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 **DS1501 机房**

学 院 大数据与软件学院 年级 专业班 软件工程 班

学 生 姓 名 学 号

开 课 时 间 **2024** 至 **2025** 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： DS1501 机房　　2024 年 10 月 7 日

学院	大数据与软件学院	年级、专业	22 级软件工程	姓名		成绩	
课程名称	汇编语言程序设计	实验项目名　称	实验二：指令系统相关程序的编写与调试 **（涉及章节：第三章）**	指导教师		陈蜀宇	
教师评语	教师签名：陈蜀宇 2024 年 10 月 日						

一【实验目的】

- 1．掌握简单的汇编语言程序编写与调试,熟悉常见指令。

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 ；
3. 在输入框状态下 Z:\>mount C D:\masm ===>“Z:\>”这个是提示符 “C”作为虚拟 C 盘 "D:\masm"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

三【实验要求】

- 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 及重复前辍 REP 的内容、串操作、逻辑指令、控制转移指令等；
- 用 DEBUG 的有关命令调试本实验中的求累加和程序、多字节加法程序、数据串搬家程序段。

四 【实验内容】

（1）将下面的程序保存为 lab2.asm 文件，将其生成可执行文件 lab2.exe .

```
code segment
    Assume cs:code
    Start: Mov ax,2000h
```

```
        Mov ss,ax
        Mov sp,0
        Add sp,4
        Pop ax
        Pop bx
        Push ax
        Push bx
        Pop ax
        Pop bx
        Mov ah,4ch
        Int 21h
Code ends
End start
```

(2) 用 debug 跟踪 lab2.exe 的执行过程，写出每一步执行后，相关寄存器中的内容和栈顶的内容。

(3) 例题： 3.50 –教材 63 页。

(4) 例题： 3.56 -教材 67 页。

(5) 习题： 3.37 -教材 114 页。

五【实验步骤】

1.

```
code segment
 assume cs:code
 start:  mov ax,2000h
         mov ss,ax
         mov sp,0
         add sp,4
         pop ax
         pop bx
         push ax
         push bx
         pop ax
         pop bx
         mov ah,4ch
         int  21h
code ends
end start
```

```
Z:\>MOUNT C "E:\dosbox\DOSBox-0.74-3"
Drive C is mounted as local directory E:\dosbox\DOSBox-0.74-3\

Z:\>C:

C:\>MASM.EXE
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Source filename [.ASM]: lab2.asm
Object filename [lab2.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:

  51760 + 464784 Bytes symbol space free

       0 Warning Errors
       0 Severe  Errors

C:\>
```

```
C:\>MASM.EXE
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Source filename [.ASM]: lab2.asm
Object filename [lab2.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:

  51760 + 464784 Bytes symbol space free

       0 Warning Errors
       0 Severe  Errors

C:\>link lab2

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Run File [LAB2.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

C:\>_
```

2.

```
C:\>lab2.exe

C:\>debug lab2.exe
-
```

```
AX=01A2  BX=7202  CX=0015  DX=0000  SP=0006  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=000E   NV UP EI PL NZ NA PO NC
076A:000E 53            PUSH    BX
-t

AX=01A2  BX=7202  CX=0015  DX=0000  SP=0004  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=000F   NV UP EI PL NZ NA PO NC
076A:000F 58            POP     AX
-t

AX=7202  BX=7202  CX=0015  DX=0000  SP=0006  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0010   NV UP EI PL NZ NA PO NC
076A:0010 5B            POP     BX
-t

AX=7202  BX=01A2  CX=0015  DX=0000  SP=0008  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0011   NV UP EI PL NZ NA PO NC
076A:0011 B44C          MOV     AH,4C
-t

AX=4C02  BX=01A2  CX=0015  DX=0000  SP=0008  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0013   NV UP EI PL NZ NA PO NC
076A:0013 CD21          INT     21
-_
```

```
AX=4C02  BX=01A2  CX=0015  DX=0000  SP=0008  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0013   NV UP EI PL NZ NA PO NC
076A:0013 CD21          INT     21
-t

AX=4C02  BX=01A2  CX=0015  DX=0000  SP=0002  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=F000  IP=14A0   NV UP DI PL NZ NA PO NC
F000:14A0 FB            STI
-t

AX=4C02  BX=01A2  CX=0015  DX=0000  SP=0002  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=F000  IP=14A1   NV UP EI PL NZ NA PO NC
F000:14A1 FE38          ???     [BX+SI]                         DS:01A2=7A
-t
```

```
AX=2000  BX=0000  CX=0015  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076A  IP=0003   NV UP EI PL NZ NA PO NC
076A:0003 8ED0          MOV     SS,AX
-t

AX=2000  BX=0000  CX=0015  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0008   NV UP EI PL NZ NA PO NC
076A:0008 83C404        ADD     SP,+04
-t

AX=2000  BX=0000  CX=0015  DX=0000  SP=0004  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=000B   NV UP EI PL NZ NA PO NC
076A:000B 58            POP     AX
-t

AX=01A2  BX=0000  CX=0015  DX=0000  SP=0006  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=000C   NV UP EI PL NZ NA PO NC
076A:000C 5B            POP     BX
-t

AX=01A2  BX=7202  CX=0015  DX=0000  SP=0008  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=000D   NV UP EI PL NZ NA PO NC
076A:000D 50            PUSH    AX
-
```

栈顶内容变化：

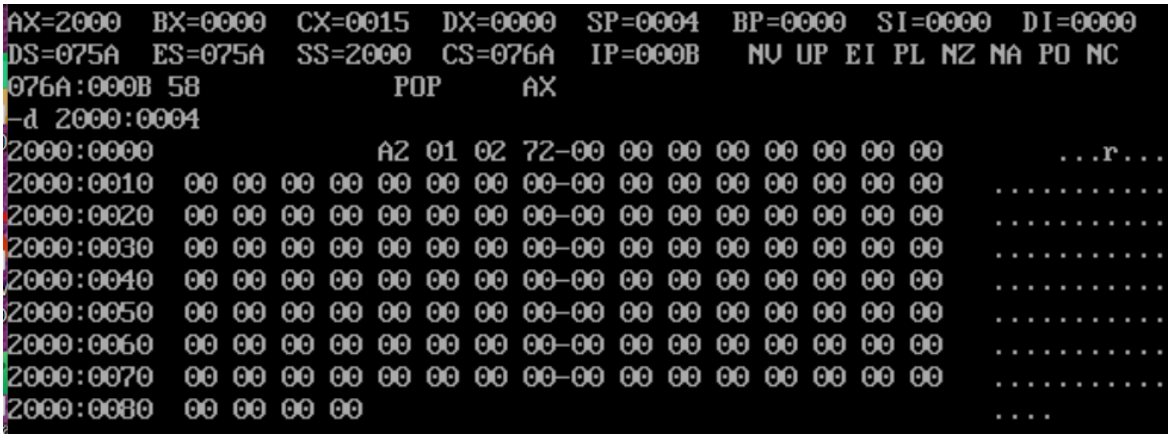
在 mov ss 寄存器之前，栈内没有变化

```
AX=2000  BX=0000  CX=0015  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076A  IP=0003   NV UP EI PL NZ NA PO NC
076A:0003 8ED0          MOV     SS,AX
-d 0769:0000
0769:0000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 0D   ................
0769:0010  B8 00 20 8E D0 BC 00 00-83 C4 04 58 5B 50 53 58   .. ........X[PSX
0769:0020  5B B4 4C CD 21 46 FC 8B-56 FE 05 0C 00 52 50 E8   [.L.!F..V....RP.
0769:0030  0E 49 83 C4 04 50 E8 9F-0E 83 C4 04 3D FF FF 74   .I...P......=..t
0769:0040  03 E9 11 01 B8 2F 00 50-8B 46 FC 8B 56 FE 05 0C   ...../.P.F..V...
0769:0050  00 52 50 E8 EA 48 83 C4-04 50 E8 7B 0E 83 C4 04   .RP..H...P.{....
0769:0060  3D FF FF 74 03 E9 ED 00-C4 5E FC 26 8A 47 0C 2A   =..t.....^.&.G.*
0769:0070  E4 40 50 8B C3 8C C2 05-0C 00 52 50 E8 C1 48 83   .@P.......RP..H.
```

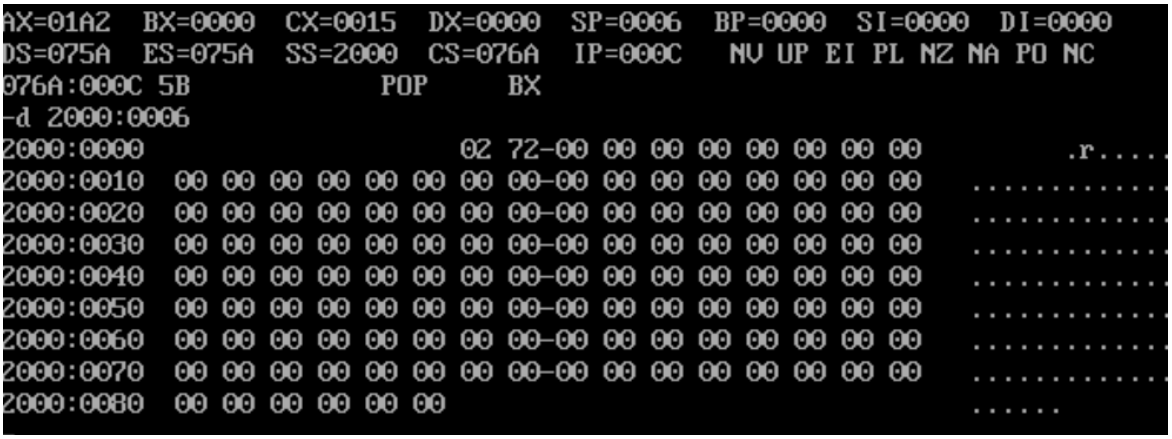
将 ax mov 到 ss 后，栈顶为 6A07

```
AX=2000  BX=0000  CX=0015  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0008   NV UP EI PL NZ NA PO NC
076A:0008 83C404        ADD     SP,+04
-d 2000:0000
2000:0000  6A 07 A3 01 A2 01 02 72-00 00 00 00 00 00 00 00   j......r........
2000:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
2000:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

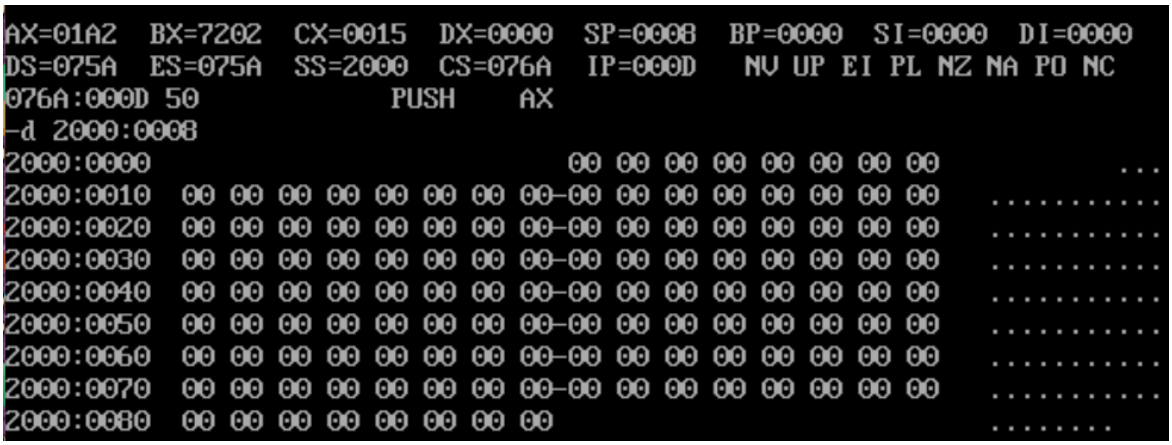
移动 sp 指针指向第四个字节



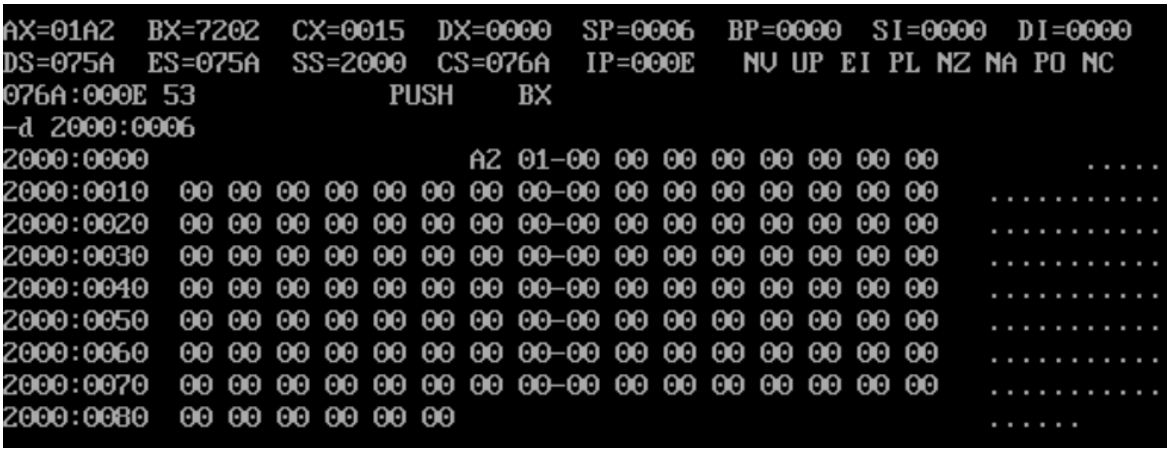
然后将 A201 弹出到 AX 中



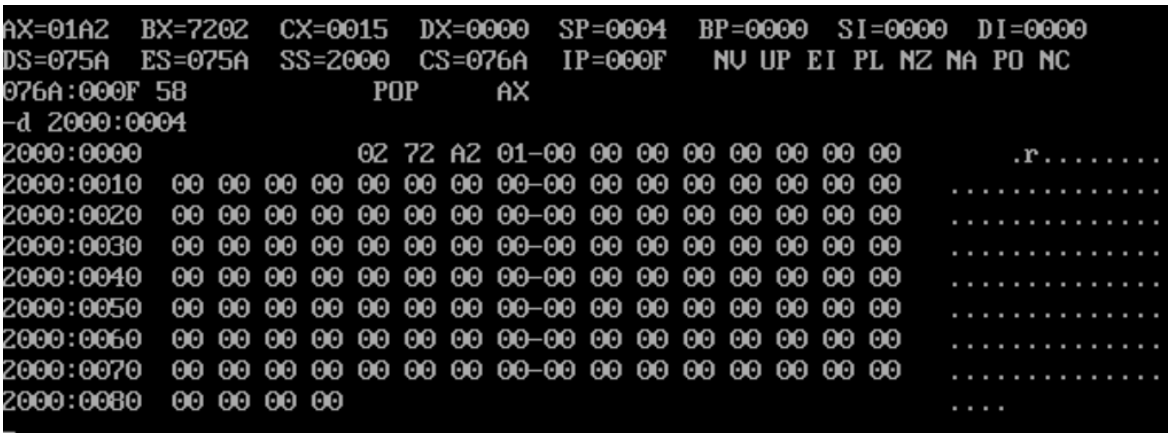
将 0272 弹出到 BX 中，栈顶为空



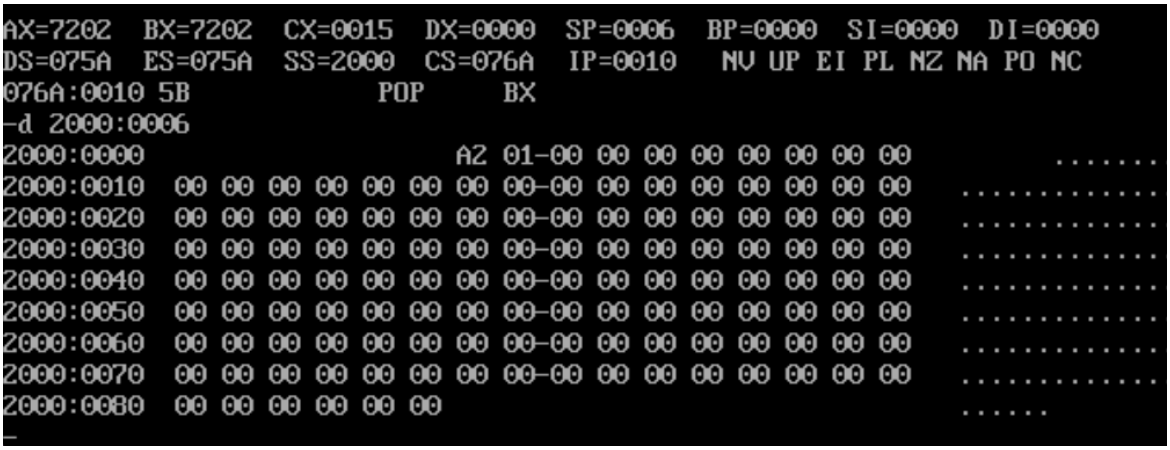
再次将 AX 压入栈内，栈顶为 A201



再次将 BX 压入栈内，栈顶为 0272



然后再依次弹出 0272，A201 给 ax，bx，然后栈内再次为空



```
AX=7202  BX=01A2  CX=0015  DX=0000  SP=0008  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=2000  CS=076A  IP=0011   NV UP EI PL NZ NA PO NC
076A:0011 B44C          MOV     AH,4C
-d 2000:0008
2000:0000                               00 00 00 00 00 00 00 00          ....
2000:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ............
2000:0080  00 00 00 00 00 00 00 00                           ........
```

（3）、（4）逻辑上没有问题，但是作为框架，并没有具体给 X,Y,Z...赋值

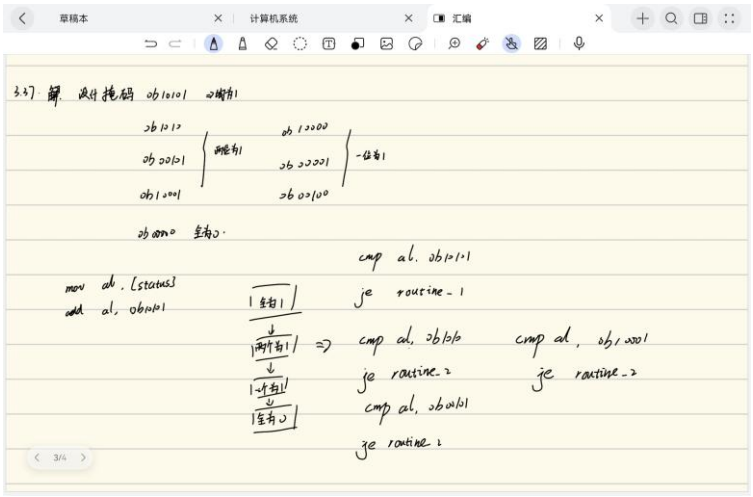
```
文件  编辑  查看

code segment
  assume cs:code
start:  mov  ax,x
        mov  dx,x+2
        add  ax,y
        adc  dx,y+2
        add  ax,24
        adc  dx,0
        sub  ax,z
        sbb  dx,z+2
        mov  w,ax
        mov  w+2,dx
code ends
end start
```

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Program:   DOSBOX
Unable to open input file: lab2.1

C:\>masm.exe
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Source filename [.ASM]: lab20
Object filename [lab20.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:
lab20.ASM(3): error A2009: Symbol not defined: X
lab20.ASM(4): error A2009: Symbol not defined: X
lab20.ASM(5): error A2009: Symbol not defined: Y
lab20.ASM(6): error A2009: Symbol not defined: Y
lab20.ASM(9): error A2009: Symbol not defined: Z
lab20.ASM(10): error A2009: Symbol not defined: Z
lab20.ASM(11): error A2009: Symbol not defined: W
lab20.ASM(12): error A2009: Symbol not defined: W

  51684 + 464860 Bytes symbol space free

      0 Warning Errors
      8 Severe  Errors
```

```
lab21.asm
文件  编辑  查看

  assume cs:code
  start:  mov   ax,x
          imul  y
          mov   cx,ax
          mov   bx,dx
          mov   ax,z
          cwd
          add   cx,ax
          adc   bx,dx
          sub   cx,540
          sbb   bx,0
          mov   ax,v
          cwd
          sub   ax,cx
          sbb   dx,bx
          idiv  x
  code ends
  end start
```

```
Drive C is mounted as local directory E:\dosbox\DOSBox-0.74-3\

Z:\>C:

C:\>MASM.EXE
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.  All rights reserved.

Source filename [.ASM]: lab21
Object filename [lab21.OBJ]:
Source listing  [NUL.LST]:
Cross-reference [NUL.CRF]:
lab21.ASM(3): error A2009: Symbol not defined: X
lab21.ASM(4): error A2009: Symbol not defined: Y
lab21.ASM(7): error A2009: Symbol not defined: Z
lab21.ASM(13): error A2009: Symbol not defined: V
lab21.ASM(17): error A2009: Symbol not defined: X

  51760 + 464784 Bytes symbol space free

      0 Warning Errors
      5 Severe  Errors
```

（5）



```
code segment
   assume cs:code
   start:      mov     al,[status]
                       and      al,0b10101
                       cmp     al,0b10101
                       je   routine_1
                       cmp     al,0b10100
                       je   routine_2
                       cmp     al,0b10001
                       je   routine_2
                       cmp     al,0b00101
                       je   routine_2
                       cmp     al,0b00001
                       je   routine_3
                       cmp     al,0b00100
                       je   routine_3
                       cmp     al,0b10000
                       je   routine_3
                       cmp      al,0b00000
                       je routine_4
code ends
start end
```

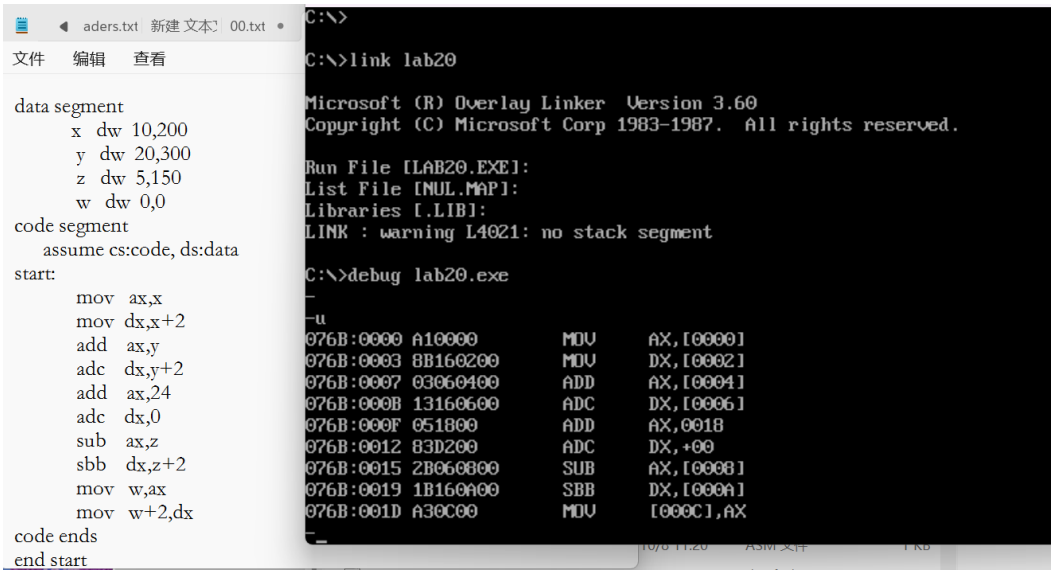
六【实验结果及分析】

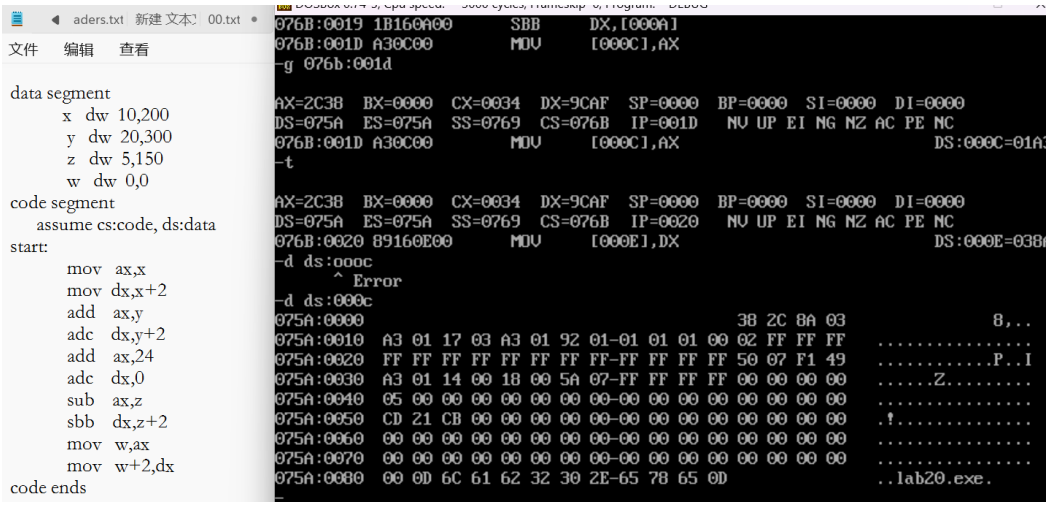
(1) 先用记事本编写程序并放入到挂载的文件夹中，然后再 masm 和 link，注意 link 时不要加后缀

(2) 出栈入栈基本流程，第五部分有跟步骤分析

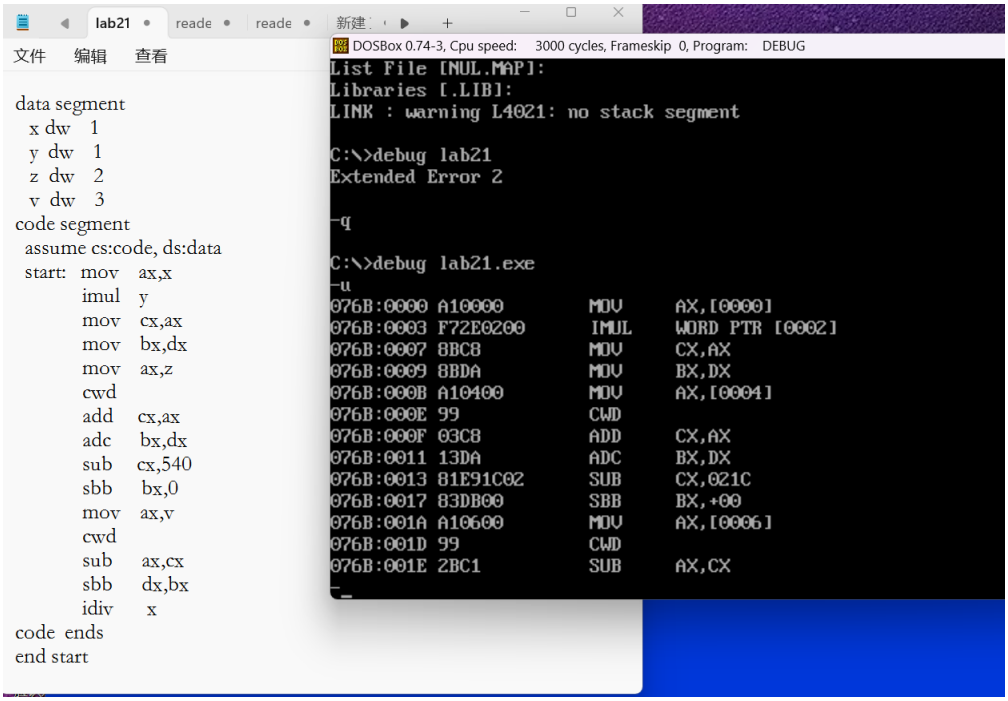
(3)、(4) 四则运算基本实现方式（注意取值时对于 x,y,z 都应该取两个数分别对应高地址和低地址）

此时举例将 x,y,z 具体化，利用 u 命令后查看到 w 的偏移地址是 000c，加上 cs 寄存器，可以查看运算后的数据

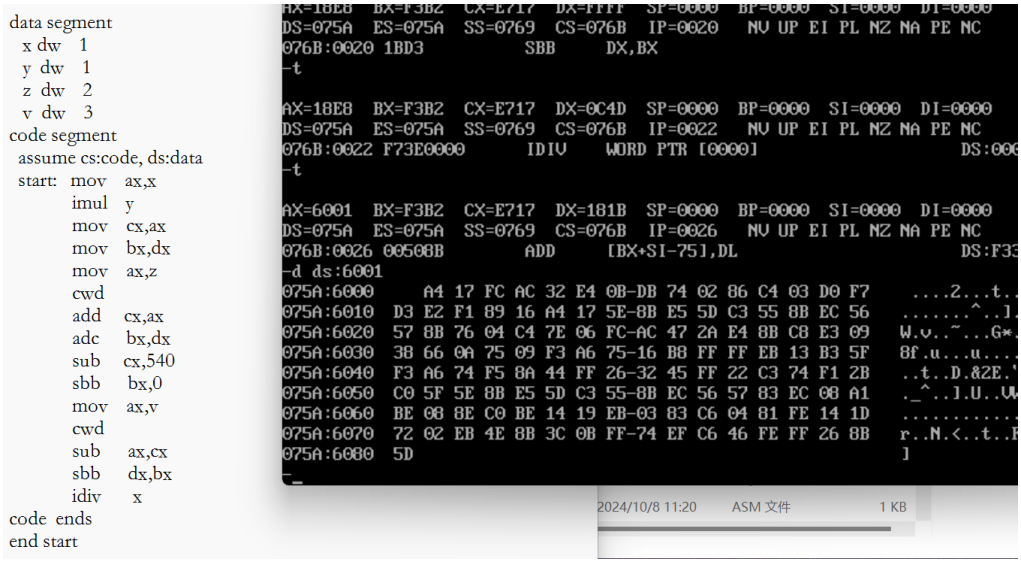




在定义变量的时候也可以在数据段，但是由于版本问题，不支持 x word 0这样的写法，定义十六位时需要用dw才行,注意赋值的时候不要把x赋值位0；而且在加载z完毕后将ax累加在cx中时可能产生溢出，导致程序异常



执行完毕后，可以查看ax，dx的内容，t命令后会直接返回



（5）在计算1的具体个数时没有找到同组的规律，但是每一个条件单独匹配跳转效率又很低下