汇编语言程序设计

期末报告

学　　号_____20020203_____

姓　　名_____王思哲_____

指导教师_____魏坚华_____

提交日期_____2022 年 5 月 28 日_____

成绩评价表

报告内容	报告结构	报告最终成绩
☐丰富正确 ☐基本正确 ☐有一些问题 ☐问题很大	☐完全符合要求 ☐基本符合要求 ☐有比较多的缺陷 ☐完全不符合要求	
报告与练习题功能一致性	**报告图表**	**总体评价**
☐完全一致 ☐基本一致 ☐基本不一致	☐符合规范 ☐基本符合规范 ☐有一些错误 ☐完全不正确	

教师签字:_____

目录

（一） 题目一

一、 题目描述

从键盘输入一个 6 位的十进制整数，从右到左两位一组，计算输出 3 组之和，并 3 组数据按大到小排序输出。

二、 编程设计思路

1、 概述

根据题目描述，程序应该分为六个部分，一是变量的定义以及初始化，二是输入模块，三是数的转换模块，四是加法运算模块，五是排序模块，六是显示输出模块。

读取并判断输入字符的合法性后，从数组尾部开始，每两个字符表示一个两位数，使用压缩的 BCD 形式表示，即高 4 位表示十位，低四位表示个位。对三个数进行加法运算，进行 DAA 调整并考虑进位问题，将加法结果从高位到低位按字符输出即可。对三个数进行排序，就是分别比较三个数两两之间的大小，定义三个寄存器永远按照从大到小顺序保存数据，故在比较过程中，如果出现数与寄存器排位不一致的情况，就交换寄存器的内容，直到三个寄存器的内容按照从大到小的顺序排列，则将三个数依次输出即可。

2、 详细设计

（1） 变量定义及初始化

根据题目描述“一个 6 位十进制整数”，考虑按字符读取，所以应该定义一个长度为 6 的数组 *BUF*，数据类型为 DB 类型。又因为要将这一个六位数转换成三个三位数，所以应该定义三个变量 *NUM1,NUM2,NUM3* 来存储这三个两位数，数据类型为 DB 类型。此外，再定义一个变量 *SUM* 来存储三个数的和。

定义一些字符串 *STR0---STR3* 来储存一些提示信息，便于用户在使用程序时清晰的知道当前的程序状态以及用户应该进行什么操作。

（2） 输入模块

输入模块主要的作用是循环读取用户用过键盘输入的字符，并检查其 ascii 码是否在数字 0-9 的 ascii 码之间。如果符合要求，则将该字符的 ascii 码转换为 BCD 码后，保存在存储单元对应位置，不符合则从头开始重新输入，直到 6 个字符全部符合要求为止。

（3） 转换模块

转换模块是将 6 个字符转换为表示 3 个两位数。主要方法是获取数组数组基地址并加 5，使指针指向数组末尾。每两个字符为一组，将从右往左的第一个字符的 BCD 码送 AL 寄存器高位、第二个字符的 BCD 码送寄存器低位保存。对三组字符进行同样的操作，将“拼接”完成的两位数存入变量 *NUM1,NUM2,NUM3* 中。

（4） 加法运算模块

累加三个数。需要注意的是，由于使用的是高四位表示十进制数十位，低四位表示十进制数个位，故在两个数相加之后需要进行 DAA 调整，并将调整后的结果保存在 *AL* 寄存器中。同时，还需要考虑到进位的问题，故调用 ADC 指令，将进位标志位与 0 相加，保存到 AH 寄存器中即可。

（5） 排序模块

三个数的排序问题。由于考虑到只有三个数，则不需要对其使用排序算法，直接两两之间比大小即可。

这里使用 *AL,BL,CL* 三个寄存器保存 *NUM1,NUM2,NUM3* 三个数。定义 *AL* 中永远保存三个数中最大的数，*CL* 中永远保存三个数中最小的数。则对三个数进行比较，不断的对三个寄存器中的内容进行交换知道符合定义的规则，则排序完成。例如，如果 *AL>BL* 且 *AL>CL*，则 *AL* 中的数不变，比较 *BL,CL* 中的大小，如果 *BL>CL* 则不用交换，直接将三个寄存器内容保存到变量并按字符输出即可。否则就在不断的交换后，再输出。

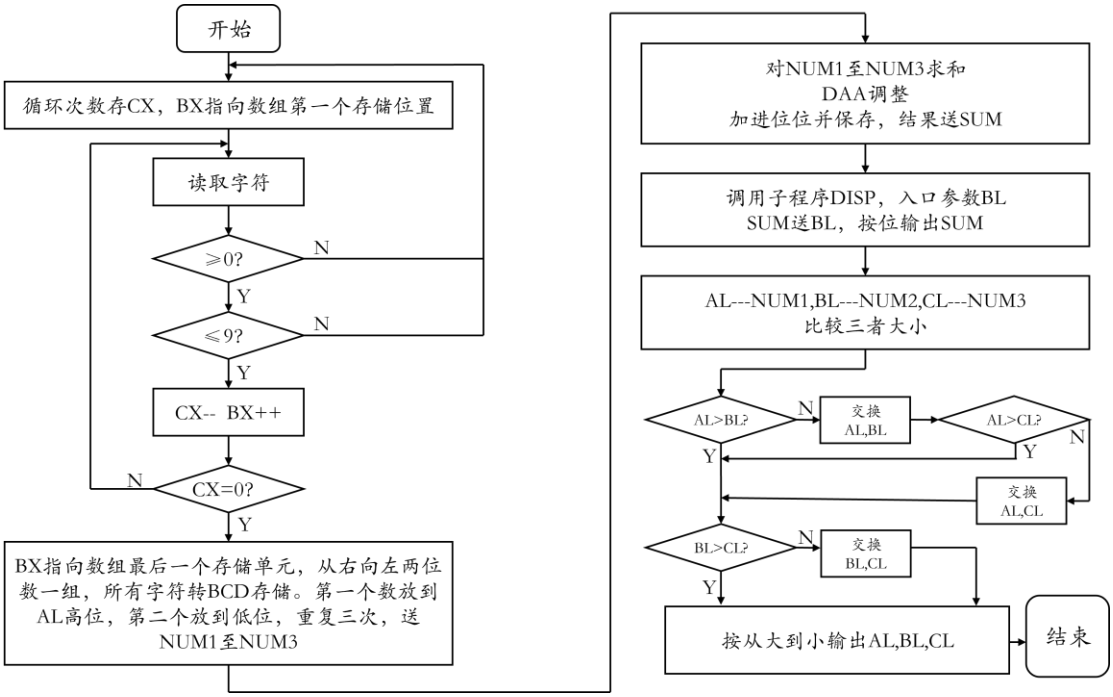
（6） 输出模块

定义一个输出函数，将两位数的字符按位的先后顺序输出。

入口参数：*BL*

这个函数主要完成两位整数的字符输出。具体实现是：根据传入的 BL 寄存器内容，先取其高四位，转换成 ascii 码后进行字符输出；随后取其低四位，转换成 ascii 码后进行字符输出。至此，就完成了两位十进制整数的输出。

三、 程序流程图



四、 程序源代码及注解

```
;默认采用ML6.11汇编程序
DATAS SEGMENT
    ;此处输入数据段代码
    BUF DB 0,0,0,0,0,0 ;存储6个字符的数组
    NEWLINE DB 13,10,'$'   ;换行
    STR0 DB 0AH,0DH,'INPUT ERROR! PLEASE INPUT
AGAIN!',0AH,0DH,'$' ;提示信息
    STR1 DB 'INPUT AN INTEGER (6WEI): $'   ;提示信息
    STR2 DB 'SUM: $'    ;提示信息
    STR3 DB 'SORT: $'   ;提示信息
    NUM1 DB 0     ;存第一个数
    NUM2 DB 0     ;存第二个数
    NUM3 DB 0     ;存第三个数
    SUM  DW 0     ;存SUM和
DATAS ENDS

STACKS SEGMENT

STACKS ENDS
    ;此处输入堆栈段代码
CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS
START: ;输错时跳转到这儿
    MOV AX,DATAS
    MOV DS,AX
    ;此处输入代码段代码
AGA:    ;循环运行程序的标志
    ;显示提示信息
    LEA DX,STR1
    MOV AH,9
    INT 21H

    MOV CX,6 ;输入6次
    LEA BX,BUF ;BX指向BUF
LL:  ;输入正确时循环
    MOV AH,1
    INT 21H  ;1号功能输入一个字符
    ;判断字符是否合法
    CMP AL,30H
    JB  ERR     ;小于0就跳ERR
    CMP AL,39H
    JA  ERR     ;大于9跳ERR
```

```
    SUB AL,30H ;转BCD
    MOV [BX],AL;送对应存储单元
    INC BX    ;BX--，指向下一个存储单元
    LOOP LL     ;循环起来
    JMP NEXT  ;循环结束，进行下一步
ERR:
    LEA DX,STR0
    MOV AH,9
    INT 21H   ;输出错误
    JMP START ;跳最开始 ，重新输入

NEXT:
    ;取字符，每两个字符拼成一个两位数，高四位是十位，低四位是个位
    LEA BX,BUF
    ADD BX,5 ;指向最后一个单元
    MOV AL,[BX]
    MOV CL,4   ;高位挪入高四位
    SHL AL,CL
    DEC BX
    OR  AL,[BX];或上低四位拼起来
    MOV NUM1,AL;送NUM1
    DEC BX

    MOV AL,[BX]
    MOV CL,4   ;高位挪入高四位
    SHL AL,CL
    DEC BX
    OR  AL,[BX];或上低四位拼起来
    MOV NUM2,AL;送NUM2
    DEC BX

    MOV AL,[BX]
    MOV CL,4   ;高位挪入高四位
    SHL AL,CL
    DEC BX
    OR  AL,[BX];或上低四位拼起来
    MOV NUM3,AL;送NUM3

    ;求和SUM
    MOV AX,0
    ADD AL,NUM1
    DAA    ;组合BCD调整
    ADC AH,0 ;加上进位
```

```
    ADD AL,NUM2
    DAA    ;组合BCD调整
    ADC AH,0 ;加上进位

    ADD AL,NUM3
    DAA    ;组合BCD调整
    ADC AH,0 ;加上进位

    MOV BX,AX ;送BX  BH存的是进位，BL存本地和

    ;换行
    LEA DX,NEWLINE
    MOV AH,9
    INT 21H
    ;显示提示信息
    LEA DX,STR2
    MOV AH,9
    INT 21H
    ;显示SUM，按字符显示
    XCHG BL,BH          ;显示高位
    CALL DISP   ;调用显示
    XCHG BL,BH          ;显示低位
    CALL DISP   ;调用显示

    ;比大小
    MOV AL,NUM1
    MOV BL,NUM2
    MOV CL,NUM3

    CMP AL,BL ;AL大就跳
    JA  Z0
    XCHG AL,BL;交换AL和BL
Z0:
    CMP AL,CL;AL大就跳
    JA Z1
    XCHG AL,CL;否则交换AL,CL
Z1:
    CMP BL,CL;BL大就跳
    JA Z2
    XCHG BL,CL   ;否则交换BL,CL
Z2:
    ;保存交换完的结果
    MOV NUM1,AL
    MOV NUM2,BL
```

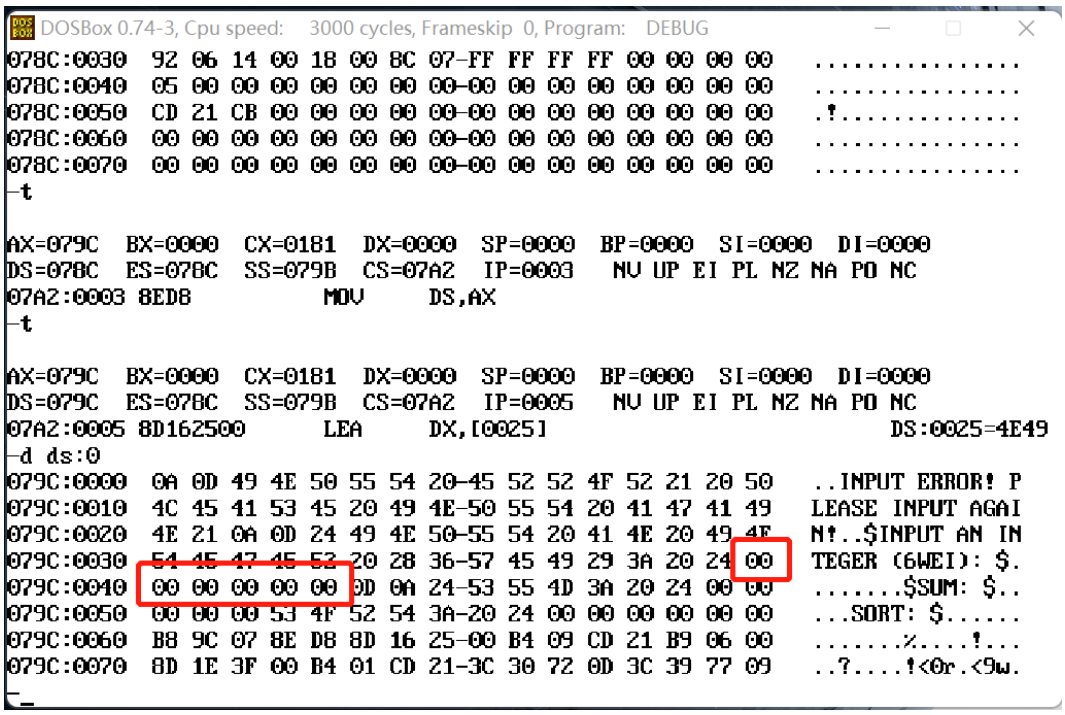
```
    MOV NUM3,CL
    ;换行
    LEA DX,NEWLINE
    MOV AH,9
    INT 21H
    ;显示提示信息
    LEA DX,STR3
    MOV AH,9
    INT 21H
    ;显示输出三个数，从大到小
    ;输出NUM1
    MOV BL,NUM1
    CALL DISP
    ;输出一个空格，和NUM2
    MOV DL,' '
    MOV AH,2
    INT 21H
    MOV BL,NUM2
    CALL DISP
    ;输出一个空格和NUM3
    MOV DL,' '
    MOV AH,2
    INT 21H
    MOV BL,NUM3
    CALL DISP
    ;换行换行
    LEA DX,NEWLINE
    MOV AH,9
    INT 21H
    LEA DX,NEWLINE
    MOV AH,9
    INT 21H
    ;循环运行这个程序
    JMP AGA

    MOV AH,4CH
    INT 21H
DISP PROC  ;入口参数BL
    MOV CL,4
    MOV DL,BL
    SHR DL,CL ;右移4位，输出高位对应的字符
    OR  DL,30H;转ASCII
    MOV AH,2
    INT 21H   ;输出
```

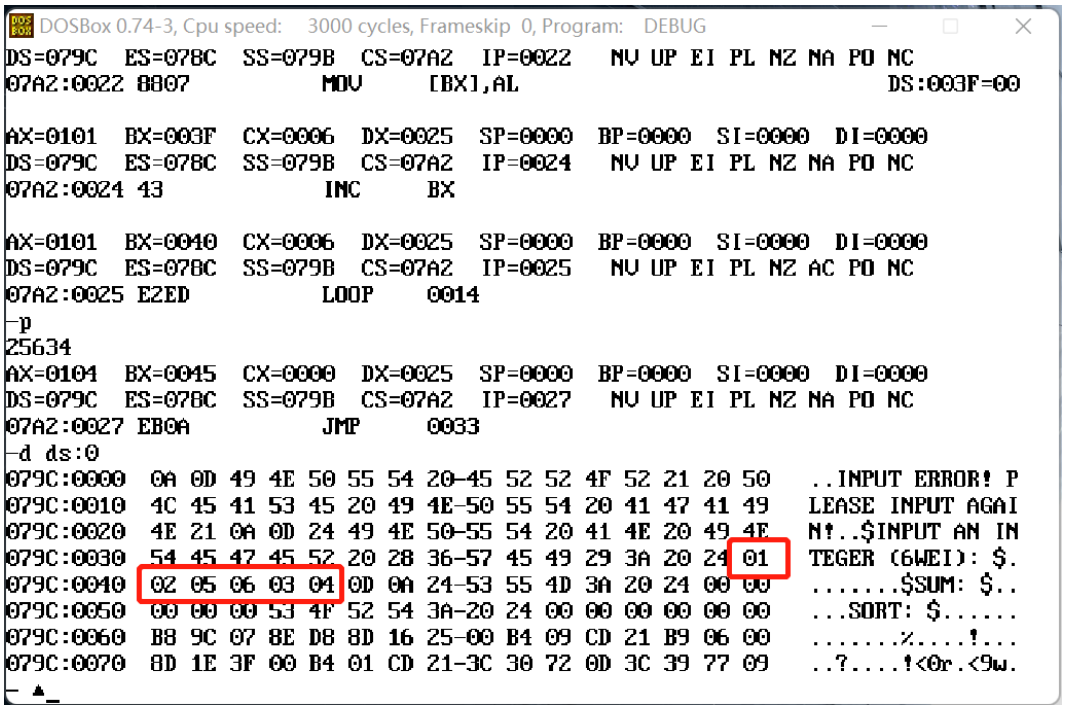
```
    MOV DL,BL
    AND DL,0FH;屏蔽高四位，输出低位对应的字符
    OR  DL,30H;转ASCII
    MOV AH,2
    INT 21H   ;输出
    RET       ;子程序结束
DISP ENDP
CODES ENDS
    END START
```

五、 核心模块调试

1、输入模块

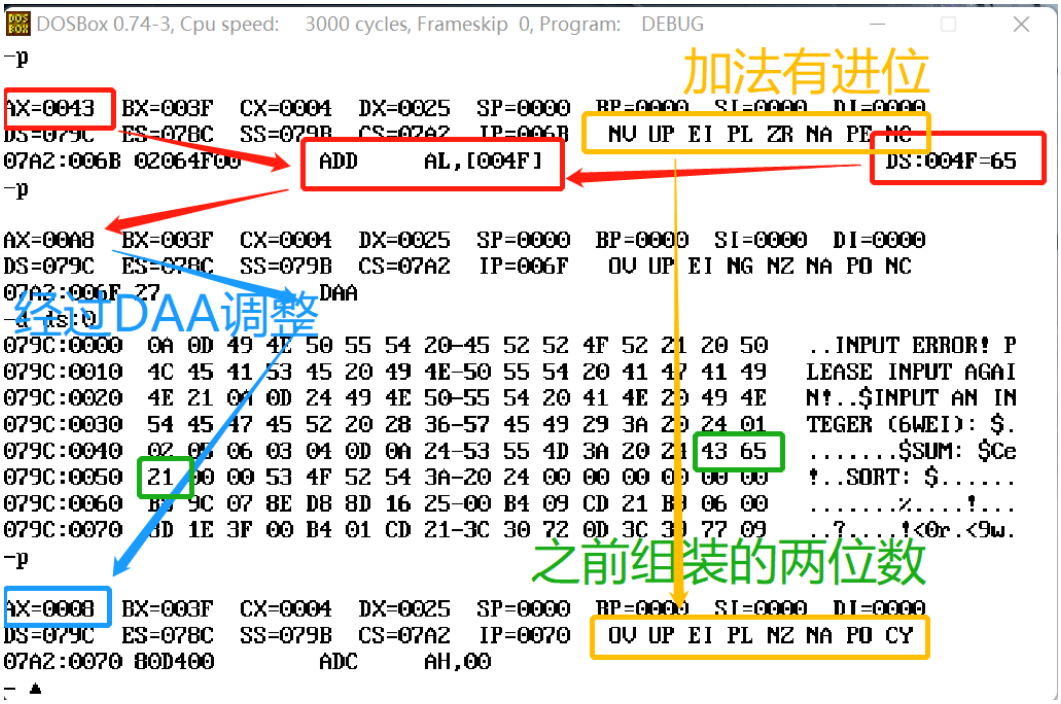
输入 6 个数，成功完成输入

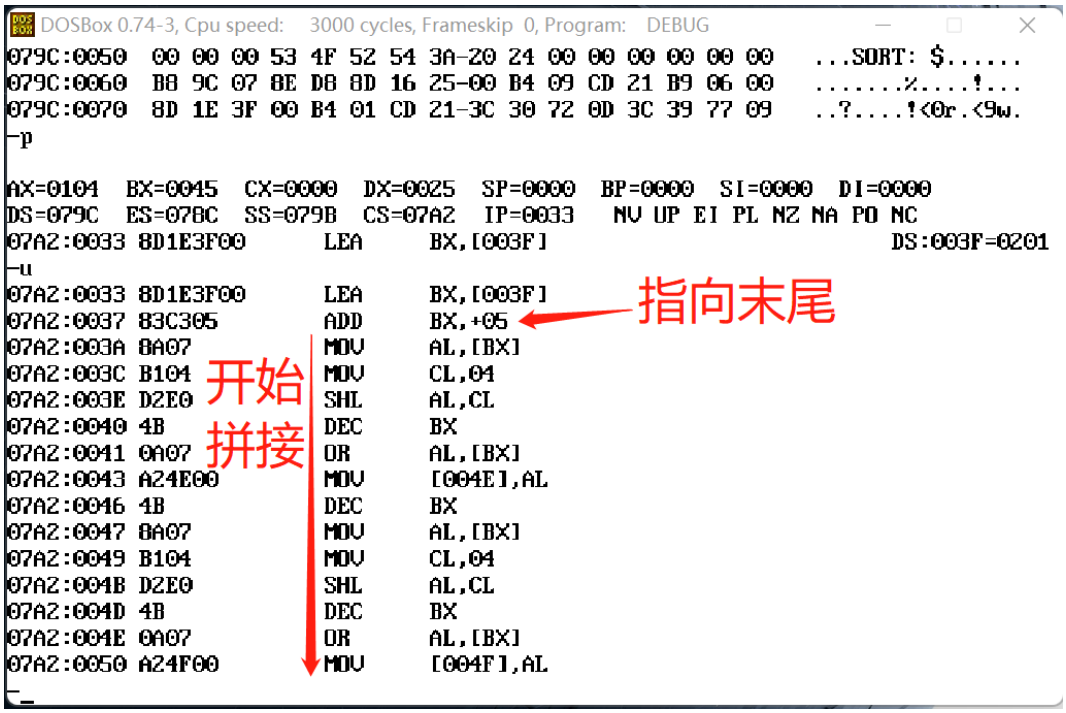




2、转换模块与加法模块

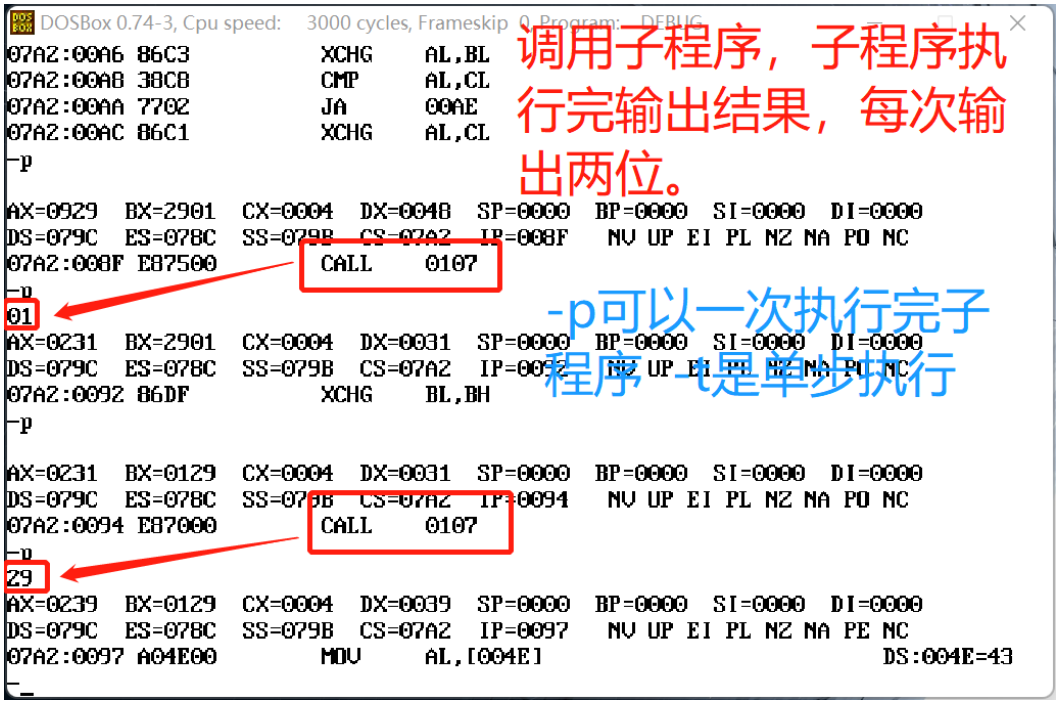
指针指向数组末尾，每两个数一组，按照上文叙述的方法拼接成两位数。拼接完成后进行加法，DAA 调整，使得结果仍为 BCD 表示。然后加上进位位，下右图中 AX 会在执行 ADC 指令后变为 108，即进位位加在了 AH 上。





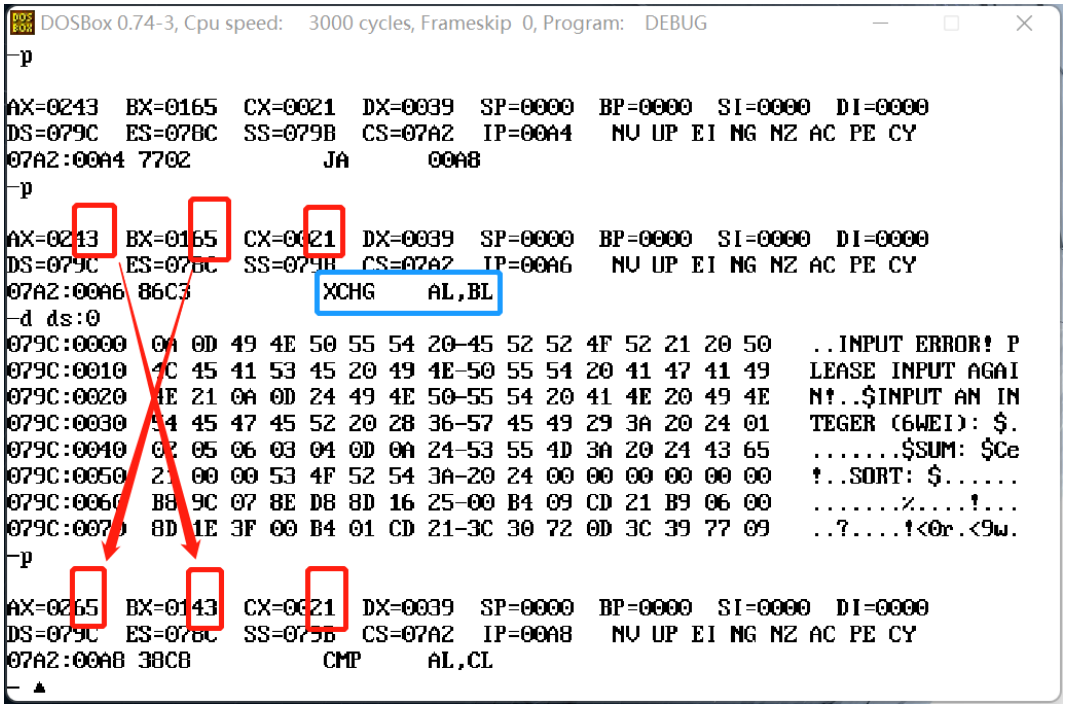
3、输出模块

调用子程序，完成输出。



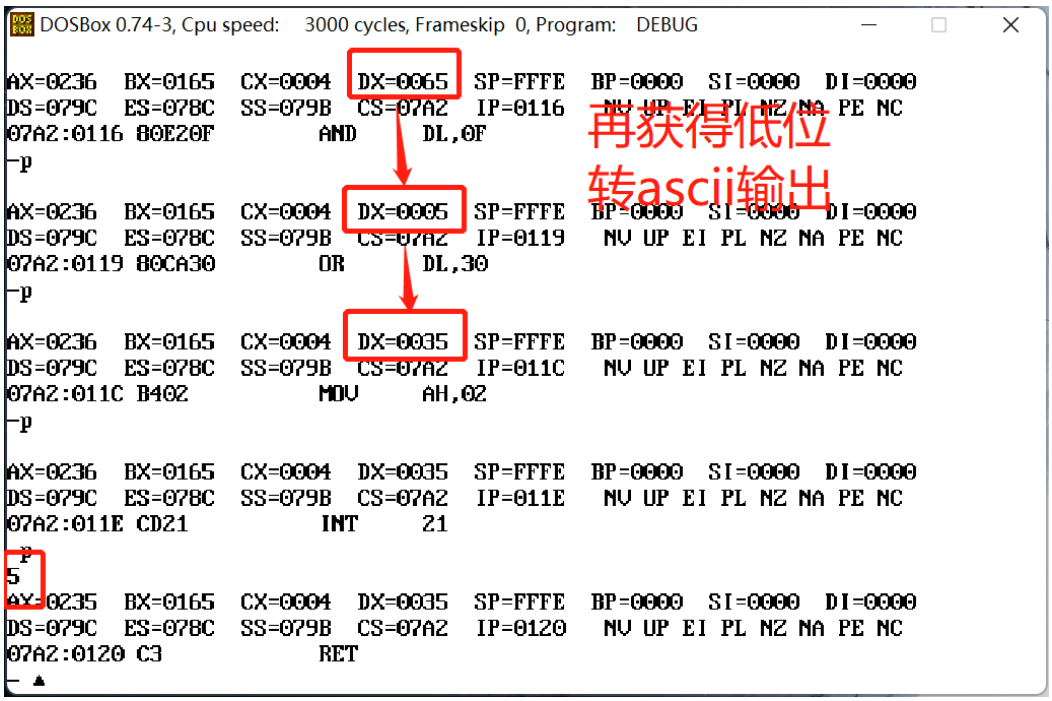
4、排序模块

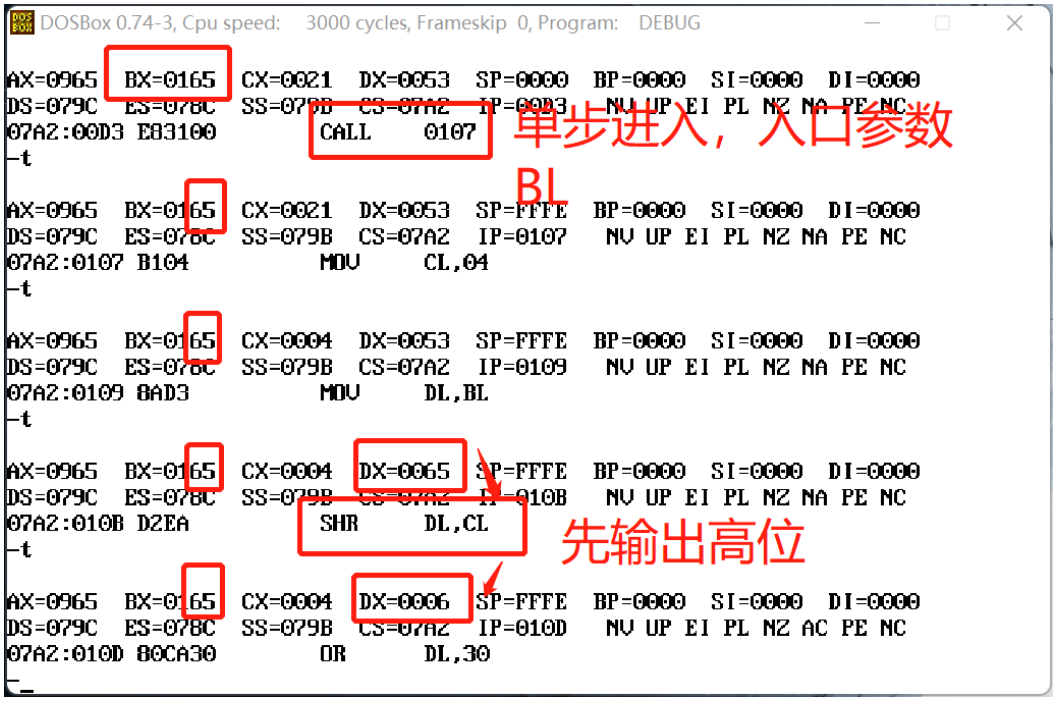
比较大小，如果寄存器中内容不是按照规定的大小顺序排序的，就交换两个寄存器内容。



5、输出模块详解

详解子程序(输出)的运行步骤。





六、程序设计收获体会

本程序实现的是将一个输入的六位十进制整数倒序拆成 3 个两位数，并输出他们的和，再按照从大到小顺序排列并输出。其难点主要在于如何转换成两位数，以及如何在考虑加法的情况下进行两位数的相加。

通过学习我得知，可以用一种压缩 BCD 码的形式进行存储两位十进制数。即，高四位代表高位数字，低四位代表低位数字。然后就可以将两位数进行相加，相加后需要注意的是，需要进行 DAA 调整，使得结果仍为 BCD 码形式。对于排序，我一开始想的是通过编写子程序实现冒泡排序，然后调用子程序完成排序。后来又考虑到只有 3 个数，故直接比较三个数的大小即可，通过不断的交换三个数，使得其呈从大到小的排列顺序(这种做法也可以看作一种排序算法)。

所以通过编写这个程序，我更加清楚的直到了每个字符是以一种什么样的形式存在单元里的，如 ascii、BCD 等，我还知道了一种新的表示两位数的方法，并且知道了这种表示下如何进行运算。还收获了显示输出多位数的思路，即按字符、从高位到低位输出。

（二） 题目二

一、 题目描述

编写一个比赛得分程序。共有 7 个评委，按百分制打分，计分原则是去掉一个最高分和一个最低分，求平均值。要求：

（1）评委的打分以十进制从键盘输入。

（2）成绩以十进制给出，并保留 1 位小数。

（3）输入输出时屏幕上要有相应提示。

二、 编程设计思路

1、 概述

用一个字符串来保存读入的一个评委的打分，并判断该分数是否合法（0-100），如果合法就将该分数加到综合中，并根据需要更新记录最大值的变量与最小值的变量。循环读取 7 次后，将总和减去最大值再减去最小值的结果除以 5，就可以得到商和余数。对商进行输出，对余数乘 5 除以 2 后再输出一位即可。如果不合法则重新输入 7 个评委的打分。

2、 详细设计

（1）字符串的输入

创建一个字符串变量保存读入的用户输入的分数。判断字符串的位数是否大于 3，如果是，就报错然后让用户重新输入。如果不是，就进行下一步（步骤(2)）。

（2）判断输入的字符串是否合法并更新总和、最大值、最小值（子程序）

子程序不需要传参，使用全局变量即可完成程序功能。

对步骤(1)中经过判断的字符串进行转换，将每一位字符的 ascii 码转换为 bcd 码，然后将三个字符组装成一个三位数。具体操作是，定义一个临时变量，用来存储已经组装好的部分，初值为 0。每次按顺序取字符串的一位字符，转换 BCD 码后，进行“临时变量 x10+当前位数字”，循环，直到组装完成。

对组装完成的数再次进行合法性判断。判断其是否>100，如果是则说明输入有误，则提示错误信息，让用户重新输入。如果不是，则说明输入正确。

输入正确后，将该数加到总和中，更新总和。判断其于最大值、最小值变量的数的大小(初值为 0 和 100)，并根据需要更新最大值与最小值。

上述操作完成后，返回步骤(1)，进行输入第二个评委的分数。循环七次，完成数的读取与变量的更新。

（3）计算平均分并输出(带一位小数)

将总和减去最大值和最小值，再除以 5，就得到了平均分。根据汇编语言 DIV

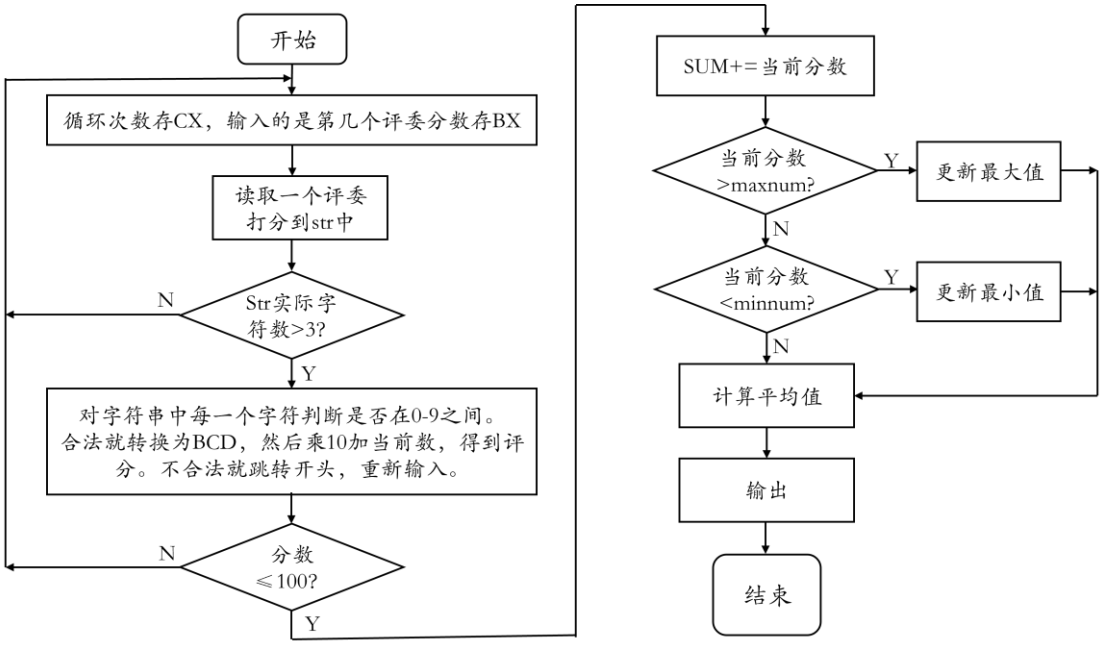
指令的特性，商和余数会分别存放在 AX 和 DX 寄存器当中，分别为 16 位。故，对商和余数分别操作并显示输出。

首先对商进行操作。获取其每一位字符，即每次对商除以 10 取余数，并将余数转 ascii 码后入栈，直到商为 0 为止，记录入栈次数。然后按照次数进行循环出栈，每获取一个字符就对其显示输出，这样，平均数整数部分的输出就完成了。

然后是会小数部分的输出。先输出一个字符'.'，然后对小数部分按照类似与整数部分的操作处理：乘 10 除以 5，即相当于乘 2，即左移一位。对左移一位的数转 ascii 码后输出，就是平均数的小数部分了。

至此，平均数的计算与显示输出工作就完成了。

三、 程序流程图



四、 程序源代码及注解

```
;默认采用ML6.11汇编程序
DATAS SEGMENT
    ;此处输入数据段代码
    ;提示信息
    STR1 DB 0AH,0DH,'PLEASE ENTER THE SCORE ONE BY ONE
(0~100)',0AH,0DH,'$'
    STR2 DB 'SCORE GIVEN BY NO.$'
    STR3 DB 0AH,0DH,'FINAL SCORE',0AH,0DH,'$'
    STR4 DB 'SCORE SUM $'
    WARNN DB 'NUMBER ILLEGAL, PLEASE INPUT ALL THE NUMBER
AGAIN',0AH,0DH,'$'
    ;换行
    NEWLINE DB 13,10,'$'
```

```
    ;存储输入的分数
    SCORE_EACH DB 30H
               DB 0
               DB 30H DUP('$')
    ;临时变量
    SCORE_INT DW 0
    ;总和
    SCORE_SUM DW 0
    ;最小值
    SCORE_MIN DW 100
    ;最大值
    SCORE_MAX DW 0
DATAS ENDS

STACKS SEGMENT
    ;此处输入堆栈段代码
    DW 40H DUP(0)
    TOP LABEL WORD;指向栈顶
STACKS ENDS

CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS

START:
    MOV AX,DATAS
    MOV DS,AX
    MOV ES,AX
    ;此处输入代码段代码

ERR_AGA:          ;输入错误时返回
    ;重置最大值、最小值、总和
    MOV AX,0
    MOV SCORE_SUM,AX
    MOV SCORE_MAX,AX
    MOV AX,100
    MOV SCORE_MIN,AX

    ;'PLEASE ENTER THE SCORE ONE BY ONE (0~100)'   STR1
    LEA DX, STR1
    MOV AH, 09H
    INT 21H
    MOV CX, 07H  ;循环计数
    MOV BX, 01H  ;评委计数
```

```
INN_7:          ;循环读7个分数
    ;清空tmp得分
    MOV SCORE_INT, 0
    ;'SCORE GIVEN BY NO.$'
    LEA DX, STR2
    MOV AH, 09H
    INT 21H
    ; 1---7 显示评委编号
    MOV DX, BX
    ADD DX,30H
    MOV AH, 02H
    INT 21H
    MOV DX,':'
    MOV AH,02H
    INT 21H
    MOV DX,' '
    MOV AH,02H
    INT 21H
    INC BX

    ; 输入第i个评委的打分
    LEA DX, SCORE_EACH
    MOV AH,0AH
    INT 21H
    ;换行
    LEA DX, NEWLINE
    MOV AH,09H
    INT 21H
    ;判断合法与否，不合法重新输入,合法就存加和
    CALL LEGAL_JUDGE

    LOOP INN_7   ;循环


RESULTT:  ;计算平均数并显示
    ;'SCORE SUM'提示信息
    LEA DX,STR4
    MOV AH,09H
    INT 21H
    MOV DX,':'
    MOV AH,02H
    INT 21H
    MOV DX,' '
    MOV AH,02H
```

```
    INT 21H
    ;计算平均数
    ;清空DX
    XOR DX,DX
    ;计算、生成被除数
    MOV AX,SCORE_SUM
    SUB AX,SCORE_MAX
    SUB AX,SCORE_MIN
    ;生成除数
    MOV BX,5
    ;进行除法
    DIV BX
    ;保护余数
    PUSH DX
    ;清空BX，生成除数10，用于获取商的单个字符
    XOR BX,BX
    MOV BX,10
    XOR CL,CL
INT_TST:
    ;每次商除以10取余数，转换ascii码并入栈，同时记录次数
    XOR DX,DX
    DIV BX
    ADD DL,30H
    PUSH DX
    INC CL
    CMP AX,0
    JNE INT_TST  ;AX不为0就继续循环

SHOWW:
    ;根据记录的次数循环输出每一位字符（平均数整数部分），字符从栈顶获取
    POP DX
    MOV AH,02H
    INT 21H
    LOOP SHOWW
    ;计算并显示小数部分
    ;显示一个小数点
    MOV DX,'.'
    INT 21H
    ;获得之前除法的余数，对其乘以10再除以2，相当于乘以2，即左移一位
    POP DX
    SAL DX,1
    ;转换ascii码
    ADD DX,30H
    ;输出该字符
```

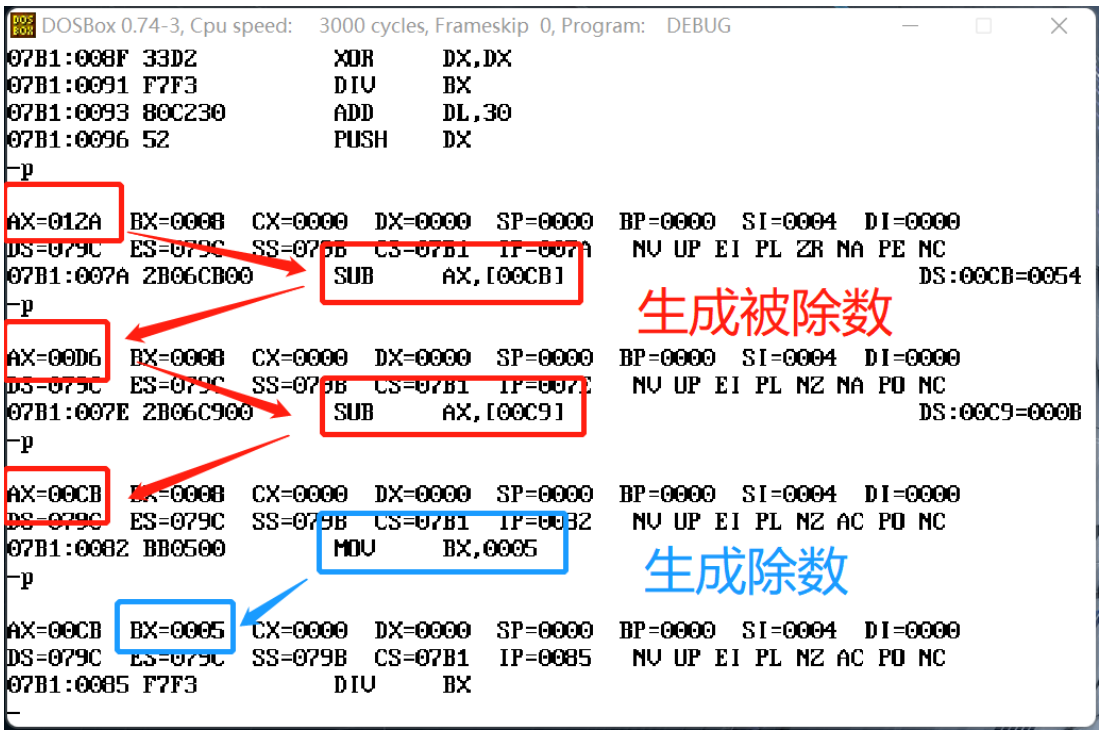
```
    INT 21H
    JMP EXITT

LEGAL_JUDGE PROC    ;判断输入是否合法，合法就组装成三位数，然后更新
SCORE_SUM，SCORE_MIN，SCORE_MAX
    PUSH CX   ;保护CX
    ;判断分数是否为3位及以下
    MOV AL,SCORE_EACH+1
    CMP AL,03H
    JA ERR_IN ;不合法，重新输入
    ;字符串首个元素位置，即SI需要偏移2
    MOV SI,2
    ;获得字符串实际长度
    MOV CL,SCORE_EACH+1
NEXT_C:

    XOR AX,AX ;清空AX
    MOV AL,SCORE_EACH[SI]  ;当前元素送AX
    ;判断是否在0-9之间
    CMP AL,'0'
    JB ERR_IN
    CMP AL,'9'
    JA ERR_IN
    ;如果输入没错,则将str转为int，即组装成3位数
    ;转bcd
    SUB AL,30H
    PUSH AX
    XOR AX,AX
    MOV AX,10
    MUL SCORE_INT
    MOV SCORE_INT,AX
    POP AX
    ADD SCORE_INT,AX
    INC SI
    LOOP NEXT_C  ;循环判断&组装
P_SUM:
    ;清空DX，AX
    XOR DX,DX
    XOR AX,AX
    ;判断分数是否小于等于100，是就继续，否则重新输入
    MOV AX,SCORE_INT
    CMP AX,100
    JA ERR_IN
    ;更新SCORE_NUM
```

```
    ADD SCORE_SUM,AX
    ;判断是否需要更新最小值，是则更新
    CMP AX,SCORE_MIN
    JA UPDATE_SCMAX
    MOV SCORE_MIN,AX
    ;判断是否需要更新最大值，是则更新
UPDATE_SCMAX:
    CMP AX,SCORE_MAX
    JB OVERR
    MOV SCORE_MAX,AX
OVERR:
    POP CX;进入子程序时保护，退出时需要还原
    RET;结束
LEGAL_JUDGE ENDP


ERR_IN:   ;输出错误信息，跳转到ERR_AGA
    LEA DX,WARNN
    MOV AH,09H
    INT 21H

    JMP ERR_AGA

EXITT:
    ;换行
    LEA DX,NEWLINE
    MOV AH,09H
    INT 21H

    JMP ERR_AGA  ;使程序循环运行

    MOV AH,4CH
    INT 21H


CODES ENDS
       END START
```

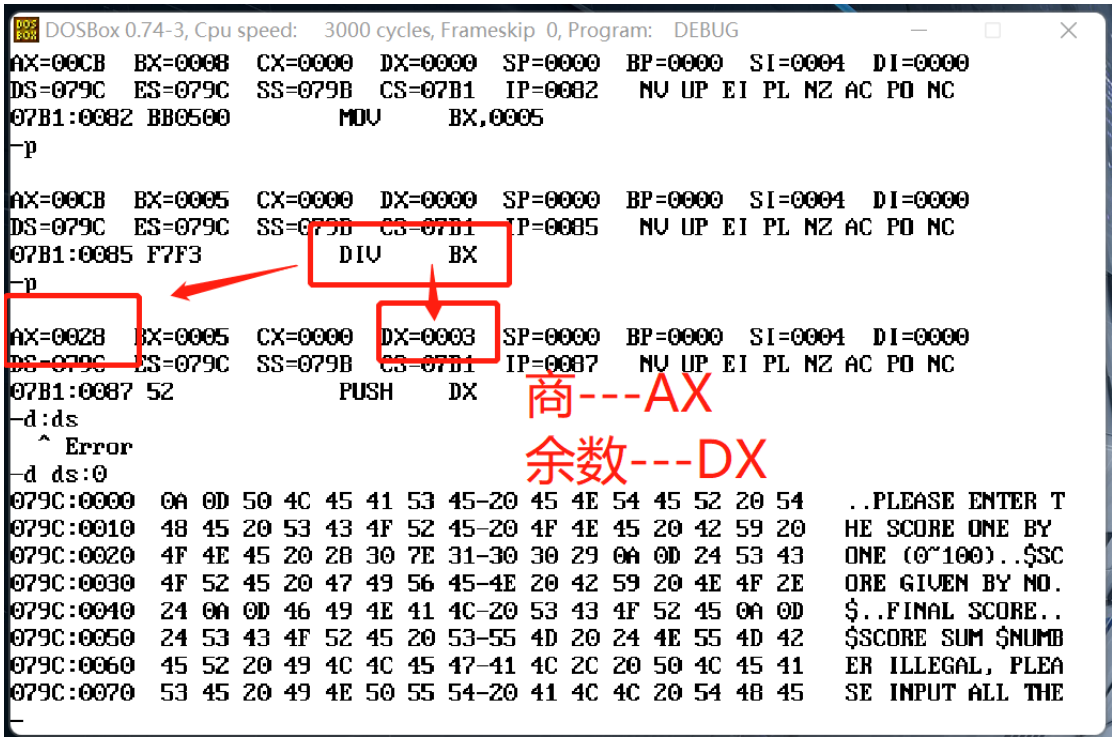
五、 核心模块调试

1、计算平均值

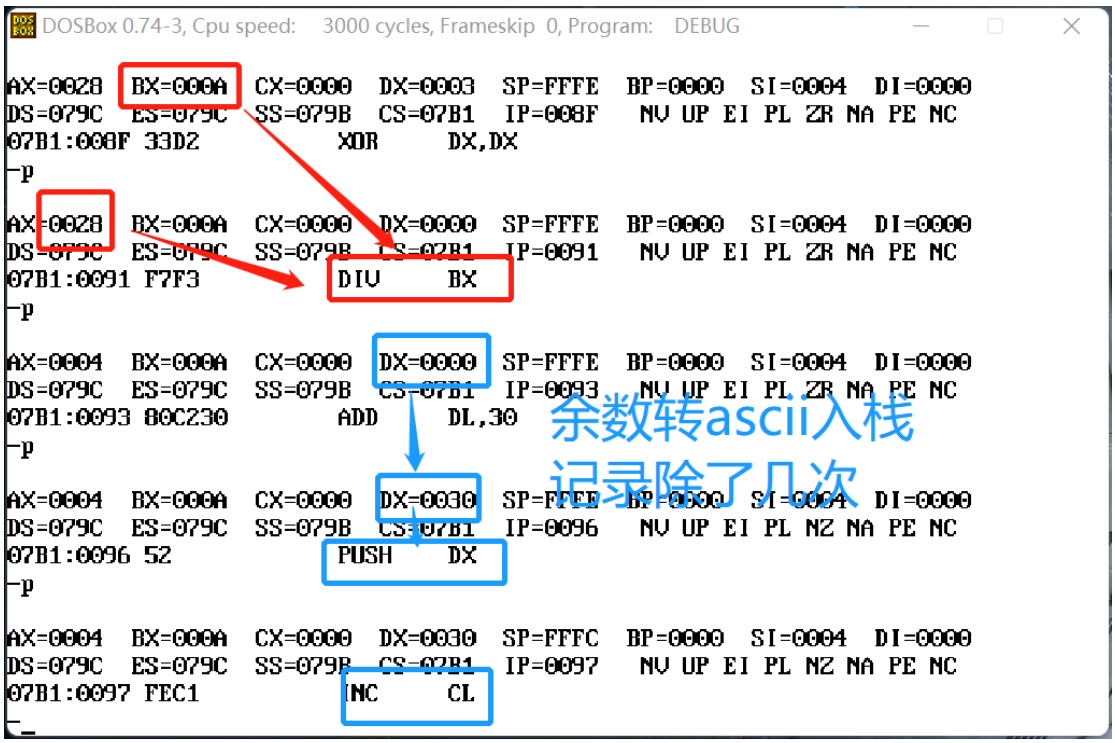
（1） 被除数=总和-最大值-最小值；除数=5

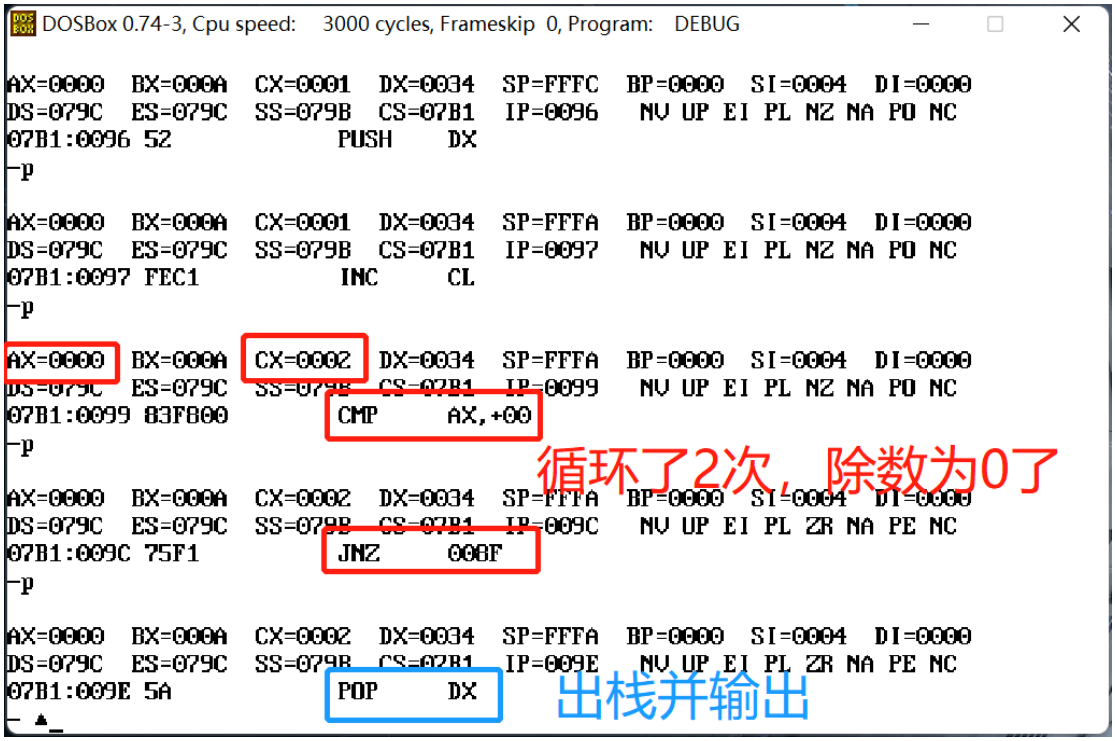


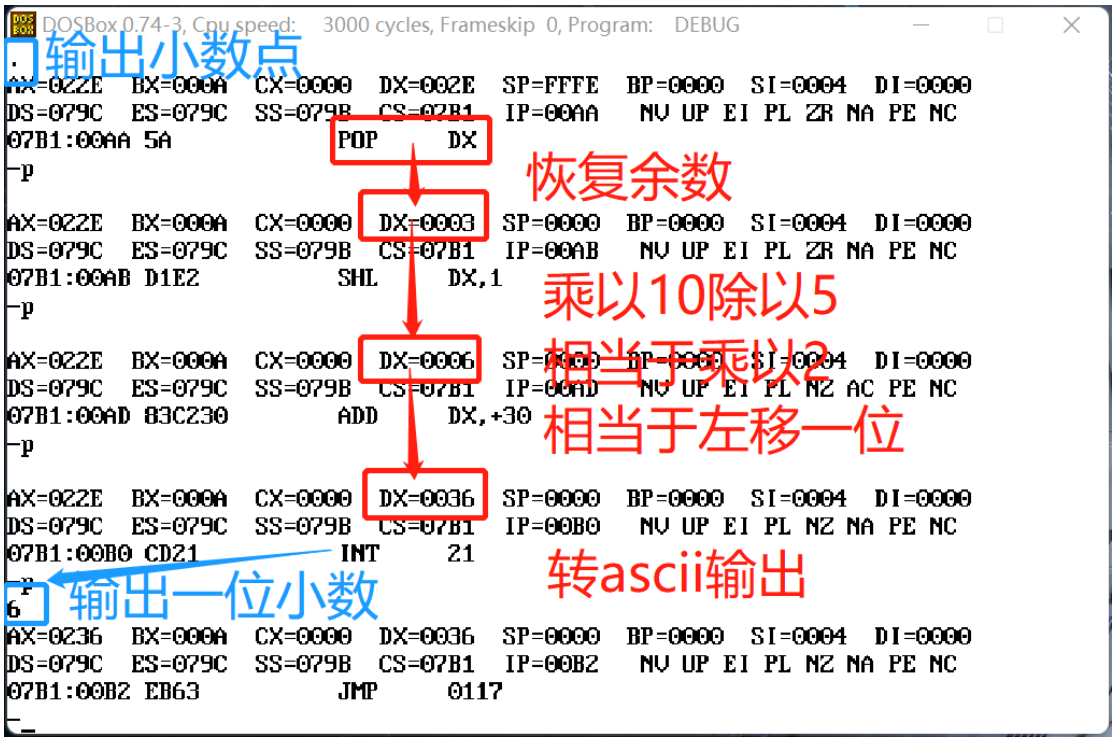
（2） 进行除法，根据 DIV 指令规则，商存在 AX 中，余数存在 DX 中。



（3）每次对商除以 10 取余数转 ascii 码入栈，直到商为 0，就获得了商的每一位数字，记录直到 0 时候的循环次数。再根据循环次数出栈并输出字符即可。对余数左移一位后转 ascii 码输出即可。







六、 程序设计收获与体会

本程序主要实现的是如何从一组数中找到最大值、最小值并计算去掉一个最大值和一个最小值后的平均值。另外需要注意的是边界条件的考虑需要全面。

对于如何找到最小值和最大值，我并没有将一组数排序后得知，而是在用户输入过程中不断的更新最大值、最小值，当输入结束后，就可以简单的获取到最大值和最小值了。我认为这是一种十分高效的方法，不需要额外的时间去处理排序问题。

在计算平均值时，我知道了 DIV 指令的其中一种模式是将商和余数分别存放在两个寄存器中，这使得我对后面的商和余数的处理更加的容易。

对于边界条件的考虑，我知道了判断一个三位数是否合法时，不仅应该判断其位数、每个字符是否合法，还需要判断其表示的三位数是否超过了 100，如果超过就是不合法。这种情况是非常容易被忽视的，通过这个程序的编写，让我记忆的更加深刻了。