ES=075A SS=0769 CS=076B IP=000B NU UP EI PL 076B:000B B001 MOV AL,01 -t</pre>	
<pre>AX=0701 BX=0000 CX=0001 DX=0000 SP=0000 BP=0000 SI=0000 DS=076A ES=075A SS=0769 CS=076B IP=000D NU UP EI PL 076B:000D D7 XLAT -t</pre>	
<pre>AX=0734 BX=0000 CX=0001 DX=0000 SP=0000 BP=0000 SI=0000 DS=076A ES=075A SS=0769 CS=076B IP=000E NU UP EI PL 076B:000E 247F AND AL,7F -t</pre>	<pre>mydata ends mycode segment assume cs:mycode,ds:mydata go: mov ax,mydata mov ds,ax mov bx,offset gray mov cx,2 cycle: mov al,1 xlat and al,7fh loop cycle ret mycode ends end go</pre>

(2)

在 s 中的 loop 循环依次 data 中的八个数据入栈，在下图第一行中，依次是 0123h,0456h...0987h

```

C:\>debug lab31.exe
-d ds:0000
075A:0000  CD 20 FF 9F 00 EA FF FF-AD DE 4F 03 A3 01 BA 03  . . . . .
075A:0010  A3 01 17 03 A3 01 92 01-01 01 01 00 02 FF FF FF  . . . . .
075A:0020  FF FF FF FF FF FF FF-FF FF FF FF 50 07 4C 01  . . . . .
075A:0030  63 06 14 00 18 00 5A 07-FF FF FF FF 00 00 00 00  c . . . . Z . . . .
075A:0040  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00  . . . . .
075A:0050  CD 21 CB 00 00 00 00 00-00 00 00 00 00 00 00  . ! . . . .
075A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00  . . . . .
075A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00  . . . . .

```

```

s:
push [bx]
add bx,2
loop s
mov bx,0
mov cx,8

s0:
pop [bx]
add bx,2
loop s0
mov ax,4c00h
int 21h
code ends

```

```

AX=076A BX=0010 CX=0000 DX=0000 SP=0010 BP=0000 SI=0000 DI=0000
DS=076A ES=075A SS=076B CS=076D IP=001A  NV UP EI PL NZ AC PO NC
076D:001A BB0000      MOV     BX,0000
-d ds:0000
076A:0000  23 01 56 04 89 07 BC 0A-EF 0D ED 0F BA 0C 87 09  #.U. . . . .
076A:0010  00 00 00 00 00 00 6A 07-00 00 1A 00 6D 07 A3 01  . . . . j . . . . m . .
076A:0020  87 09 BA 0C ED 0F EF 0D-BC 0A 89 07 56 04 23 01  . . . . . U . # .
076A:0030  B8 6B 07 8E D0 BC 20 00-B8 6A 07 8E D8 BB 00 00  . k . . . . j . . . .
076A:0040  B9 08 00 FF 37 83 C3 02-E2 F9 BB 00 00 B9 08 00  . . . . 7 . . . .
076A:0050  8F 07 83 C3 02 E2 F9 B8-00 4C CD 21 8A 47 0C 2A  . . . . . L ! . G . *
076A:0060  E4 40 50 8B C3 8C C2 05-0C 00 52 50 E8 C1 48 83  . @ P . . . . . R P . . H .
076A:0070  C4 04 50 8D 86 FA FE 50-E8 17 73 83 C4 06 8B B6  . . P . . . . P . . s . .

```

指向stack:20

然后在执行 s0 之前让 bx=0000, cx=8,使得下一个 loop 循环正常运行
 在第二个 loop 中让数据依次出栈，比如最后一次是最先入栈的数据 1023h

```

code segment
assume cs:code,ds:data,ss:stack
start:
mov ax,stack
mov ss,ax
mov sp,20h
mov ax,data
mov ds,ax
mov bx,0
mov cx,8
s:
push [bx]
add bx,2
loop s
mov bx,0
mov cx,8
s0:

```

```

076D:0027 BB004C      MOV     AX,4C00
-d ds:0010
076A:0010  00 00 00 00 00 00 6A 07-6A 07 6A 07 6A 07 6A 07  . . . . .
076A:0020  6A 07 6A 07 6A 07 6A 07-00 00 27 00 6D 07 A3 01  . j . j . j .
076A:0030  B8 6B 07 8E D0 BC 20 00-B8 6A 07 8E D8 BB 00 00  . k . . . .
076A:0040  B9 08 00 FF 37 83 C3 02-E2 F9 BB 00 00 B9 08 00  . . . . 7 .
076A:0050  8F 07 83 C3 02 E2 F9 B8-00 4C CD 21 8A 47 0C 2A  . . . . .
076A:0060  E4 40 50 8B C3 8C C2 05-0C 00 52 50 E8 C1 48 83  . @ P . . . .
076A:0070  C4 04 50 8D 86 FA FE 50-E8 17 73 83 C4 06 8B B6  . . P . . .
076A:0080  FA FE 81 E6 FF 00 C6 82-FB FE 00 2B C0 50 8D 86  . . . . .
-d ds:000e
076A:0000  23 01  . . . .
076A:0010  00 00 00 00 00 00 6A 07-6A 07 6A 07 6A 07 6A 07  . . . . .
076A:0020  6A 07 6A 07 6A 07 6A 07-00 00 27 00 6D 07 A3 01  . j . j . j .
076A:0030  B8 6B 07 8E D0 BC 20 00-B8 6A 07 8E D8 BB 00 00  . k . . . .
076A:0040  B9 08 00 FF 37 83 C3 02-E2 F9 BB 00 00 B9 08 00  . . . . 7 .
076A:0050  8F 07 83 C3 02 E2 F9 B8-00 4C CD 21 8A 47 0C 2A  . . . . .
076A:0060  E4 40 50 8B C3 8C C2 05-0C 00 52 50 E8 C1 48 83  . @ P . . . .
076A:0070  C4 04 50 8D 86 FA FE 50-E8 17 73 83 C4 06 8B B6  . . P . . .
076A:0080  FA FE 81 E6 FF 00 C6 82-FB FE 00 2B C0 50  . . . .

```

```

2.
data segment
    as dw 0h
data ends

code segment
assume cs:code,ds:data
start:
mov ax,0000h
mov cx,50
mov as,0001h

```

```

s:
add ax,[as]
add as,02h
loop s
mov bx,0000
code ends
end start

```

最终答案是 2500，转化为十六进制所以 ax=09c4

注意 cx 赋值的时候明确是十进制还是十六进制，避免累加次数不对

```

data segment
    as dw 0h
data ends

code segment
    assume cs:code,ds:data
start:
    mov ax,0000h
    mov cx,50
    mov as,0001h

s:
    add ax,[as]
    add as,02h
    loop s
    mov bx,0000
    code ends
end start

```

```

Copyright (C) Microsoft Corp 1983-1987. All Rights Reserved.

Run File [LAB32.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

C:\>debug lab32.exe
-u
076B:0000 B80000      MOV     AX,0000
076B:0003 B93200      MOV     CX,0032
076B:0006 C70600000100  MOV     WORD PTR [0000],0001
076B:000C 03060000      ADD     AX,[0000]
076B:0010 8306000002      ADD     WORD PTR [0000],+02
076B:0015 E2F5        LOOP    000C
076B:0017 BB0000      MOV     BX,0000
076B:001A C404        LES     AX,[SI]
076B:001C 3DFFFF      CMP     AX,FFFF
076B:001F 7403        JZ      0024
-g 0017

AX=09C4 BX=0000 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=075A ES=075A SS=0769 CS=076B IP=0017  NU UP EI PL NZ NA PE NC
076B:0017 BB0000      MOV     BX,0000

```

3.

(1)

```

data segment
list db 'Chongqing University (CQU) is a key national university and a member of the "Excellence League", located in Chongqing, China. It is also one of the "211 Project" and "985 Project" universities with full support in construction and development from the central government and Chongqing Municipal Government.'
blk db 0
db 0
db 0
len equ $-list-1
data ends

code segment
    assume cs:code,ds:data
start:
    mov cx,len
    lea si,list
copy:
    mov cl,byte ptr [si]
    mov blk,cl
    inc si
    dec cx
    cmp cx,0
    jnz copy
    code ends
end start

```

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
077F:0014 867EFF      XCHG    BH,[BP-01]
077F:0017 8BF8        MOV     DI,AX
077F:0019 803C3B      CMP     BYTE PTR [SI],3B
077F:001C 7405        JZ      0023
077F:001E 803C00      CMP     BYTE PTR [SI],00
-u
077F:0021 7545        JNZ     0068
077F:0023 8D067EFF    LEA     AX,[BP+FF7E]
077F:0027 3BC7        CMP     AX,DI
077F:0029 7342        JNB     006D
077F:002B 8BC7        MOV     AX,DI
077F:002D 2BC5        SUB     AX,BP
077F:002F 0402        ADD     AL,02
077F:0031 8B067EFF    MOV     [BP+FF7E],AL
077F:0035 803D3A      CMP     BYTE PTR [DI],3A
077F:0038 741B        JZ      0065
077F:003A B05C00      MOV     AX,005C
077F:003D 50          PUSH    AX
077F:003E 8D067EFF    LEA     AX,[BP+FF7E]
-g 7545

AX=8327 BX=7500 CX=0000 DX=FF7F SP=FFFE BP=0000 SI=0000 DI=0000
DS=075A ES=0003 SS=0769 CS=077F IP=FEA5  NU UP EI PL NZ NA PE NC
077F:FEA5 041C        ADD     AL,1C

```

(2)


```

data segment
    str db 'Chongqing University (CQU) is a key national university and a
member of the "Excellence League", located in Chongqing,
Southwest China. '
        db 'It is also one of the "211 Project" and "985 Project" universities
with full support in construction'
        db 'and development from the central government and Chongqing
Municipal Government. '
    len equ $- str-1
data ends

code segment
assume cs:code;ds:data
start:
mov cx,len
lea si,str
search:
push cx
mov cl,byte ptr [si]
cmp cl,'c'
jne next
inc si
mov cl,byte ptr [si]
cmp cl,'q'
jne next
inc si
mov cl,byte ptr [si]
cmp cl,'u'
je found

next:
pop cx;
inc si
dec cx
cmp cx,0
jnz search

found:
sub si,2
mov bx,si

code ends
end start

```

```
077F:0017 46      INC     SI
077F:0018 8A0C     MOV     CL,[SI]
077F:001A 80F975     CMP     CL,75
077F:001D 7408     JZ      0027
077F:001F 59      POP     CX
-u
077F:0020 46      INC     SI
077F:0021 49      DEC     CX
077F:0022 83F900     CMP     CX,+00
077F:0025 75E0     JNZ     0007
077F:0027 83EE02     SUB     SI,+02
077F:002A 8BDE     MOV     BX,SI
077F:002C C72BC504   MOV     WORD PTR [BP+DI],04C5
077F:0030 8288867EFF OR      BYTE PTR [BX+SI+7E86],FF
077F:0035 803D3A     CMP     BYTE PTR [DI],3A
077F:0038 741B     JZ      0055
077F:003A B85C00     MOV     AX,005C
077F:003D 50      PUSH    AX
077F:003E 8D867EFF   LEA     AX,[BP+FF7E]
-g 8bde
AX=0404 BX=0145 CX=0000 DX=0000 SP=FFFA BP=0000 SI=0147 DI=0001
DS=075A ES=075A SS=0769 CS=077F IP=FE37  NU UP EI PL NZ NA PE NC
077F:FE37 005300     ADD     [BP+DI+001],DL  SS:
```

在定义 str 的时候注意长度限制，需要多次定义以便占用连续的内存

