X86 汇编语言程序设计

实验报告

姓名：王子卓

学号：71115115

东南大学计算机科学与工程学院、软件学院

School of Computer Science & Engineering

College of Software Engineering

Southeast University

贰零壹柒年伍月

实验一　汇编语言程序上机过程

一）实验目的

- 学会安装“16位汇编程序开发软件”的安装，完成将汇编语言源程序录入进计算机、利用ml.exe进行汇编，LINK进行链接，并用DEBUG调试16位程序的全部过程。本实验大家不必了解程序细节，只是为了熟悉开发环境和上机过程。

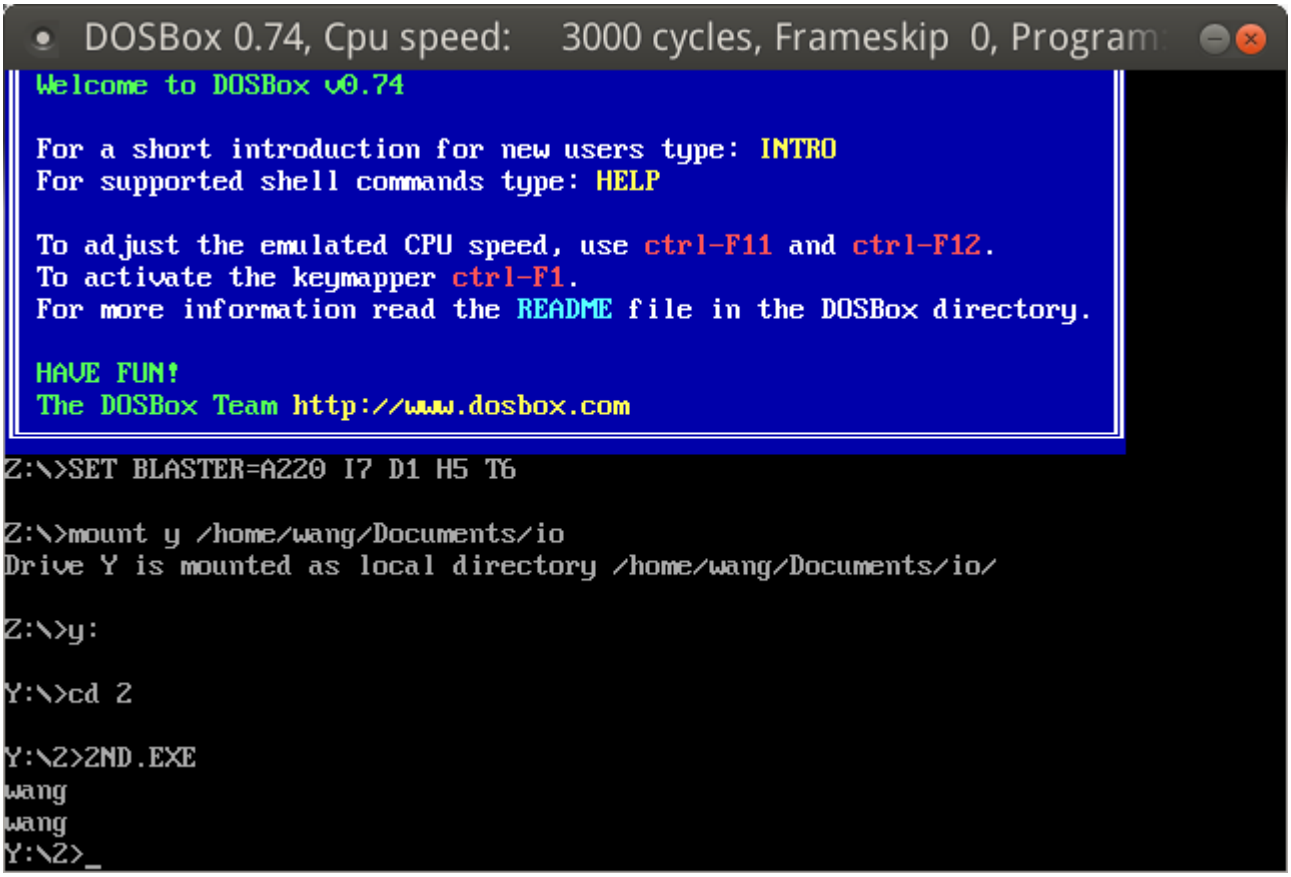
二）实验内容

> 写16位汇编程序，从键盘输入一个字符串，然后换行后将该字符串输出到屏幕。

1. 源程序

```
.8086
.MODEL SMALL
.DATA
ORG 100H;CS起始地址
STR DB 20,0,20 DUP(20H);字符串STR
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX;数据段地址
    LEA DX,STR;将STR的相对于数据段首地址的偏移地址放进DX
    MOV AH,0AH
    INT 21H;从键盘输入一个字符串

    MOV DL,0AH
    MOV AH,02
    INT 21H;输出DL中的换行
    MOV DL,0DH
    MOV AH,02
    INT 21H;输出回车

    MOV BL,STR[1]
    MOV BH,0;BX=20
    MOV BYTE PTR STR[BX+2],'$';末尾加'$'
    LEA DX,STR+2
    MOV AH,9
    INT 21H;输出这一句话
    MOV AH,4CH
    INT 21H;返回DOS
END START
```

2. 运行结果贴图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> `INT 21`的9号调用输出以DS:DX为首地址，"$"为结尾的字符串到显示器中。若要在显示字符串光标自动回车换行，则在"$"字符前面加上0DH(回车),0AH(换行)字符。

实验二　顺序程序设计

一）实验目的

> 通过这一部分的实验，进一步熟悉汇编过程和DEBUG调试过程；掌握用汇编语言编写顺序程序。

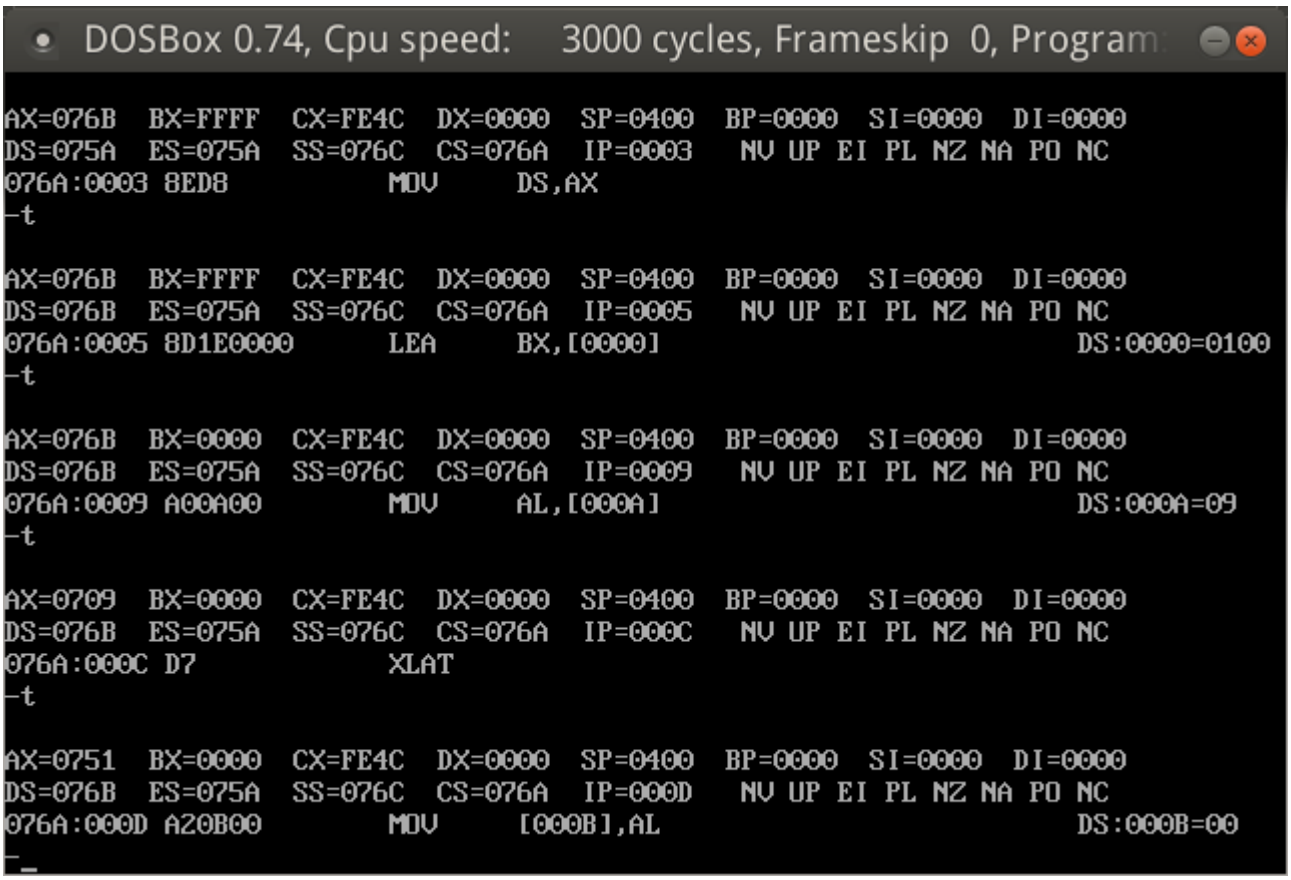
二）实验内容

> 写完整程序16位程序，在内存中从Table开始的10个单元中连续存放0到9的平方值，任给一个0到9的数X，该数存放在内存单元XX中，用XLAT指令查表求X的平方值，并将结果存于内存YY单元中。编写程序，并在DEBUG中进行调试和验证结果。(X，XX，YY都是内存中的变量）

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    Table BYTE 0,1,4,9,16,25,36,49,64,81
    XX BYTE 9
    YY BYTE ?
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX
    LEA BX,Table
    MOV AL,XX
    XLAT;以DS:[BX+AL]为地址，提取存储器中的一个字节再送入AL。
    MOV YY,AL
END START
```

2. 运行结果贴图



可以看到DS:000A放的是09H，查找后AL中放的是51H=81D，存入YY=DS:000B中。

3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> XLAT的指令功能：把待查表格的一个字节内容送到AL累加器中。在执行该指令前，应将Table先送至BX寄存器中，然后将待查字节与其在表格中距表首地址位移量送AL,即AL<--((BX)+(AL)). 执行XLAT将使待查内容送到累加器。

实验三　分支程序设计

一）实验目的

> 通过本实验，熟练运算类指令对标志位状态的影响，以及标志位状态的表示方法；掌握无条件转移、条件转移指令的使用方法；掌握分支程序设计和调试方法。

二）实验内容

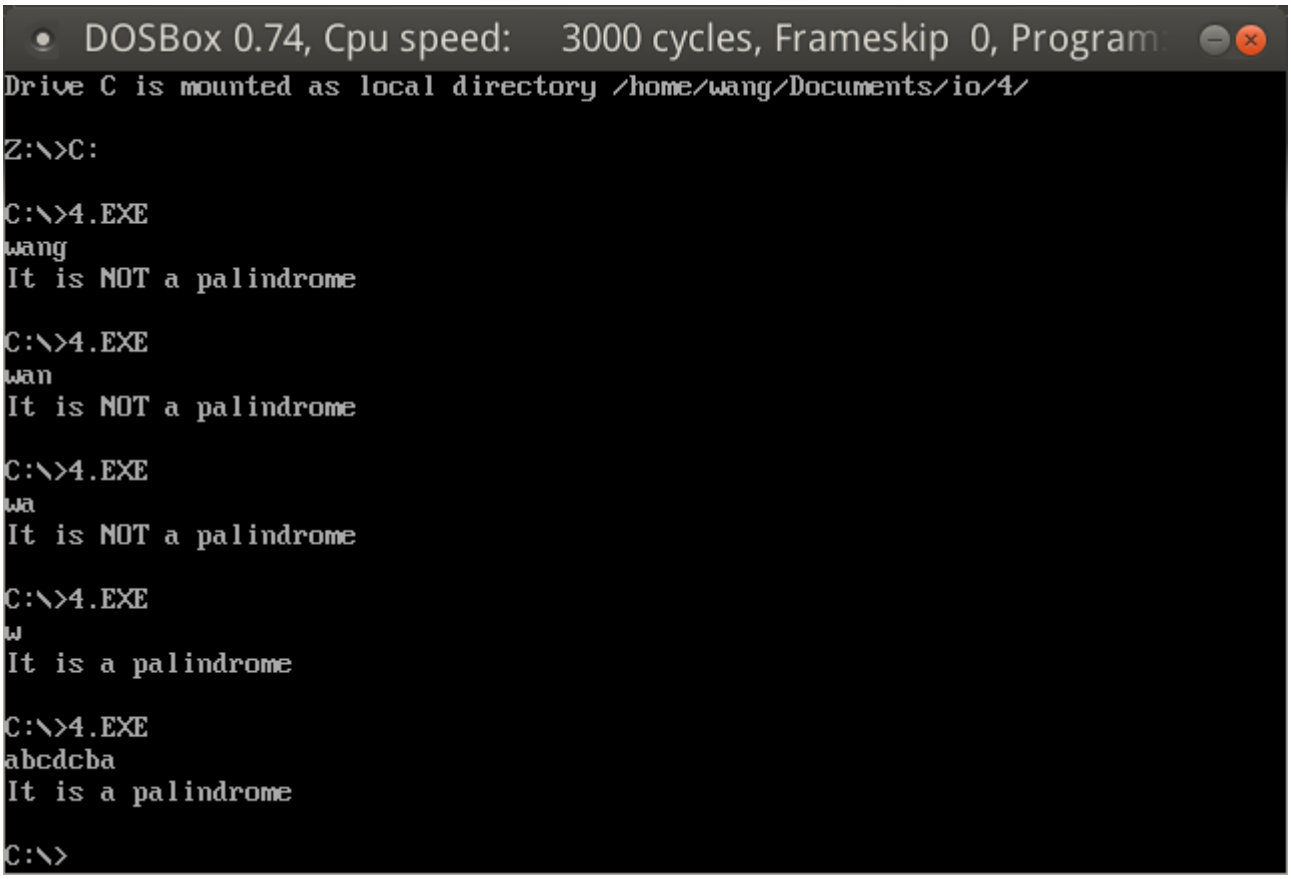
> 所谓回文字符串是指一个字符串正读和倒读都是一样的，例如字符串‘ABCDEFFEDCBA’就是一个回文字符串，而字符串‘ABCFDDCAB’就不是回文字符串。现在编写完整的16位汇编程序，输入一个字符串，判断该字符串是否为回文字符串，并用“It is a palindrome”或“It is NOT a palindrome”作为输出。

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    N EQU 20
    MAXLEN BYTE N
    ACTLEN BYTE ?
    STRING BYTE N DUP('$')
    TRUE BYTE 0AH,0DH,'It is a palindrome',0AH,0DH,'$'   ;开头加0AH、0DH是为了防止回车把输入冲掉
    FALSE BYTE 0AH,0DH,'It is NOT a palindrome',0AH,0DH,'$'
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX
    MOV AH,0AH
    MOV DX,OFFSET MAXLEN
    INT 21H;缓冲区的第一个字节指定容纳的最大字符个数，由用户给出；第二个字节存放实际的最大字符个数，由系统最后填入；从第三个字节开始存放从键盘接受的字符，直到ENTER键结束。
    MOV AL,ACTLEN
    MOV AH,0
    MOV BL,2
    DIV BL
    CMP AH,1
    JE ITSODD;如果是奇数字数的字符串，跳到"It's odd"
    CMP AL,1
    JE ITSTWO;如果是两个字符的字符串，跳到"It's two"
    ADD AL,AH;奇数个数和偶数个数的区别就在这句话
    MOV AH,0
    MOV BL,ACTLEN
    MOV BH,0
    MOV SI,BX
    MOV BX,OFFSET STRING
    DEC SI
ITSEVEN:
    CMP SI,AX
    JNA T
    MOV DL,[BX+SI]
    MOV DH,[BX]
    CMP DL,DH
    JNZ F
    INC BX
    SUB SI,2;因为BX加了一，所以BX+SI要想减一，就需要SI减二
    JMP ITSEVEN
ITSODD:
    MOV AH,0
    MOV BL,ACTLEN
    MOV BH,0
    MOV SI,BX
    MOV BX,OFFSET STRING
    DEC SI
JUDGEODD:
    CMP SI,AX
```

```
    JNA T
    MOV DL,[BX+SI]
    MOV DH,[BX]
    CMP DL,DH
    JNZ F
    INC BX
    SUB SI,2
    JMP JUDGEODD
ITSTWO:
    MOV BX,OFFSET STRING
    MOV DL,[BX+1]
    MOV DH,[BX]
    CMP DL,DH
    JNE F;如果这两个字符不相同，就不是。否则直接向下执行T
T:  MOV DX,OFFSET TRUE
    JMP QUIT
F:  MOV DX,OFFSET FALSE
    JMP QUIT
QUIT:
    MOV AH,09H
    INT 21H
    MOV AH,4CH
    INT 21H
END START
```

2. 运行结果贴图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 1. TRUE和FALSE字符串开头加0AH、0DH可以防止回车把输入冲掉
> 2. 奇偶情况不同，并且两个字符的情况与其他的偶数情况不同

实验四　循环程序设计

一）实验目的

> 通过实验，可以掌握循环结构的各种实现方法，进一步了解循环结构中初始化部分、循环体部分、循环控制部分的功能以及他们彼此之间的关系。尤其是多重循环中外层循环和内层循环之间的关系。

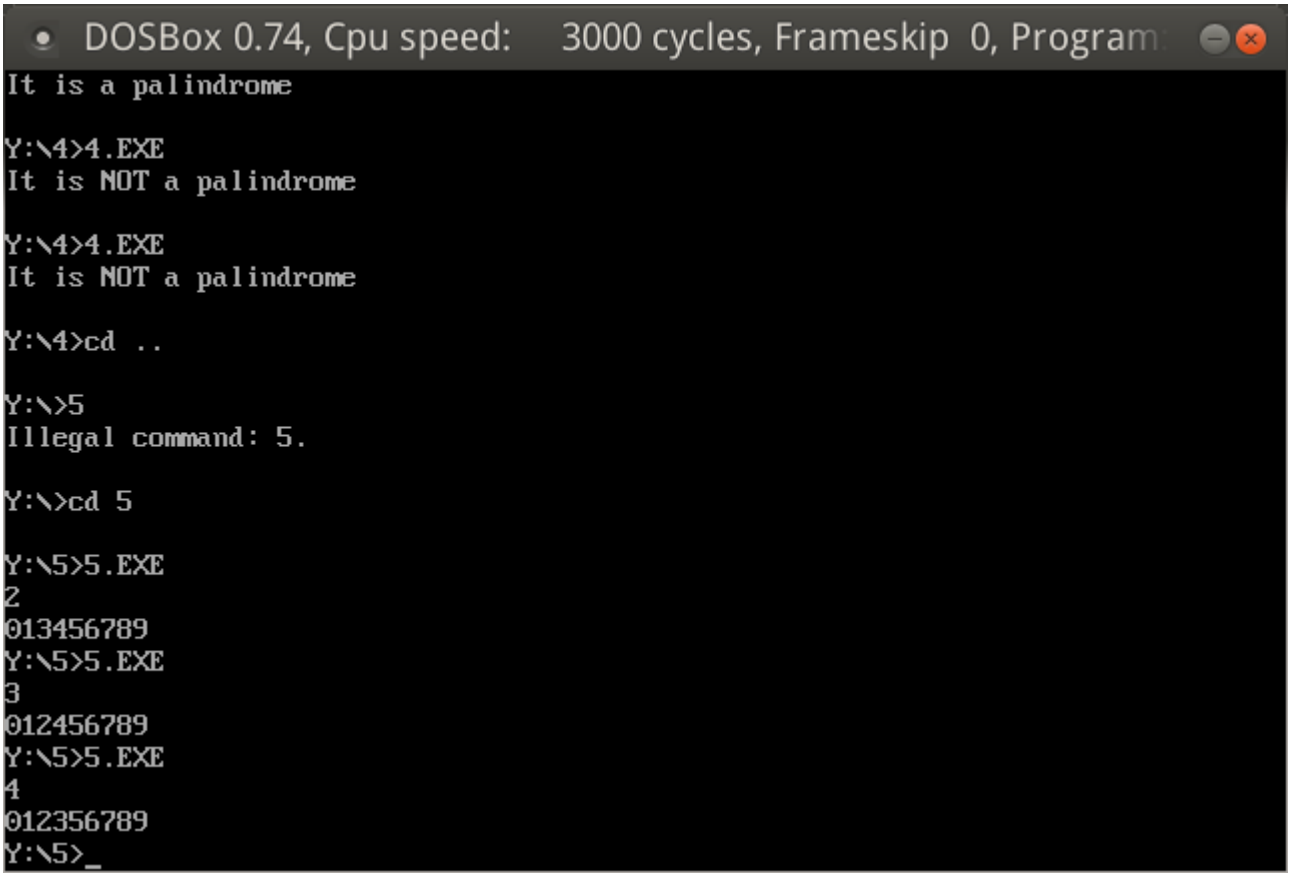
二）实验内容

> 请编写16位完整汇编程序，在一个升序字节数组BUFF中查找数N，找到后将此数从数组中删除，并使得CF=0；没找到返回CF＝1。

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    BUFF BYTE 0,1,2,3,4,5,6,7,8,9
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX
    MOV CX,10;计数初始化为10
    LEA SI,BUFF;BUFF首地址给SI
    MOV AH,01H
    INT 21H;从键盘输入一个字符
    SUB AL,30H;将ASCII码变成数字
    MOV AH,0
NEXT:
    MOV BL,[SI]
    MOV BH,0
    CMP AX,BX
    JE DEL;找到了就跳至删除
    INC SI
    LOOP NEXT
    JMP STOP;找不到就停
DEL:
    MOV DI,SI;记住当前元素的位置
    INC SI
    MOV AL,[SI]
    MOV AH,0
    MOV [DI],AX;后面的内容前移
    LOOP DEL
    MOV [SI],0;找到的最后一位换成0
    LEA SI,BUFF;SI指向BUFF首地址
    MOV CX,9;设计数器为9
    MOV AH,02H
    MOV DL,0DH
    INT 21H;回车
    MOV DL,0AH
    INT 21H;换行
DO:
    MOV DL,[SI]
    MOV DH,0
    ADD DX,3030H;十六进制转ASCII
    INC SI
    INT 21H;一个个输出此时BUFF内容
    LOOP DO
STOP:
    MOV AH,4CH
    INT 21H;退回DOS
END START
```

2. 运行结果截图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 八位寄存器中的两位十六进制数可以加减30H变成想要的ASCII码或者数字，十六位寄存器中的四位十六进制数可以加减3030H变成想要的ASCII码或者数字。

实验五　子程序设计

一）实验目的

> 通过本实验，掌握子程序的定义和调用方法。通过程序调试，进一步理解CALL指令和RET指令的功能，掌握子程序调用时参数传递的方法。

二）实验内容

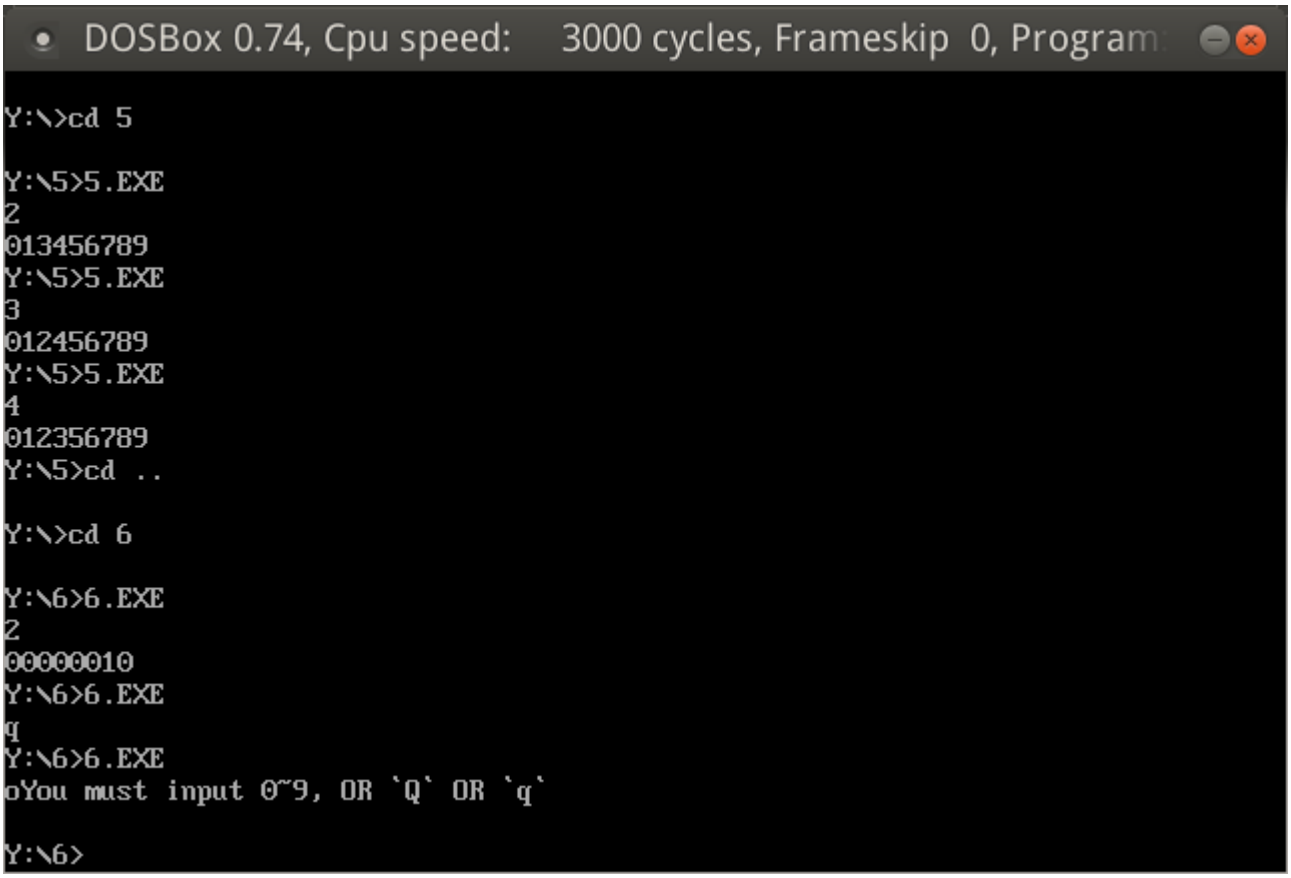
> 1. 请编写完整16位汇编程序从键盘读取字符，如果是十进制的‘0’～‘9’则在屏幕上输出该数的8位二进制码，并将数字依次存放到BUF开头的数组中，如果读入的字符是‘Q’或者‘q’，则程序退出，其他情况在屏幕上打印“You must input 0~9, or ‘q’ or ‘Q’”。（如输入的字符是‘9’，则输出“00001001”）.提示：输出一个数的2进制形式应该从最高位开始输出。要求打印一个数的2进制形式和输出回车换行分别定义成一个子程序可以将此段程序定义成一个过程。

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    STRING DB 'You must input 0~9, OR `Q` OR `q`',0AH,0DH,'$'
    BUF DB 8 DUP(30H),'$'
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX
    MOV AH,01H
    INT 21H
    CMP AL,'q'
    JE EXIT
    CMP AL,'Q'
    JE EXIT;输入'q'或'Q'时退出
    CMP AL,'0'
    JL WRONG
    CMP AL,'9'
    JG WRONG;输入小于零或者大于九时错误

    MOV BL,AL
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV DL,0AH
    INT 21H;回车换行

    MOV AL,BL
    SUB AL,30H;ASCII码转数字
    MOV DX,0
    MOV AH,0
    MOV BX,OFFSET BUF+7
AGAIN:
    MOV DX,0;余数清零
    MOV CX,2
    DIV CX
    ADD DL,30H;转ASCII码
    MOV [BX],DL;将零或一附给BX所指内存中的元素
    DEC BX;向左移一位
    AND AX,AX
    JNE AGAIN;AX不为零就继续循环
    JMP RIGHT;其实这句可有可无，当时随便就写上了
RIGHT:
    MOV AH,09H
    LEA DX,BUF
    INT 21H;输出BUF内容
    JMP EXIT
WRONG:
    MOV AH,09H
    LEA DX,STRING
    INT 21H;输出STRING内容
    JMP EXIT
```

```
EXIT:
    MOV AH,4CH
    INT 21H;退出到DOS
END START
```

2. 运行结果截图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 刚开始没想取余数从十六进制转到二进制，后来发现我也不知道什么太好的办法-_-|||
>
> 通过写这个程序，我对DIV有了更深的认识。

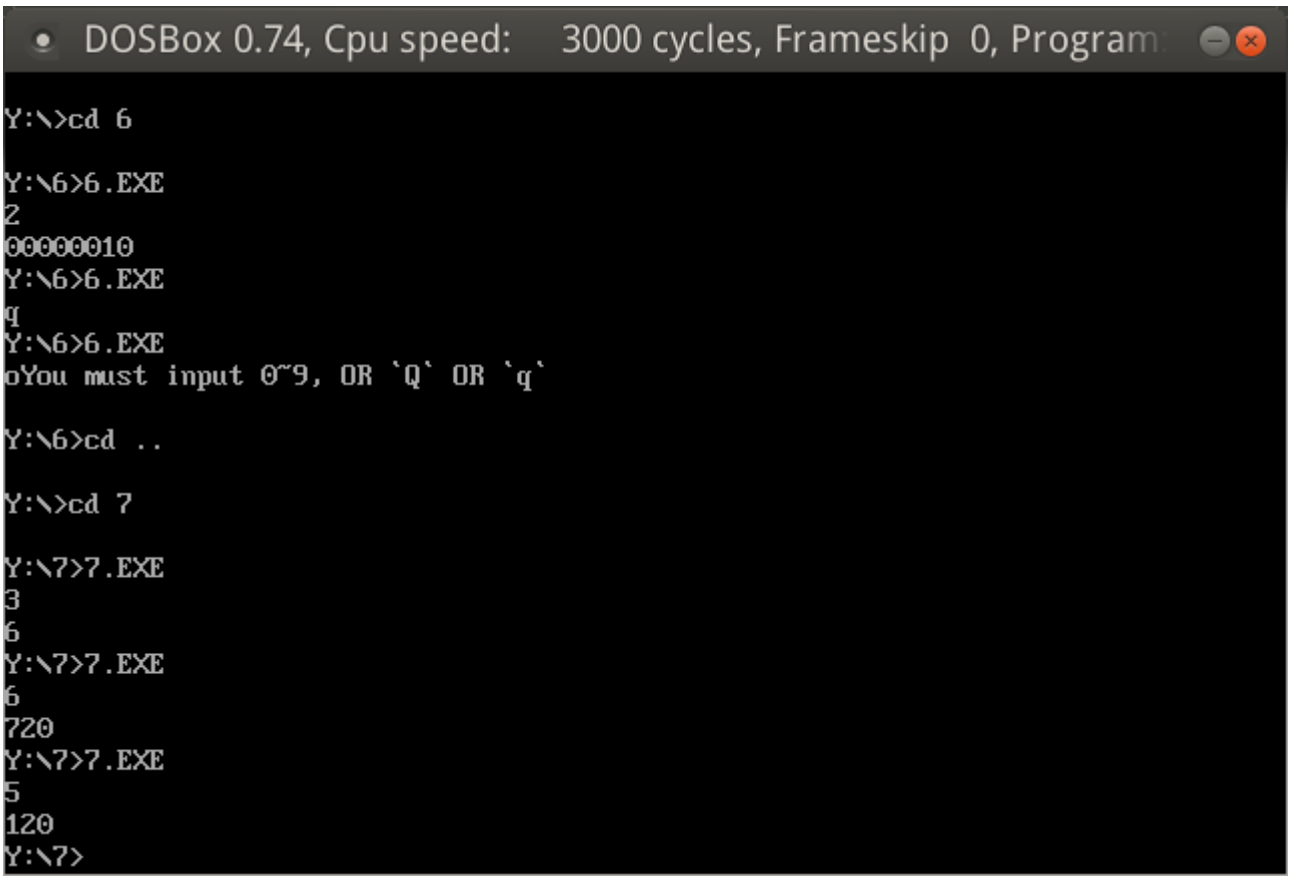
> 2. Programming a sub routine to calculate N!. Specific requirements：Read a number N(1~6) from keyboard，programming a sub routine named DAC to calcute N！, then print the result to screen in decimal form.

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    STRING BYTE 6 DUP(20H),'$'
    ENDL BYTE 0AH,0DH,'$'
    ERRMSG BYTE 'You must input a number from 1~6',0AH,0DH,'$'
.CODE
START:
    MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    MOV AH,01H
    INT 21H
    CMP AL,'1'
    JL ERR;小于1的ASCII码错误
    CMP AL,'6'
    JG ERR;大于6的ASCII码错误
    SUB AL,30H
    MOV CL,AL
    LEA DX,ENDL
    MOV AH,09H
    INT 21H
    MOV DX,0
    LEA BX,STRING+5
    MOV AL,01H
    MOV AH,0
    MOV CH,0
    CALL DAC
    CALL PRINT
    MAIN ENDP

    DAC PROC
AGAIN:
    MOV DX,CX
    MUL DX
    LOOP AGAIN
    DAC ENDP

    PRINT PROC
    MOV CX,10
DO:
    MOV DX,0
    DIV CX
    ADD DL,30H
    DEC BX
    MOV [BX],DL
    AND AX,AX
    JNE DO
    MOV DX,BX
    MOV AH,09H
    INT 21H
    MOV AH,4CH
```

```
    INT 21H
    PRINT ENDP

ERR:
    LEA DX,ENDL
    MOV AH,09H
    INT 21H
    LEA DX,ERRMSG
    MOV AH,09H
    INT 21H
    MOV AH,4CH
    INT 21H
END START
```

2. 运行结果截图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 通过写这个程序，我对子程序设计更加熟悉。

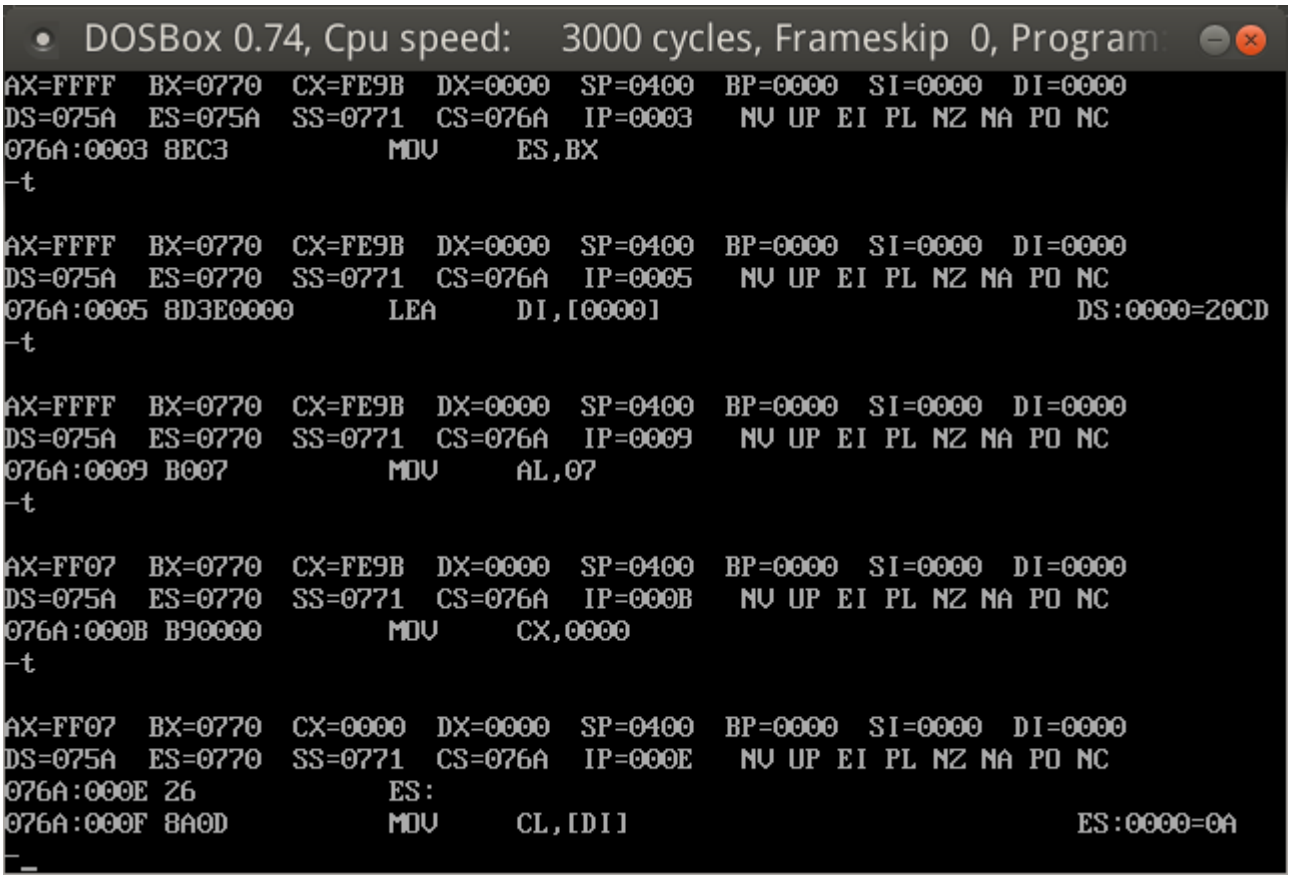
> 3. 在附加段中有一个从小到大排序的无符号数字数组，其首地址在DI中，数组的第一个单元存放数组长度。要求用折半查找法在数组中查找数N，假设该数已在AX中，如找到，CF=0，并在SI中给出该元素在数组中的偏移地址；如未找到，CF=1。

1. 源程序

```
.8086
.MODEL SMALL
EXTRA SEGMENT
    ARRAY BYTE 10,1,2,3,4,5,6,7,8,9,10
EXTRA ENDS
.CODE
ASSUME ES:EXTRA
START:
    MOV BX,EXTRA
    MOV ES,BX
    LEA DI,ES:ARRAY

    MOV AL,7

    MOV CX,0
    MOV CL,ES:[DI]
    INC DI
    MOV DX,DI
    ADD DX,CX
    MOV SI,DX
    MOV DX,0

    CMP AL,ES:[DI]
    MOV BX,DI
    JB NOTFOUND
    JE FOUND
    CMP AL,ES:[SI-1]
    MOV BX,SI
    JA NOTFOUND
    JE FOUND

WORK:
    MOV BX,DI
    ADD BX,SI
    SHR BX,1
    CMP AL,ES:[BX]
    JZ FOUND
    PUSHF
    CMP BX,DI
    JZ NOTFOUND
    POPF
    JL LESS
    MOV DI,BX
    JMP WORK
LESS:
    MOV SI,BX
    JMP WORK
NOTFOUND:
    STC
    JMP EXIT
FOUND:
    CLC
    MOV BX,SI
```

```
    JMP EXIT
EXIT:
    MOV AH,4CH
    INT 21H
END START
```

2. 运行结果截图



```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program
AX=FF07  BX=0770  CX=0000  DX=0000  SP=0400  BP=0000  SI=0000  DI=0000
DS=075A  ES=0770  SS=0771  CS=076A  IP=000E   NV UP EI PL NZ NA PO NC
076A:000E 26            ES:
076A:000F 8A0D          MOV     CL,[DI]                          ES:0000=0A
-t

AX=FF07  BX=0770  CX=000A  DX=0000  SP=0400  BP=0000  SI=0000  DI=0000
DS=075A  ES=0770  SS=0771  CS=076A  IP=0011   NV UP EI PL NZ NA PO NC
076A:0011 47            INC     DI
-t

AX=FF07  BX=0770  CX=000A  DX=0000  SP=0400  BP=0000  SI=0000  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0012   NV UP EI PL NZ NA PO NC
076A:0012 8BD7          MOV     DX,DI
-t

AX=FF07  BX=0770  CX=000A  DX=0001  SP=0400  BP=0000  SI=0000  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0014   NV UP EI PL NZ NA PO NC
076A:0014 03D1          ADD     DX,CX
-t

AX=FF07  BX=0770  CX=000A  DX=000B  SP=0400  BP=0000  SI=0000  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0016   NV UP EI PL NZ NA PO NC
076A:0016 8BF2          MOV     SI,DX
-
```

```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program
AX=FF07  BX=0770  CX=000A  DX=000B  SP=0400  BP=0000  SI=0000  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0016   NV UP EI PL NZ NA PO NC
076A:0016 8BF2          MOV     SI,DX
-t

AX=FF07  BX=0770  CX=000A  DX=000B  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0018   NV UP EI PL NZ NA PO NC
076A:0018 BA0000        MOV     DX,0000
-t

AX=FF07  BX=0770  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=001B   NV UP EI PL NZ NA PO NC
076A:001B 26            ES:
076A:001C 3A05          CMP     AL,[DI]                          ES:0001=01
-t

AX=FF07  BX=0770  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=001E   NV UP EI PL NZ NA PE NC
076A:001E 8BDF          MOV     BX,DI
-t

AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0020   NV UP EI PL NZ NA PE NC
076A:0020 7227          JB      0049
-_
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0020   NV UP EI PL NZ NA PE NC
076A:0020 7227          JB      0049
-t

AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0022   NV UP EI PL NZ NA PE NC
076A:0022 7428          JZ      004C
-t

AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0024   NV UP EI PL NZ NA PE NC
076A:0024 26            ES:
076A:0025 3A44FF        CMP     AL,[SI-01]                      ES:000A=0A
-t

AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0028   NV UP EI NG NZ AC PO CY
076A:0028 8BDE          MOV     BX,SI
-t

AX=FF07  BX=000B  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=002A   NV UP EI NG NZ AC PO CY
076A:002A 771D          JA      0049
-_
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=000B  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=002A   NV UP EI NG NZ AC PO CY
076A:002A 771D          JA      0049
-t

AX=FF07  BX=000B  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=002C   NV UP EI NG NZ AC PO CY
076A:002C 741E          JZ      004C
-t

AX=FF07  BX=000B  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=002E   NV UP EI NG NZ AC PO CY
076A:002E 8BDF          MOV     BX,DI
-t

AX=FF07  BX=0001  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0030   NV UP EI NG NZ AC PO CY
076A:0030 03DE          ADD     BX,SI
-t

AX=FF07  BX=000C  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ NA PE NC
076A:0032 D1EB          SHR     BX,1
-
```

```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program

AX=FF07  BX=000C  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ NA PE NC
076A:0032 D1EB          SHR     BX,1
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0034   NV UP EI PL NZ AC PE NC
076A:0034 26            ES:
076A:0035 3A07          CMP     AL,[BX]                          ES:0006=06
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0037   NV UP EI PL NZ NA PO NC
076A:0037 7413          JZ      004C
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0039   NV UP EI PL NZ NA PO NC
076A:0039 9C            PUSHF
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=003A   NV UP EI PL NZ NA PO NC
076A:003A 3BDF          CMP     BX,DI
-
```

```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program

AX=FF07  BX=0006  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=003A   NV UP EI PL NZ NA PO NC
076A:003A 3BDF          CMP     BX,DI
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=003C   NV UP EI PL NZ NA PE NC
076A:003C 740B          JZ      0049
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=003E   NV UP EI PL NZ NA PE NC
076A:003E 9D            POPF
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=003F   NV UP EI PL NZ NA PO NC
076A:003F 7C04          JL      0045
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0041   NV UP EI PL NZ NA PO NC
076A:0041 8BFB          MOV     DI,BX
-_
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0001
DS=075A  ES=0770  SS=0771  CS=076A  IP=0041   NV UP EI PL NZ NA PO NC
076A:0041 8BFB          MOV     DI,BX
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0043   NV UP EI PL NZ NA PO NC
076A:0043 EBE9          JMP     002E
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=002E   NV UP EI PL NZ NA PO NC
076A:002E 8BDF          MOV     BX,DI
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0030   NV UP EI PL NZ NA PO NC
076A:0030 03DE          ADD     BX,SI
-t

AX=FF07  BX=0011  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ AC PE NC
076A:0032 D1EB          SHR     BX,1
-_
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0011  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ AC PE NC
076A:0032 D1EB          SHR     BX,1
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0034   NV UP EI PL NZ AC PO CY
076A:0034 26            ES:
076A:0035 3A07          CMP     AL,[BX]                          ES:0008=08
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0037   NV UP EI NG NZ AC PE CY
076A:0037 7413          JZ      004C
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0039   NV UP EI NG NZ AC PE CY
076A:0039 9C            PUSHF
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=003A   NV UP EI NG NZ AC PE CY
076A:003A 3BDF          CMP     BX,DI
-
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0008  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=003A   NV UP EI NG NZ AC PE CY
076A:003A 3BDF          CMP     BX,DI
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=003C   NV UP EI PL NZ NA PO NC
076A:003C 740B          JZ      0049
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=03FE  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=003E   NV UP EI PL NZ NA PO NC
076A:003E 9D            POPF
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=003F   NV UP EI NG NZ AC PE CY
076A:003F 7C04          JL      0045
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0045   NV UP EI NG NZ AC PE CY
076A:0045 8BF3          MOV     SI,BX
-
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=000B  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0045   NV UP EI NG NZ AC PE CY
076A:0045 8BF3          MOV     SI,BX
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0047   NV UP EI NG NZ AC PE CY
076A:0047 EBE5          JMP     002E
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=002E   NV UP EI NG NZ AC PE CY
076A:002E 8BDF          MOV     BX,DI
-t

AX=FF07  BX=0006  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0030   NV UP EI NG NZ AC PE CY
076A:0030 03DE          ADD     BX,SI
-t

AX=FF07  BX=000E  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ NA PO NC
076A:0032 D1EB          SHR     BX,1
-
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

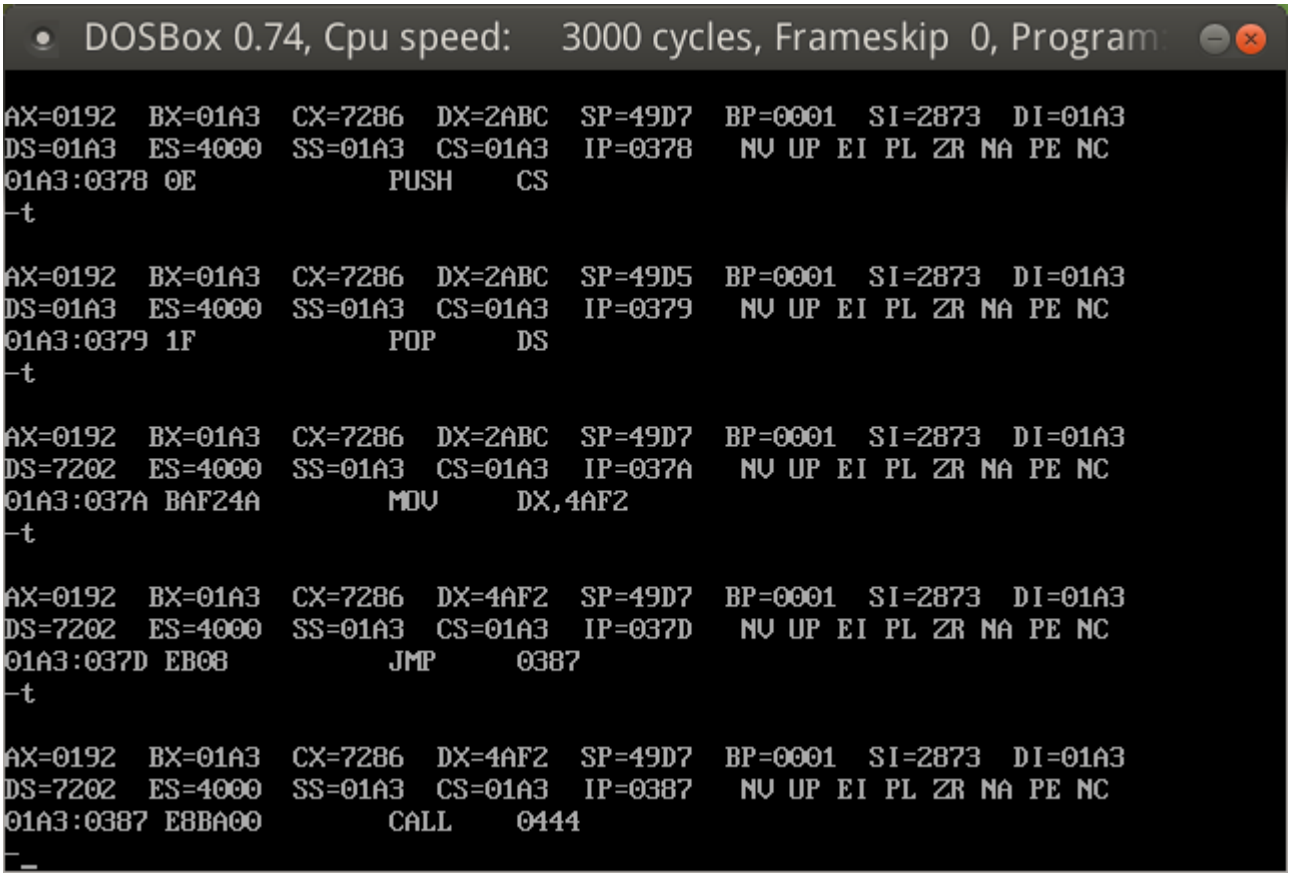
```
AX=FF07  BX=000E  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0032   NV UP EI PL NZ NA PO NC
076A:0032 D1EB          SHR     BX,1
-t

AX=FF07  BX=0007  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0034   NV UP EI PL NZ AC PO NC
076A:0034 26            ES:
076A:0035 3A07          CMP     AL,[BX]                         ES:0007=07
-t

AX=FF07  BX=0007  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0037   NV UP EI PL ZR NA PE NC
076A:0037 7413          JZ      004C
-t

AX=FF07  BX=0007  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=004C   NV UP EI PL ZR NA PE NC
076A:004C F8            CLC
-t

AX=FF07  BX=0007  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=004D   NV UP EI PL ZR NA PE NC
076A:004D 8BDE          MOV     BX,SI
-
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program

```
AX=FF07  BX=0007  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=004D   NV UP EI PL ZR NA PE NC
076A:004D 8BDE          MOV     BX,SI
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=004F   NV UP EI PL ZR NA PE NC
076A:004F EB00          JMP     0051
-t

AX=FF07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0051   NV UP EI PL ZR NA PE NC
076A:0051 B44C          MOV     AH,4C
-t

AX=4C07  BX=0008  CX=000A  DX=0000  SP=0400  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=076A  IP=0053   NV UP EI PL ZR NA PE NC
076A:0053 CD21          INT     21
-t

AX=4C07  BX=0008  CX=000A  DX=0000  SP=03FA  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=F000  IP=14A0   NV UP DI PL ZR NA PE NC
F000:14A0 FB            STI
-
```

```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program
DS=075A  ES=0770  SS=0771  CS=F000  IP=14A0   NV UP DI PL ZR NA PE NC
F000:14A0 FB            STI
-t

AX=4C07  BX=0008  CX=000A  DX=0000  SP=03FA  BP=0000  SI=0008  DI=0006
DS=075A  ES=0770  SS=0771  CS=F000  IP=14A1   NV UP EI PL ZR NA PE NC
F000:14A1 FE38          ???     [BX+SI]                           DS:0010=A3
-t

AX=01A3  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=034F   NV UP EI PL NZ NA PO NC
01A3:034F 2E            CS:
01A3:0350 803E8B4A00    CMP     BYTE PTR [4A8B],00                CS:4A8B=00
-t

AX=01A3  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=0355   NV UP EI PL ZR NA PE NC
01A3:0355 7528          JNZ     037F
-t

AX=01A3  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=0357   NV UP EI PL ZR NA PE NC
01A3:0357 2E            CS:
01A3:0358 A19B56        MOV     AX,[569B]                         CS:569B=0192
-_
```

```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program
01A3:0357 2E            CS:
01A3:0358 A19B56        MOV     AX,[569B]                         CS:569B=0192
-t

AX=0192  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=035B   NV UP EI PL ZR NA PE NC
01A3:035B 2E            CS:
01A3:035C A38F4A        MOV     [4A8F],AX                         CS:4A8F=0192
-t

AX=0192  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=035F   NV UP EI PL ZR NA PE NC
01A3:035F 2E            CS:
01A3:0360 803E8C4A00    CMP     BYTE PTR [4A8C],00                CS:4A8C=00
-t

AX=0192  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=0365   NV UP EI PL ZR NA PE NC
01A3:0365 7411          JZ      0378
-t

AX=0192  BX=01A3  CX=7286  DX=2ABC  SP=49D7  BP=0001  SI=2873  DI=01A3
DS=01A3  ES=4000  SS=01A3  CS=01A3  IP=0378   NV UP EI PL ZR NA PE NC
01A3:0378 0E            PUSH    CS
-_
```



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 我对二分查找的本质有了更深刻的理解，了解了ES的声明方式。

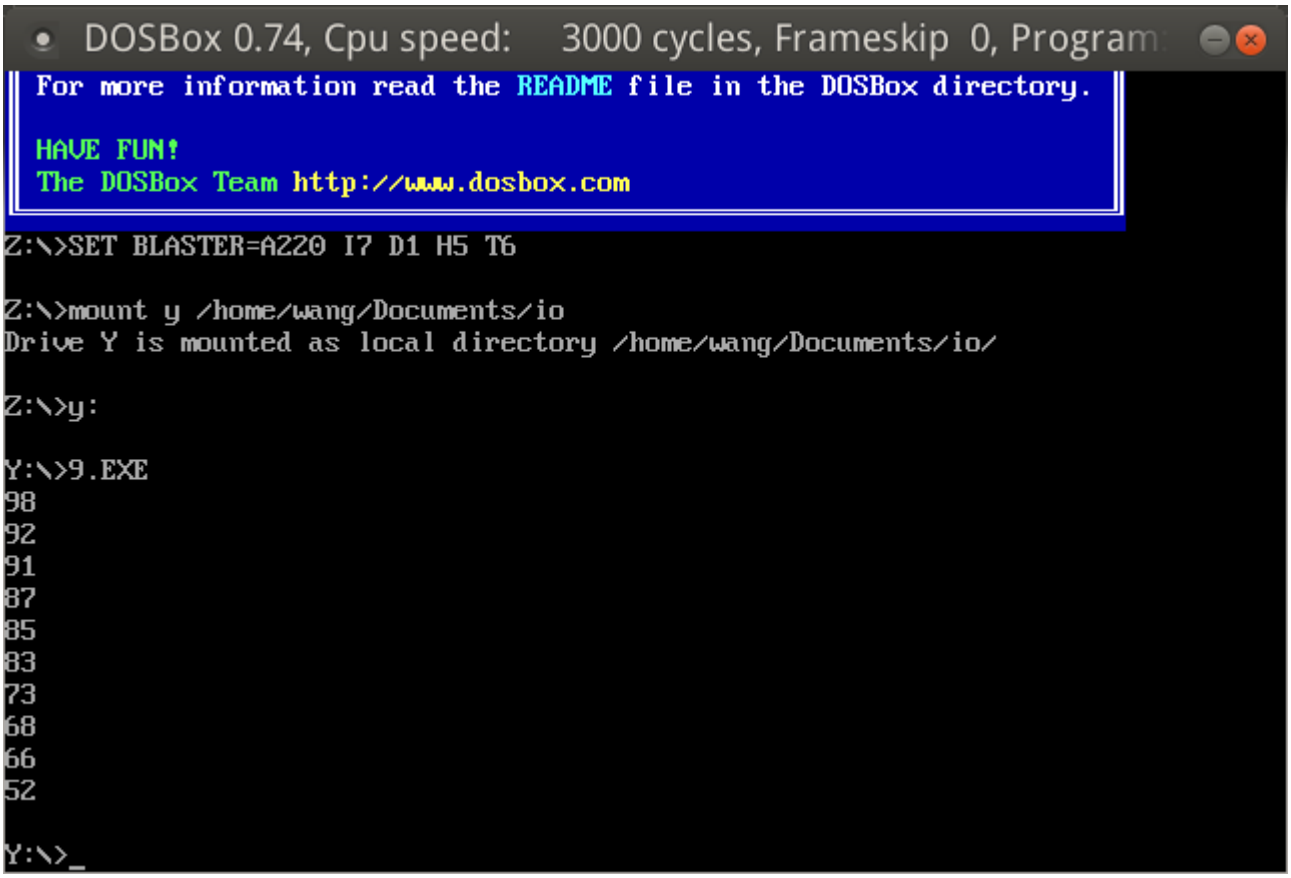
> 4. 在内存中有一个数组，里面是放着10个学生的某科的成绩，分别是：85,73,92,66,91,98,52,87,83,68，请用冒泡排序法将这10个数从大到小排序，并将排序的结果在屏幕上打印出来，要求一个数一行的格式输出。

1. 源程序

```
.8086
.MODEL SMALL
.STACK
.DATA
    SCORE DB 85,73,92,66,91,98,52,87,83,68
    ENDL DB 0AH,0DH,'$'
.CODE
START:
    MOV AX,@DATA
    MOV DS,AX
    MOV DI,OFFSET SCORE
    MOV AX,0
    MOV BX,0
    MOV CX,09H
    MOV DX,0
C1:
    MOV DX,CX
    MOV DI,OFFSET SCORE
C2:
    MOV AL,[DI]
    MOV BL,[DI+1]
    CMP AL,BL
    JB CHANGE
    JMP NEXT
CHANGE:
    MOV [DI],BL
    MOV [DI+1],AL
NEXT:
    INC DI
    LOOP C2
    MOV CX,DX
    LOOP C1

    MOV CX,10
    MOV DI,OFFSET SCORE
PRINT:
    MOV AH,0
    MOV AL,[DI]
    MOV BL,10
    DIV BL
    MOV DX,AX
    ADD DX,3030H
    MOV AH,02H
    INT 21H
    MOV DL,DH
    INT 21H
    MOV DX,OFFSET ENDL
    MOV AH,09H
    INT 21H
    INC DI
    LOOP PRINT
    MOV AH,4CH
    INT 21H
```

```
END START
```

2. 运行结果贴图



3. 编程与调试心得（遇到的问题和解决的办法，以及获得的收获）

> 要注意在排序的过程中对CX值的保存。以及数组元素大小比较。
>
> 但是很奇怪的是在我的WindowsXP上的CMD中无法正确执行，但是在DOSBOX中可以执行，所以。。。