**汇编语言程序设计**

**期末报告**

**学 号\_\_\_\_\_\_\_\_\_\_\_20074221\_\_\_\_\_\_\_\_\_**

**姓 名\_\_\_\_\_\_\_\_\_\_\_游佳慧\_\_\_\_\_\_\_\_\_\_\_**

**指导教师\_\_\_\_\_\_\_\_\_\_\_魏坚华\_\_\_\_\_\_\_\_\_\_\_**

**提交日期\_\_\_\_\_\_\_\_\_\_2022.5.31\_\_\_\_\_\_\_\_\_**

**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

[1 题目要求 3](#_Toc104928324)

[2 设计思路 3](#_Toc104928325)

[2.1 综合题1设计思路 3](#_Toc104928326)

[2.2 综合题2设计思路 4](#_Toc104928327)

[3 程序流程图 6](#_Toc104928328)

[4 源代码 7](#_Toc104928329)

[4.1 综合题1源代码 7](#_Toc104928330)

[4.2 综合题2源代码 12](#_Toc104928331)

[5 模块调试 17](#_Toc104928332)

[5.1 综合题1模块调试 17](#_Toc104928333)

[5.1.1 INPUT\_DEC模块 17](#_Toc104928334)

[5.1.2 TRANSITION模块 18](#_Toc104928335)

[5.1.3 NUM\_SUM模块 19](#_Toc104928336)

[5.1.4 PRINT\_SORT模块 20](#_Toc104928337)

[5.2 综合题2模块调试 21](#_Toc104928338)

[5.2.1 INPUT模块 21](#_Toc104928339)

[5.2.2 SORT模块 22](#_Toc104928340)

[5.2.3 PRINT\_AVERAGE模块 23](#_Toc104928341)

[6 总结与收获 24](#_Toc104928342)

# 题目要求

**综合题1：**从键盘输入一个6位的十进制整数，从右到左两位一截一组，计算输出3组之和，并3组数据按从大到小排序输出。

**综合题2：**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）输人输出时屏幕上要有相应提示。

# 设计思路

## 综合题1设计思路

程序开始时输入一个6位十进制整数，首先检查输入数据的正确性，依次对长度、特殊字符和数字进行判断，若输入数据非法，则直接跳转至结束，重置数据段并重新开始程序；若输入合法，下一步是从右往左将6位十进制整数转换为3个2位十进制整数，而后依次将三个数相加计算总和，输出时采用除10取余法得到各个位上的数值并存入数组逐位输出，最后采用冒泡排序法对3个数据进行非递增（由大到小）排序，并以空格为间隔输出排序后的三个数。每次输入输出结束后返回程序开始，实现循环输入输出。

本次共设计了4个子程序，具体实现思路如下：

1. 检测输入正确性

输入的数据存在字符串缓冲区中，缓冲区内第二个字节即为实际输入字符数，若用CX做循环计数则类型不匹配，故用CL存放字符串长度，并先与6进行比较，若不相等，则跳转至ERROR；若相等，则依次比较各个字符，若每个字符均在0-9之间，则跳转至子程序出口，否则跳转至ERROR，并返回主程序。在ERROR分支中将输出对应提示语并将CX置为0FFH便于主程序识别。

2. 6位十进制数转3个2位十进制数

进入循环，从字符串末位开始，取出的第一个字符对应十进制数的十位，第二个字符对应个位，由于字符在机器中都是以ASCII码的二进制格式存储，每次将字符的ASCII码减30H即可得到对应的十进制数值，而后将十位对应数字乘10并加上个位，将转换后的2位十进制数依次存放在数组中，继续循环，直到三个数转换完成并存入数组结束。

3. 计算并输出总和

将AX置零，用于存放总和。进入循环，每次令DX置零并取数组中的一个数存放在DL中，DX与AX做加法运算并将结果保存在AX中，依次加上三个数后循环结束。

输出时采用除十取余法得到各个位上的数，由于AX是十六位，字节型的10是4位，若采用字节除法会导致寄存器溢出，故采用字除法，AX作为被除数，TEN\_DW（定义的常量，值为16位的10）作为除数，商存放在AX中，余数存放在DX中。循环进行除法直至AX为0跳转至输出，每次循环将DL中的一位十进制数加上30H转换为ASCII码，存入数组中。

数组中储存结果逆序，故输出时从末位往前依次输出各字符，得到即为正序。

4. 冒泡排序

冒泡排序基本思想是：重复地走访过要排序的元素列，依次比较相邻的两个数，若前者更小，则交换这两个数，重复该过程直至排序完成。对于数组中的三个数，冒泡排序外层循环次数为2，用CX记录外层循环次数，DI记录内层循环次数，进入循环，AL和DL寄存器中依次存放当前指针所指向数组元素和相邻元素，比较AL和DL的大小，若前者更小则先交换二者，而后数组的地址指针加一，指向下一个元素，内层循环次数DI加一，当DI小于CX时继续内层循环，否则DI清零，跳转至下一次外层循环。

排序结束后数组中即由大到小存放三组数据，输出时同样采用除十取余法，本次采用字节除法，商存放在AL中，余数存放在AH中。与输出总和时的其他不同之处在于，每次取数组中一个数据存放在AL中，进行一次除法后需判断商是否为0，若为0则说明该数为一位十进制数，直接跳转输出个位（存在AH中的余数），否则先输出十位（存在AL中的商），再输出个位，而后输出一个空格作为两数间隔，继续下一次循环除法直至三个数均输出。

## 2.2 综合题2设计思路

程序开始时进入循环，每次输入1个0-100间的十进制整数，并检查每个输入数据的正确性，依次对长度、特殊字符、数字和大小进行判断，若输入数据非法，则直接跳转至结束，重置数据段并重新开始程序；若输入合法，则跳转至下一次循环，输入7个数循环结束，注意每次进入循环前应先将计数器CX压入堆栈保存，子程序调用结束后再恢复CX。下一步是采用冒泡排序法对7个数据进行非递增（由大到小）排序，计算除最大数和最小数之外五个数的总和及平均值，并保留一位小数，输出时采用除10取余法得到各个位上的数值并存入数组逐位输出。每次输入输出结束后返回程序开始，实现循环输入输出。

本次共设计了3个子程序，具体实现思路如下：

1. 检测输入正确性

与题1类似，先将输入字符串长度存入CL中，并与3进行比较，若大于则直接跳转至ERROR；否则依次比较各个字符，输入字符合法需满足以下两点：

1.字符在0-9之间

2.若字符为2-9之间的数，CL需不等于3（该数字不位于百位）

若满足以上两点则将其转换成十进制数并判断是否大于100，若不大于则存入数组，跳转至子程序出口，否则跳转至ERROR，并返回主程序。在ERROR分支中将输出对应提示语并将CX置为0FFH便于主程序识别。

2. 冒泡排序

采用冒泡排序法对存放于NUM\_Dec数组中的7个数进行排序，外层循环次数为6，其余与题1中算法类似，此处不再赘述。

3. 计算平均值并输出

经过排序后此时7个数已按由大到小的次序存放在NUM\_Dec数组中，因题目要求去掉最大值和最小值，只需取数组下标为1-5的五个元素相加并将和存放在AX寄存器中。

接着计算保留1位小数的平均数。将CX置为5，并作为除数，被除数为AX，进行一次除法运算后存放在AL中的商即为平均数的整数部分，将存放在DL中的余数进行左移一位运算，结果即为平均数的小数部分（左移一位相当于数值乘以2）。

输出平均数整数部分时与题1中算法类似，此处作简要描述：循环除十取余直至AX为0，每次将余数转换为ASCII码后存入数组，输出时按存入顺序逆序输出。小数部分则直接加上30H转换为ASCII码后输出。

# 程序流程图

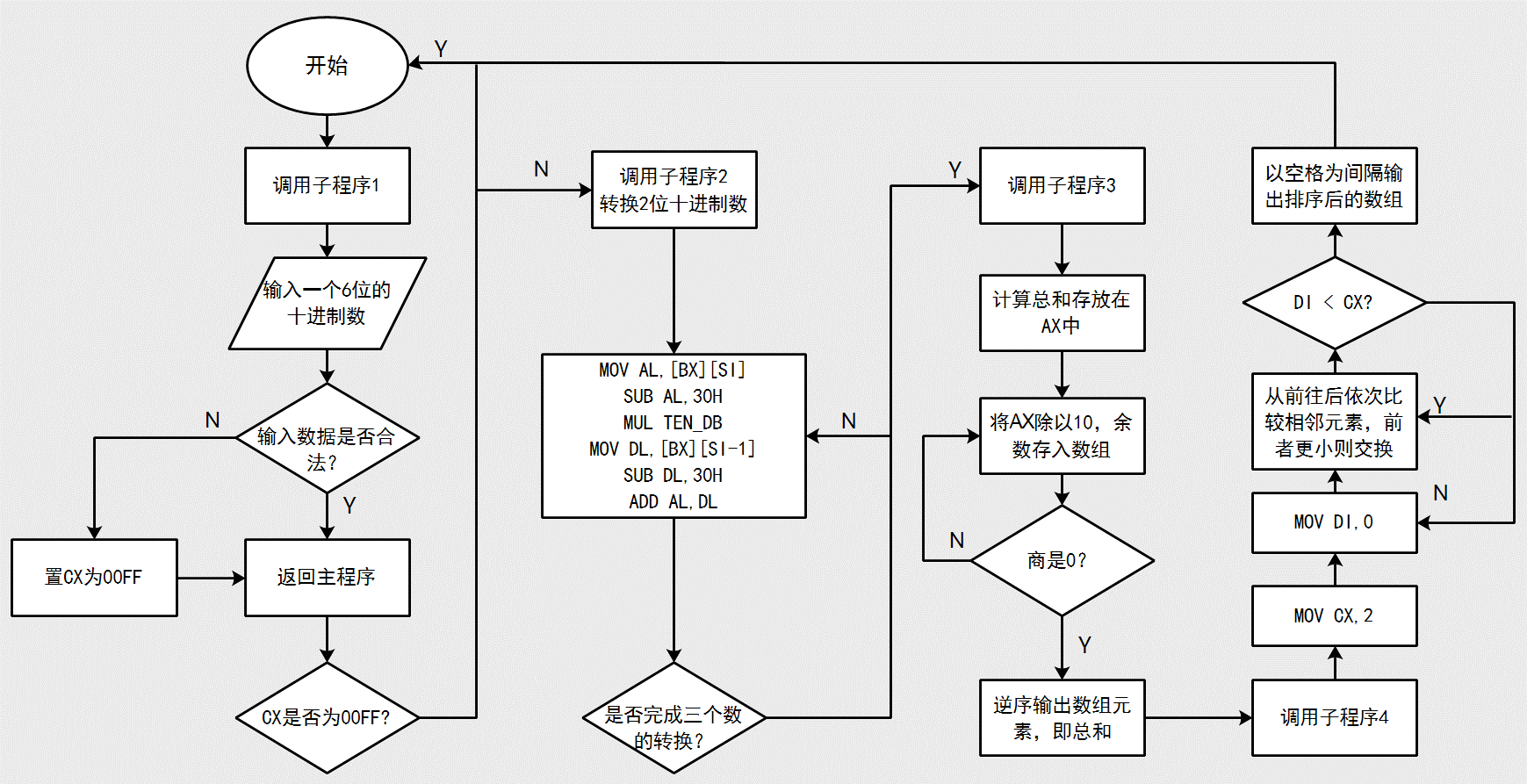


图 1：综合题1流程图

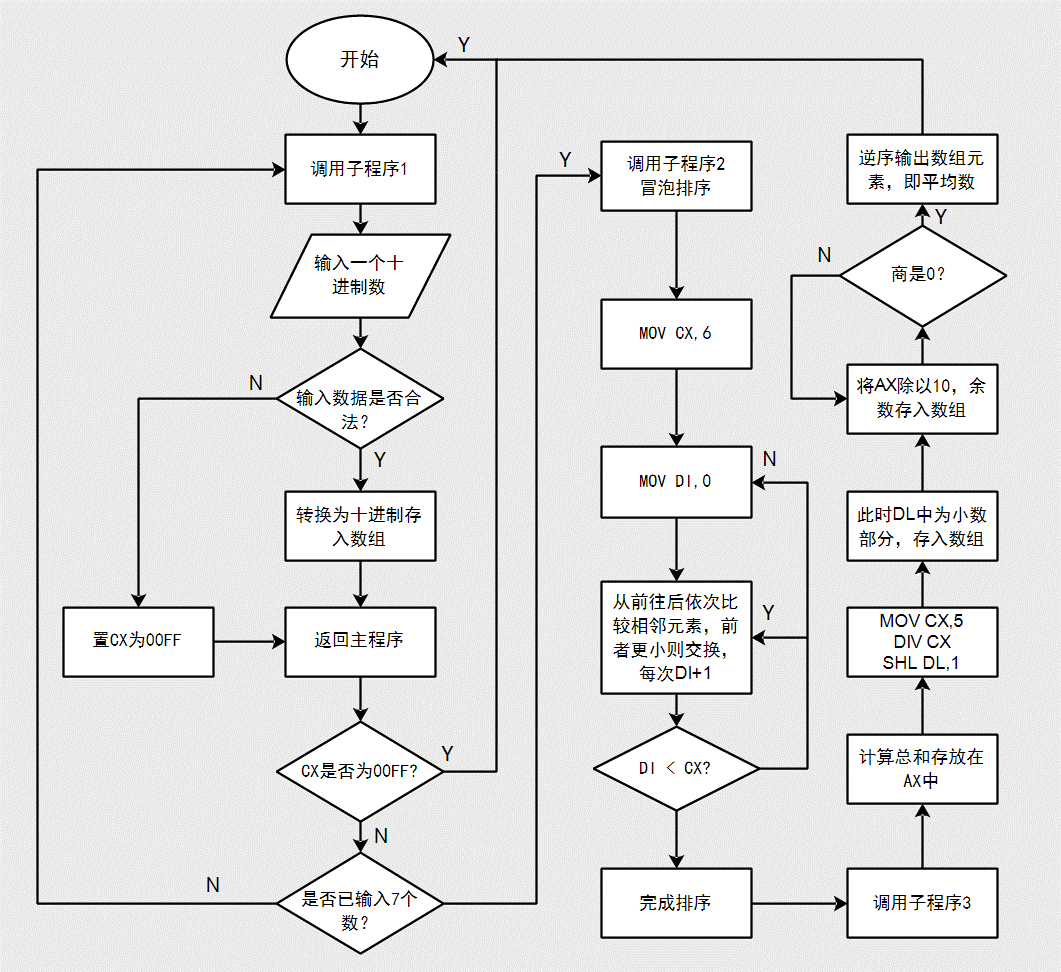


图 2：综合题2流程图

# 源代码

## 4.1 综合题1源代码

|  |
| --- |
| DATAS SEGMENT  ;此处输入数据段代码  TIPS1 DB 'Enter Six Number:',13,10,'$'  TIPS2 DB 'Error Input! Try again!',13,10,'$'  TIPS3 DB 'After Sorting:',13,10,'$'  TIPS4 DB 'The Sum is: ',13,10,'$'  NUM DB 10 ;缓冲区长度  DB ? ;实际输入个数  DB 10 DUP('$') ;存储空间  NUM\_Dec DB 10 DUP('$') ;存放3组数据  SUM DB 8 DUP(' ') ;数据和  TEN\_DB DB 10 ;8位10  TEN\_DW DW 10 ;16位10  LINE DB 0AH,0DH,'$' ;换行  DATAS ENDS  STACKS SEGMENT  ;此处输入堆栈段代码  DW 20H DUP(0)  STACKS ENDS  CODES SEGMENT  ASSUME CS:CODES,DS:DATAS,SS:STACKS  START:  MOV AX,DATAS  MOV DS,AX  ;此处输入代码段代码  LEA DX,TIPS1 ;输出提示语1  MOV AH,9  INT 21H    LEA BX,NUM ;BX指向输入的字符串  CALL INPUT\_DEC ;输入并检查合法性  CMP CX,0FFH ;若CX被置为0FFH则不合法  JZ OVER  CALL PRINT\_LINE ;换行    LEA BX,NUM+2 ;BX指向字符串起始地址  CALL TRANSITION ;将6位十进制数转换为3个2位十进制数    LEA BX,NUM\_Dec ;BX指向存放二位十进制数的数组  CALL NUM\_SUM ;计算总和并输出    CALL PRINT\_SORT ;排序并输出结果    JMP OVER  ;主程序结束  ;输入6位十进制数并检查合法性  INPUT\_DEC PROC  MOV DX,BX ;输入十进制数  MOV AH,10  INT 21H  MOV SI,1  MOV CL,[BX][SI] ;寄存器相对寻址  CMP CL,6 ;若长度不为6则非法  JNE ERROR  CHECK:  INC SI ;字符串从偏移量为2的地方开始  MOV AL,[BX][SI] ;AL依次存每个数  CMP AL,30H ;若小于0则非法  JB ERROR  CMP AL,39H ;若大于9则非法  JA ERROR  LOOP CHECK  JMP EXIT ;比较结束，合法，跳转至出口  ERROR:  MOV CX,0FFH ;置CX为0FFH  LEA DX,TIPS2 ;输出提示语2  MOV AH,9  INT 21H  EXIT: RET ;返回  INPUT\_DEC ENDP  ;将6位十进制数转换为3个2位十进制数  TRANSITION PROC  MOV CX,3 ;三个数，循环三次  MOV SI,5 ;从末位开始  MOV DI,0 ;NUM\_Dec数组的偏移量  XOR AX,AX ;清零  XOR DX,DX  TRAN:  MOV AL,[BX][SI] ;取十位  SUB AL,30H ;转为数字  MUL TEN\_DB ;乘10  MOV DL,[BX][SI-1] ;取个位  SUB DL,30H ;转为数字  ADD AL,DL ;十位加上个位  MOV NUM\_Dec[DI],AL ;转换后的二位十进制数存入数组  INC DI  SUB SI,2 ;偏移量每次减2  LOOP TRAN  RET ;返回  TRANSITION ENDP  ;计算总和并输出  NUM\_SUM PROC  LEA DX,TIPS4 ;输出提示语4  MOV AH,9  INT 21H    XOR AX,AX ;清零  MOV SI,0 ;SI为BX偏移量  PLUS:  XOR DX,DX ;清零  MOV DL,[BX][SI] ;取一个数存在DL中  ADD AX,DX ;AX依次加上三个数  INC SI  CMP SI,3 ;没加到3个数则继续循环  JNZ PLUS    MOV SI,0  TURN\_SUM:  XOR DX,DX ;清零  DIV TEN\_DW ;除10取余得到各个位上的数值，余数存放在DL中  ADD DL,30H ;余数转换为ASCII  MOV SUM[SI],DL ;存入数组  INC SI  CMP AX,0 ;商为0则算法结束  JA TURN\_SUM  PRINT\_SUM:  DEC SI ;SI为数组中元素个数  MOV DL,SUM[SI] ;按存放逆序输出即为正序  MOV AH,2 ;依次输出各个数位上的数值  INT 21H  CMP SI,0 ;SI为0说明最后一个数已输出  JNZ PRINT\_SUM  CALL PRINT\_LINE ;换行  RET ;返回  NUM\_SUM ENDP  PRINT\_SORT PROC  ;冒泡排序  MOV CX,2 ;三个数，外层循环执行2次  SORT1:  MOV SI,0 ;清零  MOV DI,0  SORT2:  MOV AL,[BX][SI] ;取第一个元素  MOV DL,[BX][SI+1] ;取第二个元素  CMP AL,DL ;比较第一个元素和第二个元素的大小，若前者更大则跳转  JNB SORT3  MOV [BX][SI],DL ;若前者更小，则交换  MOV [BX][SI+1],AL  SORT3:  INC SI ;偏移量加1  INC DI ;计数加1  CMP DI,CX ;DI记录内层循环个数，若DI<CX,继续比较后面的元素  JB SORT2  LOOP SORT1 ;内层循环结束，开始下一次外层循环    ;输出排序后的数  LEA DX,TIPS3 ;输出提示语3  MOV AH,9  INT 21H  MOV CX,3 ;循环三次  MOV SI,0 ;SI为BX偏移量  OUTPUT:  XOR AX,AX ;清零  XOR DX,DX  MOV AL,[BX][SI] ;取一个数存在AL中  DIV TEN\_DB ;除以10,字节除法余数在AH中  MOV DL,AL ;商在AL中  CMP DL,0 ;若商为0说明十位为0，跳转输出个位  JZ OUTPUT2  ADD DL,30H ;商不为0，转换为ASCII输出十位  PUSH AX ;保存，因为下面会使AH值改变  MOV AH,2  INT 21H  POP AX ;恢复AX  OUTPUT2:  MOV DL,AH ;将余数存在DL中  ADD DL,30H ;转换为ASCII输出十位  MOV AH,2  INT 21H    MOV DX,20H ;输出一个空格  MOV AH,2  INT 21H    INC SI  LOOP OUTPUT ;继续输出下一个数    CALL PRINT\_LINE ;换行  RET  PRINT\_SORT ENDP  PRINT\_LINE PROC  LEA DX,LINE ;换行  MOV AH,9  INT 21H  RET  PRINT\_LINE ENDP    OVER:  CALL PRINT\_LINE ;换行  JMP START ;重新开始    MOV AH,4CH  INT 21H  CODES ENDS  END START |

## 4.2 综合题2源代码

|  |
| --- |
| DATAS SEGMENT  ;此处输入数据段代码  TIPS1 DB 'Enter Seven Number Separated By Space:',13,10,'$'  TIPS2 DB 'Error Input! Try again!',13,10,'$'  TIPS3 DB 'The Average Score is: $'  NUM DB 255 ;缓冲区长度  DB ? ;实际输入个数  DB 255 DUP('$') ;存储空间  NUM\_Dec DB 10 DUP(?) ;存放7组数据  AVERAGE DB 10 DUP(?) ;存放平均值  NUM\_TEMP DB 10 DUP(?) ;临时数组  TEN\_DB DB 10 ;8位10  TEN\_DW DW 10 ;16位10  HUN\_DB DB 100 ;8位100  LINE DB 0AH,0DH,'$' ;换行  DATAS ENDS  STACKS SEGMENT  DW 20H DUP(0)  STACKS ENDS  CODES SEGMENT  ASSUME CS:CODES,DS:DATAS,SS:STACKS  START:  MOV AX,DATAS  MOV DS,AX    LEA DX,TIPS1 ;输出提示语1  MOV AH,9  INT 21H    MOV CX,7 ;循环7次  MOV DI,0 ;DI为存放十进制数数组的偏移量  INPUT\_7:  LEA BX,NUM ;BX指向输入的字符串  PUSH CX ;保存CX，子程序中CX会被更改  CALL INPUT ;输入并检查合法性  CMP CX,0FFH ;若CX被置为0FFH则不合法  JZ OVER  POP CX ;恢复CX  CALL PRINT\_LINE ;换行  LOOP INPUT\_7 ;循环输入7个数    LEA BX,NUM\_Dec ;BX指向储存7个数的NUM\_Dec数组  CALL SORT ;排序    CALL PRINT\_AVERAGE ;计算并打印平均值  JMP OVER  INPUT PROC  MOV DX,BX ;输入一个数  MOV AH,10  INT 21H  XOR DX,DX ;清零  MOV SI,1  MOV CL,[BX][SI] ;若长度大于3则非法  CMP CL,3  JA ERROR  CHECK:  XOR AX,AX  INC SI ;SI=SI+1  MOV AL,[BX][SI] ;字符串是从偏移量为2的地方开头  ;MOV BYTE PTR [BX][SI],'$' ;将原位置置为‘$’  CMP AL,30H ;若小于0则非法  JB ERROR  CMP AL,39H ;若大于9则非法  JA ERROR  CMP AL,31H ;若为0或1则跳转  JNA TRANS1  CMP CL,3 ;若为2-9之间，则此时若为百位则非法  JE ERROR  TRANS1:  SUB AL,30H ;从ASCII值转换为数字  CMP CL,3 ;若为百位，乘100  JNZ CHECK2  MUL HUN\_DB  JMP CHECK3  CHECK2:  CMP CL,2 ;若为十位，乘10  JNZ CHECK3 ;若为个位，不做处理  MUL TEN\_DB  CHECK3:  ADD DL,AL ;将该数值加入DL中  LOOP CHECK  CHECK\_UPPER:  CMP DL,100 ;比较输入十进制数是否大于100，大于则非法  JA ERROR  MOV NUM\_Dec[DI],DL ;将转换后的十进制数存入数组中  INC DI ;偏移量加1  JMP EXIT    ERROR:  CALL PRINT\_LINE ;换行  LEA DX,TIPS2 ;输出提示语2  MOV AH,9  INT 21H  MOV CX,0FFH ;置CX为0FFH  EXIT: RET ;返回  INPUT ENDP  ;计算并打印平均值  PRINT\_AVERAGE PROC  XOR AX,AX ;AX清零  MOV SI,1 ;排序后的数组取下标为1-5的元素相加  MOV CX,5 ;循环5次  SUM:  XOR DX,DX ;DX清零  MOV DL,[BX][SI] ;取一个数存在DL中  ADD AX,DX ;AX依次加上五个数  INC SI ;偏移量加1  LOOP SUM    XOR DX,DX ;清零  MOV CX,5  DIV CX ;总和除以5  MOV AVERAGE,AL ;AL中为整数部分，存入数组  SHL DL,1 ;余数\*10/5，相当于\*2，即为小数部分  MOV AVERAGE[1],DL ;小数部分存入数组    MOV SI,0 ;SI为临时数组的偏移量  TURN\_AVERAGE:  XOR DX,DX ;清零  DIV TEN\_DW ;除10取余得到各个位上的数值，余数存放在DL中  ADD DL,30H ;余数转换为ASCII  MOV NUM\_TEMP[SI],DL ;整数部分存入临时数组  INC SI  CMP AX,0 ;商为0则算法结束  JA TURN\_AVERAGE    LEA DX,TIPS3 ;输出提示语3  MOV AH,9  INT 21H  PRINT:  DEC SI ;SI为数组中元素个数  MOV DL,NUM\_TEMP[SI] ;按存放逆序输出即为正序  MOV AH,2 ;依次输出各个数位上的数值  INT 21H  CMP SI,0 ;SI为0说明最后一个数已输出  JNZ PRINT    MOV DX,'.' ;输出小数点  MOV AH,2  INT 21H    XOR DX,DX ;DX清零，后续要存放小数部分  MOV DL,AVERAGE[1] ;输出小数部分  ADD DL,30H ;转换为ASCII  MOV AH,2  INT 21H  RET  PRINT\_AVERAGE ENDP  SORT PROC  ;冒泡排序  MOV CX,6 ;七个数，外层循环执行6次  SORT1:  MOV SI,0 ;BX偏移量  MOV DI,0 ;内层循环次数  SORT2:  MOV AL,[BX][SI] ;取第一个元素  MOV DL,[BX][SI+1] ;取第二个元素  CMP AL,DL ;比较第一个元素和第二个元素的大小，若前者更大则跳转  JNB SORT3  MOV [BX][SI],DL ;若前者更小，则交换  MOV [BX][SI+1],AL  SORT3:  INC SI ;偏移量加1  INC DI ;计数加1  CMP DI,CX ;DI记录内层循环个数，若DI<CX,继续比较后面的元素  JB SORT2  LOOP SORT1 ;内层循环结束，开始下一次外层循环  RET  SORT ENDP  PRINT\_LINE PROC  LEA DX,LINE ;换行  MOV AH,9  INT 21H  RET  PRINT\_LINE ENDP    OVER:  LEA BX,NUM ;BX指向NUM字符串  MOV AL,24H ;AL存‘$’  MOV SI,0 ;SI为BX偏移量  MOV CX,255 ;循环次数为缓冲区长度  CLEAR\_NUM: ;重置NUM  MOV [BX][SI],AL ;全部重置为‘$’  INC SI  LOOP CLEAR\_NUM    MOV SI,0 ;SI为偏移量，三个数组长度相等  MOV CX,10 ;循环十次  CLEAR\_ALL: ;重置三个数组  MOV NUM\_Dec[SI],AL  MOV AVERAGE[SI],AL  MOV NUM\_TEMP[SI],AL  INC SI  LOOP CLEAR\_ALL  CALL PRINT\_LINE ;换行  JMP START ;重新开始    MOV AH,4CH  INT 21H  CODES ENDS  END START |

# 模块调试

## 5.1 综合题1模块调试

### 5.1.1 INPUT\_DEC模块

|  |
| --- |
| 图 3：输入数据非法 |
| 说明：将输入字符串长度存于CL中，此时输入字符串为12345，长度不为6则直接跳转，将CX置为00FF便于主程序识别错误。 |

|  |
| --- |
| **图 4：输入数据合法** |
| 说明：将输入字符串长度存于CL中，此时输入字符串为123456，长度为6，则跳转至依次比较AL中存放字符是否为0-9。 |

### 5.1.2 TRANSITION模块

|  |
| --- |
| **图 5：转换二位十进制数** |
| 说明：图示过程为依次取十位和个位存放在 AL和DL中，经过乘法和加法运算后AL中存放即为转换后的二位十进制数。输入字符串为123456，可以看到图中最后一步AL存放数字为65。 |

### 5.1.3 NUM\_SUM模块

|  |
| --- |
| **图 6：计算总和** |
| 说明：图示过程为依次取数组中一个元素存放在DL中并与AX做加法，循环三次，最终总和存放于AX中。 |

|  |
| --- |
| **图 7：输出总和** |
| 说明：图示过程为对存放总和的AX寄存器循环除十，并将存放在DL中的余数转换为ASCII码后存于数组。 |

### 5.1.4 PRINT\_SORT模块

|  |
| --- |
| **图 8：冒泡排序** |
| 说明：图示为部分冒泡排序过程，AL和DL寄存器中依次存放数组中相邻两个元素，比较二者大小，图中AL和DL中分别为43和21，满足大小关系，故不做交换，跳转至下一次循环。 |

## 5.2 综合题2模块调试

### 5.2.1 INPUT模块

|  |
| --- |
| **图 8：转换十进制** |
| 说明：图示为将输入字符转换为十进制数，依次判断该数位于百位、十位还是个位，转换成正确数值后与DL相加并存入DL。 |

|  |
| --- |
| **图 9：判断正确性** |
| 说明：图示为判断存放于DL中的数值是否大于100，图中DL存放的是输入的数据100，故数据合法，存入数组。 |

### 5.2.2 SORT模块

|  |
| --- |
| **图 10：冒泡排序** |
| 说明：图示为冒泡排序部分过程，若AL与DL寄存器中后者所存放数值更大，则交换，图中AL存放数值为1，DL存放数值为2，故在原数组中交换二者。 |

### 5.2.3 PRINT\_AVERAGE模块

|  |
| --- |
| **图 11：小数处理** |
| 说明：图示为对平均数小数部分的处理。将DX置零，用于存放总和除以五的余数部分，而后将DL进行左移一位操作，可以看到此时DL中存放的4即为平均数的一位小数。 |

|  |
| --- |
| **图 12：小数输出** |
| 说明：图示为输出小数部分。将DL中的数值加上30H转换为ASCII码，可以看到屏幕上输出小数部分数值为4。 |

# 总结与收获

汇编语言是一种面向机器的程序设计语言，不同于大一时所学的C/C++、Java等高级语言，汇编更接近于机器语言，能够直接对硬件进行操作，所以在学习和编程时我们需要去迁就电脑的思维方式。本学期的汇编上机题难度和复杂度循序渐进，我们依次完成了对分支、循环、子程序设计的练习，最后是两道综合题，需要我们综合运用前面所学知识解决。经过这一学期的学习和练习，我也慢慢对于汇编语言有了更深的理解，想要编写汇编程序，需要熟悉汇编指令的功能及格式，了解并学会使用各种运算符，掌握包括直接寻址、寄存器寻址、基址变址寻址等寻址方式，在刚接触汇编语言时，我对这多种寻址方式的概念和使用方法难以理解，在后来自己亲自编写程序的过程中才慢慢理解并掌握了，这也使我体会到，学习一门程序设计语言，不能光停留在课本上，通过实验可以很大程度上加深理解和印象。

通过对汇编语言的学习，我学到和收获了很多，对于操作系统和组成原理方面的知识也有了更加深入的理解，相信对于未来的学习生活都会有所帮助。