



東南大學  
SOUTHEAST UNIVERSITY

# 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

```
1  .8086
2  .MODEL SMALL
3  .DATA
4  ORG 100H ;CS起始地址
5  STR DB 20,0,20 DUP(20H) ;字符串STR
6  .CODE
7  START:
8      MOV AX,@DATA
9      MOV DS,AX ;数据段地址
10     LEA DX,STR ;将STR的相对于数据段首地址的偏移地址放进DX
11     MOV AH,0AH
12     INT 21H ;从键盘输入一个字符串
13
14     MOV DL,0AH
15     MOV AH,02
16     INT 21H ;输出DL中的换行
17     MOV DL,0DH
18     MOV AH,02
19     INT 21H ;输出回车
20
21     MOV BL,STR[1]
22     MOV BH,0 ;BX=20
23     MOV BYTE PTR STR[BX+2],'$' ;末尾加'$'
24     LEA DX,STR+2
25     MOV AH,9
26     INT 21H ;输出这一句话
27     MOV AH,4CH
28     INT 21H ;返回DOS
29 END START
```

### 2. 运行结果贴图

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program
Welcome to DOSBox v0.74

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount y /home/wang/Documents/io
Drive Y is mounted as local directory /home/wang/Documents/io/

Z:\>y:

Y:\>cd 2

Y:\>2ND.EXE
wang
wang
Y:\>_
```

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

```

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

```

## 2. 运行结果贴图

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program
AX=076B BX=FFFF CX=FE4C DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=075A ES=075A SS=076C CS=076A IP=0003  NU UP EI PL NZ NA PO NC
076A:0003 8ED8          MOV     DS,AX
-t
AX=076B BX=FFFF CX=FE4C DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=076B ES=075A SS=076C CS=076A IP=0005  NU UP EI PL NZ NA PO NC
076A:0005 8D1E0000     LEA     BX,[0000]          DS:0000=0100
-t
AX=076B BX=0000 CX=FE4C DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=076B ES=075A SS=076C CS=076A IP=0009  NU UP EI PL NZ NA PO NC
076A:0009 A00A00      MOV     AL,[000A]          DS:000A=09
-t
AX=0709 BX=0000 CX=FE4C DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=076B ES=075A SS=076C CS=076A IP=000C  NU UP EI PL NZ NA PO NC
076A:000C D7          XLAT
-t
AX=0751 BX=0000 CX=FE4C DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=076B ES=075A SS=076C CS=076A IP=000D  NU UP EI PL NZ NA PO NC
076A:000D A20B00      MOV     [000B],AL          DS:000B=00

```

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 $AL \leftarrow ((BX) + (AL))$ . 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

```

1  .8086
2  .MODEL SMALL
3  .STACK
4  .DATA
5      N EQU 20
6      MAXLEN BYTE N
7      ACTLEN BYTE ?
8      STRING BYTE N DUP('$')
9      TRUE BYTE 0AH,0DH,'It is a palindrome',0AH,0DH,'$' ;开头加0AH、0DH是为了防止回车
把输入冲掉
10     FALSE BYTE 0AH,0DH,'It is NOT a palindrome',0AH,0DH,'$'
11  .CODE
12  START:
13      MOV AX,@DATA
14      MOV DS,AX
15      MOV AH,0AH
16      MOV DX,OFFSET MAXLEN
17      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。
18      MOV AL,ACTLEN
19      MOV AH,0
20      MOV BL,2
21      DIV BL
22      CMP AH,1
23      JE ITSODD;如果是奇数字数的字符串，跳到"It's odd"
24      CMP AL,1
25      JE ITSTWO;如果是两个字符的字符串，跳到"It's two"
26      ADD AL,AH;奇数个数和偶数个数的区别就在这句话
27      MOV AH,0
28      MOV BL,ACTLEN
29      MOV BH,0
30      MOV SI,BX
31      MOV BX,OFFSET STRING
32      DEC SI
33  ITSEVEN:
34      CMP SI,AX
35      JNA T
36      MOV DL,[BX+SI]
37      MOV DH,[BX]
38      CMP DL,DH
39      JNZ F
40      INC BX
41      SUB SI,2;因为BX加了一，所以BX+SI要想减一，就需要SI减二
42      JMP ITSEVEN
43  ITSODD:
44      MOV AH,0
45      MOV BL,ACTLEN
46      MOV BH,0
47      MOV SI,BX
48      MOV BX,OFFSET STRING
49      DEC SI
50  JUDGEODD:
51      CMP SI,AX

```

```

52     JNA T
53     MOV DL,[BX+SI]
54     MOV DH,[BX]
55     CMP DL,DH
56     JNZ F
57     INC BX
58     SUB SI,2
59     JMP JUDGEODD
60 ITSTWO:
61     MOV BX,OFFSET STRING
62     MOV DL,[BX+1]
63     MOV DH,[BX]
64     CMP DL,DH
65     JNE F;如果这两个字符不相同,就不是。否则直接向下执行T
66 T:  MOV DX,OFFSET TRUE
67     JMP QUIT
68 F:  MOV DX,OFFSET FALSE
69     JMP QUIT
70 QUIT:
71     MOV AH,09H
72     INT 21H
73     MOV AH,4CH
74     INT 21H
75 END START

```

## 2. 运行结果贴图

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program
Drive C is mounted as local directory /home/wang/Documents/io/4/
Z:\>C:
C:\>4.EXE
wang
It is NOT a palindrome
C:\>4.EXE
wan
It is NOT a palindrome
C:\>4.EXE
wa
It is NOT a palindrome
C:\>4.EXE
w
It is a palindrome
C:\>4.EXE
abdcba
It is a palindrome
C:\>

```

## 3. 编程与调试心得（遇到的问题 and 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



```

1  .8086
2  .MODEL SMALL
3  .STACK
4  .DATA
5      BUFF BYTE 0,1,2,3,4,5,6,7,8,9
6  .CODE
7  START:
8      MOV AX,@DATA
9      MOV DS,AX
10     MOV CX,10;计数初始化为10
11     LEA SI,BUFF;BUFF首地址给SI
12     MOV AH,01H
13     INT 21H;从键盘输入一个字符
14     SUB AL,30H;将ASCII码变成数字
15     MOV AH,0
16 NEXT:
17     MOV BL,[SI]
18     MOV BH,0
19     CMP AX,BX
20     JE DEL;找到了就跳至删除
21     INC SI
22     LOOP NEXT
23     JMP STOP;找不到就停
24 DEL:
25     MOV DI,SI;记住当前元素的位置
26     INC SI
27     MOV AL,[SI]
28     MOV AH,0
29     MOV [DI],AX;后面的内容前移
30     LOOP DEL
31     MOV [SI],0;找到的最后一位换成0
32     LEA SI,BUFF;SI指向BUFF首地址
33     MOV CX,9;设计数为9
34     MOV AH,02H
35     MOV DL,0DH
36     INT 21H;回车
37     MOV DL,0AH
38     INT 21H;换行
39 DO:
40     MOV DL,[SI]
41     MOV DH,0
42     ADD DX,3030H;十六进制转ASCII
43     INC SI
44     INT 21H;一个个输出此时BUFF内容
45     LOOP DO
46 STOP:
47     MOV AH,4CH
48     INT 21H;退回DOS
49 END START

```

## 2. 运行结果截图

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program
It is a palindrome
Y:\4>4.EXE
It is NOT a palindrome
Y:\4>4.EXE
It is NOT a palindrome
Y:\4>cd ..
Y:\>5
Illegal command: 5.
Y:\>cd 5
Y:\5>5.EXE
2
013456789
Y:\5>5.EXE
3
012456789
Y:\5>5.EXE
4
012356789
Y:\5>_
```

### 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

```

1  .8086
2  .MODEL SMALL
3  .STACK
4  .DATA
5      STRING DB 'You must input 0~9, OR `Q` OR `q`,0AH,0DH','$'
6      BUF DB 8 DUP(30H),'$'
7  .CODE
8  START:
9      MOV AX,@DATA
10     MOV DS,AX
11     MOV AH,01H
12     INT 21H
13     CMP AL,'q'
14     JE EXIT
15     CMP AL,'Q'
16     JE EXIT;输入'q'或'Q'时退出
17     CMP AL,'0'
18     JL WRONG
19     CMP AL,'9'
20     JG WRONG;输入小于零或者大于九时错误
21
22     MOV BL,AL
23     MOV AH,02H
24     MOV DL,0DH
25     INT 21H
26     MOV DL,0AH
27     INT 21H;回车换行
28
29     MOV AL,BL
30     SUB AL,30H;ASCII码转数字
31     MOV DX,0
32     MOV AH,0
33     MOV BX,OFFSET BUF+7
34 AGAIN:
35     MOV DX,0;余数清零
36     MOV CX,2
37     DIV CX
38     ADD DL,30H;转ASCII码
39     MOV [BX],DL;将零或一附给BX所指内存中的元素
40     DEC BX;向左移一位
41     AND AX,AX
42     JNE AGAIN;AX不为零就继续循环
43     JMP RIGHT;其实这句可有可无,当时随便就写上了
44 RIGHT:
45     MOV AH,09H
46     LEA DX,BUF
47     INT 21H;输出BUF内容
48     JMP EXIT
49 WRONG:
50     MOV AH,09H
51     LEA DX,STRING
52     INT 21H;输出STRING内容
53     JMP EXIT

```

```

54  EXIT:
55      MOV AH, 4CH
56      INT 21H;退出到DOS
57  END START

```

## 2. 运行结果截图

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program
Y:\>cd 5
Y:\5>5.EXE
2
013456789
Y:\5>5.EXE
3
012456789
Y:\5>5.EXE
4
012356789
Y:\5>cd ..
Y:\>cd 6
Y:\6>6.EXE
2
00000010
Y:\6>6.EXE
q
Y:\6>6.EXE
oYou must input 0~9, OR `Q` OR `q`
Y:\6>

```

## 3. 编程与调试心得（遇到的问题 and 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

```

1  .8086
2  .MODEL SMALL
3  .STACK
4  .DATA
5      STRING BYTE 6 DUP(20H), '$'
6      ENDL BYTE 0AH, 0DH, '$'
7      ERRMSG BYTE 'You must input a number from 1~6', 0AH, 0DH, '$'
8  .CODE
9  START:
10     MAIN PROC
11     MOV AX, @DATA
12     MOV DS, AX
13     MOV AH, 01H
14     INT 21H
15     CMP AL, '1'
16     JL ERR; 小于1的ASCII码错误
17     CMP AL, '6'
18     JG ERR; 大于6的ASCII码错误
19     SUB AL, 30H
20     MOV CL, AL
21     LEA DX, ENDL
22     MOV AH, 09H
23     INT 21H
24     MOV DX, 0
25     LEA BX, STRING+5
26     MOV AL, 01H
27     MOV AH, 0
28     MOV CH, 0
29     CALL DAC
30     CALL PRINT
31     MAIN ENDP
32
33     DAC PROC
34 AGAIN:
35     MOV DX, CX
36     MUL DX
37     LOOP AGAIN
38     DAC ENDP
39
40     PRINT PROC
41     MOV CX, 10
42 DO:
43     MOV DX, 0
44     DIV CX
45     ADD DL, 30H
46     DEC BX
47     MOV [BX], DL
48     AND AX, AX
49     JNE DO
50     MOV DX, BX
51     MOV AH, 09H
52     INT 21H
53     MOV AH, 4CH

```

```

54     INT 21H
55     PRINT ENDP
56
57 ERR:
58     LEA DX, ENDL
59     MOV AH, 09H
60     INT 21H
61     LEA DX, ERRMSG
62     MOV AH, 09H
63     INT 21H
64     MOV AH, 4CH
65     INT 21H
66 END START

```

## 2. 运行结果截图

```

Y:\>cd 6
Y:\6>6.EXE
2
00000010
Y:\6>6.EXE
q
Y:\6>6.EXE
oYou must input 0~9, OR `Q` OR `q`
Y:\6>cd ..
Y:\>cd 7
Y:\7>7.EXE
3
6
Y:\7>7.EXE
6
720
Y:\7>7.EXE
5
120
Y:\7>

```

## 3. 编程与调试心得（遇到的问题 and 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。

- 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

```

1  .8086
2  .MODEL SMALL
3  EXTRA SEGMENT
4      ARRAY BYTE 10,1,2,3,4,5,6,7,8,9,10
5  EXTRA ENDS
6  .CODE
7  ASSUME ES:EXTRA
8  START:
9      MOV BX,EXTRA
10     MOV ES,BX
11     LEA DI,ES:ARRAY
12
13     MOV AL,7
14
15     MOV CX,0
16     MOV CL,ES:[DI]
17     INC DI
18     MOV DX,DI
19     ADD DX,CX
20     MOV SI,DX
21     MOV DX,0
22
23     CMP AL,ES:[DI]
24     MOV BX,DI
25     JB NOTFOUND
26     JE FOUND
27     CMP AL,ES:[SI-1]
28     MOV BX,SI
29     JA NOTFOUND
30     JE FOUND
31
32  WORK:
33     MOV BX,DI
34     ADD BX,SI
35     SHR BX,1
36     CMP AL,ES:[BX]
37     JZ FOUND
38     PUSHF
39     CMP BX,DI
40     JZ NOTFOUND
41     POPF
42     JL LESS
43     MOV DI,BX
44     JMP WORK
45  LESS:
46     MOV SI,BX
47     JMP WORK
48  NOTFOUND:
49     STC
50     JMP EXIT
51  FOUND:
52     CLC
53     MOV BX,SI

```

```

54     JMP EXIT
55 EXIT:
56     MOV AH,4CH
57     INT 21H
58     END START

```

## 2. 运行结果截图

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

```

AX=FFFF BX=0770 CX=FE9B DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=075A ES=075A SS=0771 CS=076A IP=0003  NV UP EI PL NZ NA PO NC
076A:0003 8EC3      MOV     ES,BX
-t

AX=FFFF BX=0770 CX=FE9B DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=075A ES=0770 SS=0771 CS=076A IP=0005  NV UP EI PL NZ NA PO NC
076A:0005 8D3E0000   LEA     DI,[0000]          DS:0000=20CD
-t

AX=FFFF BX=0770 CX=FE9B DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=075A ES=0770 SS=0771 CS=076A IP=0009  NV UP EI PL NZ NA PO NC
076A:0009 B007      MOV     AL,07
-t

AX=FF07 BX=0770 CX=FE9B DX=0000 SP=0400 BP=0000 SI=0000 DI=0000
DS=075A ES=0770 SS=0771 CS=076A IP=000B  NV UP EI PL NZ NA PO NC
076A:000B B90000   MOV     CX,0000
-t

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
-

```



```
• 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=000B 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=000B 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=000B 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=000B 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
_

```

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

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
-t

AX=0192 BX=01A3 CX=7286 DX=2ABC SP=49D5 BP=0001 SI=2873 DI=01A3
DS=01A3 ES=4000 SS=01A3 CS=01A3 IP=0379  NV UP EI PL ZR NA PE NC
01A3:0379 1F          POP     DS
-t

AX=0192 BX=01A3 CX=7286 DX=2ABC SP=49D7 BP=0001 SI=2873 DI=01A3
DS=7202 ES=4000 SS=01A3 CS=01A3 IP=037A  NV UP EI PL ZR NA PE NC
01A3:037A BAF24A      MOV     DX,4AF2
-t

AX=0192 BX=01A3 CX=7286 DX=4AF2 SP=49D7 BP=0001 SI=2873 DI=01A3
DS=7202 ES=4000 SS=01A3 CS=01A3 IP=037D  NV UP EI PL ZR NA PE NC
01A3:037D EB08      JMP     0387
-t

AX=0192 BX=01A3 CX=7286 DX=4AF2 SP=49D7 BP=0001 SI=2873 DI=01A3
DS=7202 ES=4000 SS=01A3 CS=01A3 IP=0387  NV UP EI PL ZR NA PE NC
01A3:0387 E8BA00      CALL    0444
-

```

### 3. 编程与调试心得（遇到的问题 and 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

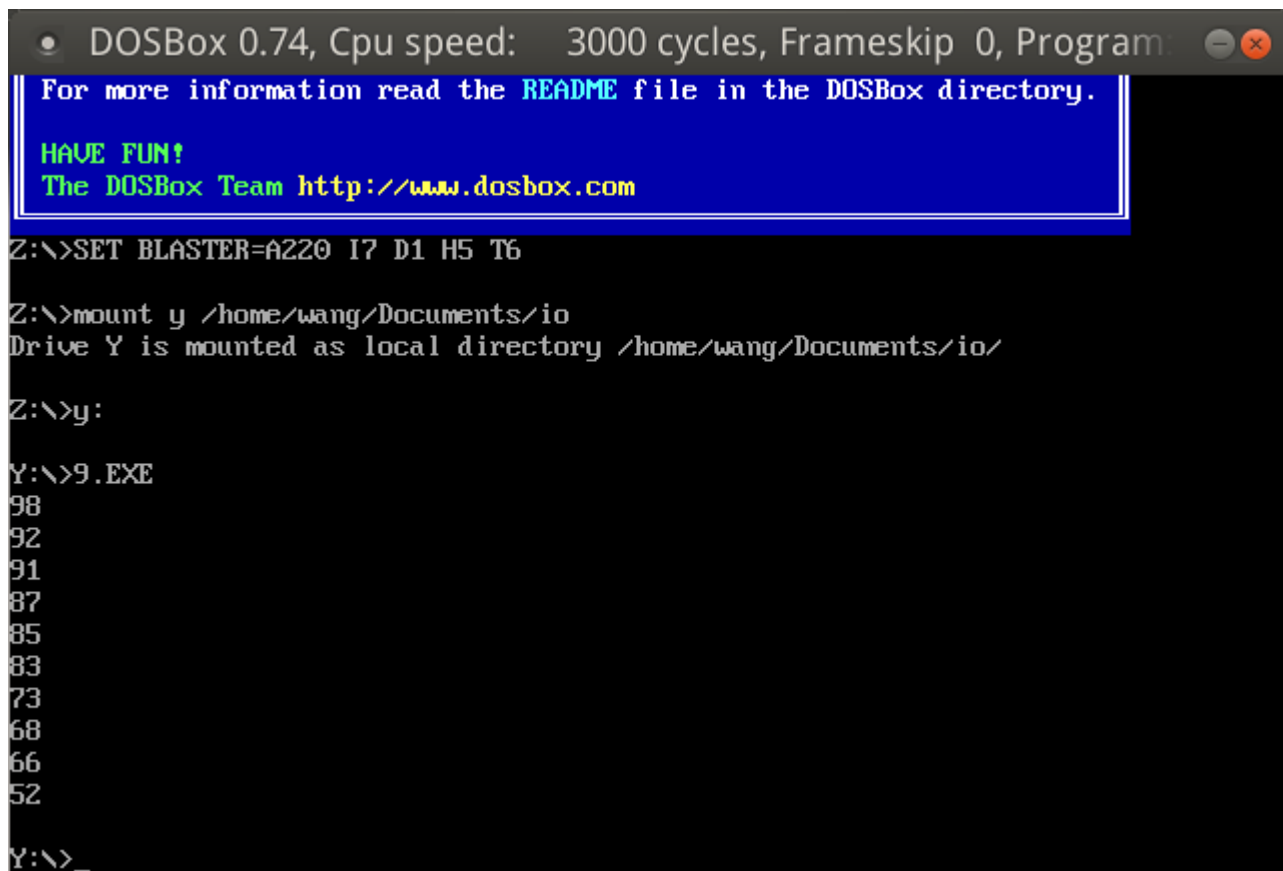


```

1  .8086
2  .MODEL SMALL
3  .STACK
4  .DATA
5      SCORE DB 85,73,92,66,91,98,52,87,83,68
6      ENDL DB 0AH,0DH,'$'
7  .CODE
8  START:
9      MOV AX,@DATA
10     MOV DS,AX
11     MOV DI,OFFSET SCORE
12     MOV AX,0
13     MOV BX,0
14     MOV CX,09H
15     MOV DX,0
16 C1:
17     MOV DX,CX
18     MOV DI,OFFSET SCORE
19 C2:
20     MOV AL,[DI]
21     MOV BL,[DI+1]
22     CMP AL,BL
23     JB CHANGE
24     JMP NEXT
25 CHANGE:
26     MOV [DI],BL
27     MOV [DI+1],AL
28 NEXT:
29     INC DI
30     LOOP C2
31     MOV CX,DX
32     LOOP C1
33
34     MOV CX,10
35     MOV DI,OFFSET SCORE
36 PRINT:
37     MOV AH,0
38     MOV AL,[DI]
39     MOV BL,10
40     DIV BL
41     MOV DX,AX
42     ADD DX,3030H
43     MOV AH,02H
44     INT 21H
45     MOV DL,DH
46     INT 21H
47     MOV DX,OFFSET ENDL
48     MOV AH,09H
49     INT 21H
50     INC DI
51     LOOP PRINT
52     MOV AH,4CH
53     INT 21H

```

## 2. 运行结果贴图



The screenshot shows a DOSBox 0.74 window. At the top, a status bar displays 'DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program'. A blue message box with a white border is overlaid on the top half of the window. It contains the text: 'For more information read the README file in the DOSBox directory.', 'HAVE FUN!', and 'The DOSBox Team http://www.dosbox.com'. Below the message box, the command prompt shows the following commands and output:

```
Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount y /home/wang/Documents/io
Drive Y is mounted as local directory /home/wang/Documents/io/

Z:\>y:

Y:\>9.EXE
98
92
91
87
85
83
73
68
66
52

Y:\>_
```

## 3. 编程与调试心得（遇到的问题 and 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。。。