

写代码前的构思：

分为三个部分，输入，计算，输出

继续划分可以分为成绩输入，第二问计算、第三问计算、第四问计算，各自输出

输入部分一开始我采用21号中断的1号功能一位一位的输入，并以此来分别对数字、空格、回车和其余字符进行处理：数字计算后遇空格存入开辟的储存单元，空格标志一个数的结束，回车结束，其余字符报错。

计算部分，在输入时将获取到的前一位数乘相应数升为高位与新获取的数本身相加，构成输入的数字储存；第二问的计算在输入时空格标志一个成绩成功输入后与60对比进行计算；第三问编写排序子程序，排序后输出；第四问平均分在输入时将获取到的成绩全部累加到另外一个储存空间，全部输入后再一位一位计算并输出。

enterline macro ;定义回车换行的宏指令

mov dl,13

mov ah,2

int 21h

mov dl,10

mov ah,2

int 21h

endm

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DATAS SEGMENT

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_计算变量\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

fail DB '0'

pass DB '6'

sum DW 0

num DB 6

flag db 0

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_随手用的临时变量\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

AFP DB ?

BFP DB ?

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_储存单元/缓冲区\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

stu DB 50,?,50 dup('$')

SCORE DB 6 dup(0)

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_提示语句\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

stringerript DB 'wrong input $'

string0 DB 'input 6 scores (0-100) : $'

failnum DB 'failed number:$'

passnum DB 'passed number:$'

average DB 'average score:$'

string1 DB 'reverse string is:$'

string2 db 'scores in order:$'

DATAS ENDS

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_栈\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

STACKS SEGMENT

;此处输入堆栈段代码

DW 50 DUP(0)

STACKS ENDS

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_初始化\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CODES SEGMENT

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

START:

MOV AX,DATAS

MOV DS,AX

MOV AX,STACKS

MOV SS,AX

MOV SP,100

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_完成输入\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

XOR CX,CX

XOR AX,AX

XOR DX,DX

call INPUTSTU

;\_\_\_\_\_\_\_\_\_\_输出不/及格人数、平均分\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ENTERLINE

XOR CX,CX

XOR AX,AX

XOR DX,DX

call prtpf

;\_\_\_\_\_\_\_\_\_\_冒泡排序\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ENTERLINE

MOV CX,5

LEA DI,SCORE

LOO1:

CALL BUB

INC DI

LOOP LOO1

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_输出排好序的成绩\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

lea dx, string2

mov ah,9

int 21h

enterline

XOR CX,CX

XOR AX,AX

XOR DX,DX

MOV CX,6

LEA SI,SCORE

JMP PP1

ROF:

INC SI

PP1:

MOV AL,[SI]

MOV AH,0

call CALCU

LOOP ROF

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_结束\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOV AH,4CH

INT 21H

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_以下为子程序\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

inputstu PROC NEAR

stustart:

lea dx, string0;9号功能字符串输入缓冲区

mov ah,9

int 21h

enterline

mov ah,0ah

lea dx,STU

int 21h

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_一些寄存器的初始设置\_\_\_\_\_\_\_\_\_\_\_

xor bx,bx

LEA SI,STU

LEA DI,SCORE

MOV AL,[SI+2]

MOV BX,0

JMP S1

;\_\_\_\_\_\_\_\_\_\_\_输入算法\_\_\_\_\_\_\_\_\_\_\_\_\_\_

S0:

INC SI

MOV AL,[SI+2]

s1:

CMP AL,13

JE S5

cmp al,' '

JE S3

CMP al,48 ;判断是否是数字

jb erriptstu

cmp al,57

ja erriptstu

NXT:

SHL BX,1

mov CX,BX

SHL CX,1

SHL CX,1

add BX,CX

XOR cx,cx;一位一位转换，存入scores储存单元

mov cL,al

sub cL,30H

add BX,CX

jmp S0

S3:

add sum,BX

cmp BL,60

jae S4

inc fail

dec pass

;-----------

S4:

mov [DI],BX

cmp bx,100

ja erriptstu

cmp bx,0

jb erriptstu

add di,1

XOR BX,BX

JMP S0

;-----------

S5:

mov [DI],BX

cmp bx,100

ja erriptstu

cmp bx,0

jb erriptstu

add di,1

ADD SUM,BX

CMP BL,60

JAE S6

INC FAIL

DEC PASS

S6:

XOR BX,BX

jmp endstu

erriptstu:

enterline

lea dx, stringerript

mov ah,9

int 21h

endstu:

ret

inputstu endp

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

prtpf proc near

mov dx,offset failnum

mov ah,9

int 21h

mov dl,fail

mov ah,2

int 21h

enterline

mov dx,offset passnum

mov ah,9

int 21h

mov dl,pass

mov ah,2

int 21h

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ENTERLINE

mov dx,offset average

mov ah,9

int 21h

XOR AX,AX

XOR DX,DX

;-----------

MOV AX,SUM

cmp ax,600

je special

mov cl,6

div cl ;总和除6 此时al商ah余数

mov BFP,al ;b商a余

mov AFP,ah

xor ax,ax

mov cl,10

mov al,BFP

div cl

mov cl,ah

add al,30h

mov dl,al

mov ah,02h

int 21h ;十位

add cl,30h

mov dl,cl

mov ah,02h

int 21h ;个位

mov dl,2eh ;.

mov ah,02h

int 21h

mov cl,10

mov al,AFP

mul cl

mov cl,6

div cl ;al商ah余数

add al,30h

mov dl,al

mov ah,02h

int 21h

jmp nonospecial

special:

mov dl,49

mov ah,2

int 21h

mov dl,48

mov ah,2

int 21h

mov dl,48

mov ah,2

int 21h

mov dl,2eh

mov ah,2

int 21h

mov dl,48

mov ah,2

int 21h

nonospecial:

ret

prtpf endp

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_冒泡排序\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

BUB PROC NEAR

PUSH CX

PUSH BX

MOV CX,5;6-1

FORFOR:;\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\外

MOV BX,0

FORFORFOR:;================================内

MOV AL,SCORE[BX]

CMP AL,SCORE[BX+1]

JLE FORFORFORFOR;be

MOV AL,SCORE[BX]

MOV DL,SCORE[BX+1]

MOV SCORE[BX],DL

MOV SCORE[BX+1],AL

FORFORFORFOR:

INC BX

CMP BX,CX

JL FORFORFOR;===========================

LOOP FORFOR;\\\\\\\\\\\\\\\\\\\\\\\\\\\

POP BX

POP CX

RET

BUB ENDP

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CALCU PROC NEAR

PUSH BX

PUSH CX

MOV BL,100

DIV BL

MOV CH,AH;商al余数ah

MOV DL,AL

ADD DL,30H

CMP DL,0

JE CAL1

MOV AH,2

INT 21H;百位

CAL1:

MOV AL,CH

MOV AH,0

MOV BL,10

DIV BL

MOV CH,AH;十位al个位ah

MOV DL,AL

ADD DL,30H

MOV AH,2

INT 21H;十位

MOV DL,CH

ADD DL,30H

MOV AH,2

INT 21H

MOV DL,' ';空格隔开

MOV AH,2

INT 21H

POP CX

POP BX

RET

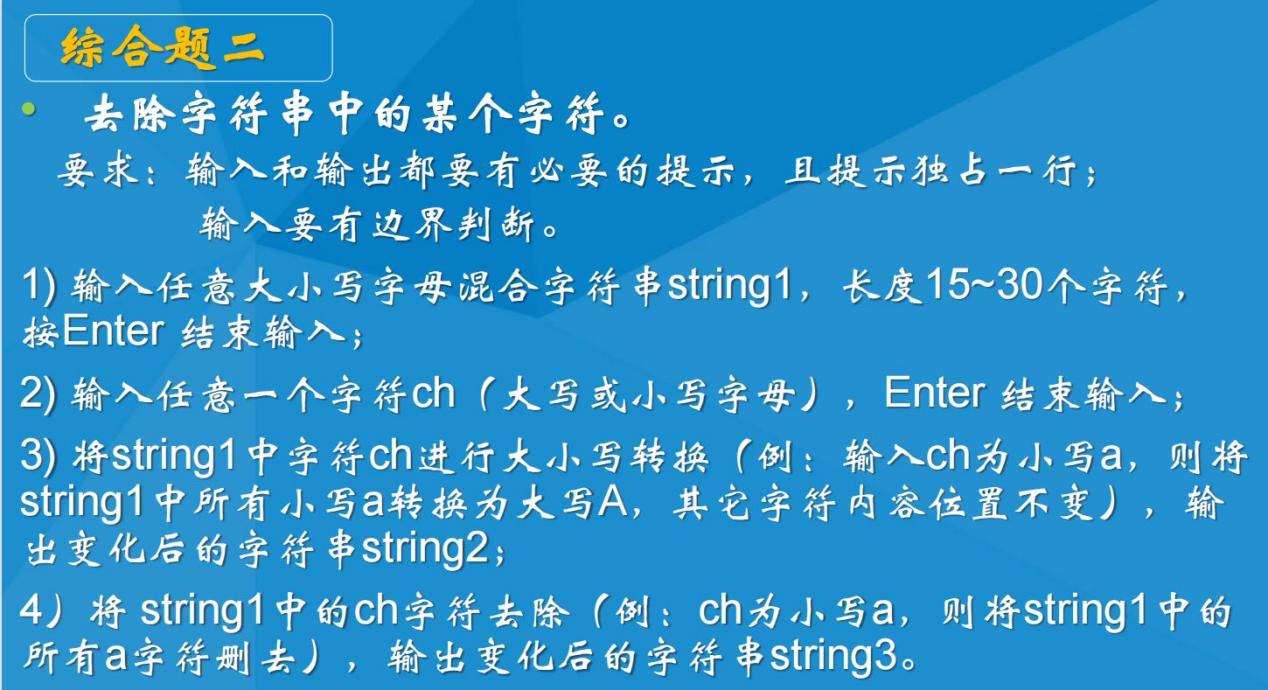
CALCU ENDP

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CODES ENDS

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

END START



思路：

分为三个部分，输入，转换，输出

其中第一第二问在输入部分完成，第三问单独子程序完成，第四问输出中完成

enterline macro ;定义回车换行的宏指令

mov dl,13

mov ah,2

int 21h

mov dl,10

mov ah,2

int 21h

endm

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DATAS SEGMENT

;此处输入数据段代码

STRING0 db "please input a string:$"

STRING2 DB "THE BIGGEST ASCII IS:$"

INPUTSTR DB 50,?,50 dup('$')

;能容纳字符个数;实际接受的字符个数;字符串缓冲区

ERROR DB "ERRORERROR$"

ERRORipt DB "ERRORinput$"

output db 'LENGTH is$:'

input db 'Please Input STR:$'

flag db 0

flag1 db 0

NUM DB 0

NUMIN DB 4,?,4 DUP(0)

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DATAS ENDS

STACKS SEGMENT

;此处输入堆栈段代码

DW 50 DUP(0)

STACKS ENDS

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CODES SEGMENT

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

START:

MOV AX,DATAS

MOV DS,AX

MOV AX,STACKS

MOV SS,AX

MOV SP,100

;此处输入代码段代码

LEA dx,STRING0 ;提示

mov ah,09h

int 21h

enterline

;=========

mov ah,0ah ;输入长字符串

lea dx,INPUTSTR

int 21h

;=========

XOR AX,AX

XOR BX,BX

MOV BX,30 ;边界判断

LEA si,INPUTSTR

MOV AX,[SI+1]

MOV AH,0

CMP AX,BX

JA ERRiptipt

MOV BX,15

CMP AX,BX

JB ERRiptipt

;=========

ENTERLINE

XOR CX,CX

XOR AX,AX

XOR DX,DX

CALL CHANGE;==============输入新的字符以及大小写替换

;=========

ENTERLINE

XOR CX,CX

XOR AX,AX

XOR DX,DX

CALL DELETE

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

JMP STOP

ERR:

enterline

MOV DX,OFFSET ERROR;报错

MOV AH,09H

INT 21H

jmp stop

erriptipt:

enterline

MOV DX,OFFSET ERRORipt;报错

MOV AH,09H

INT 21H

stop:

MOV AH,4CH

INT 21H

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CHANGE PROC NEAR

beginint:

lea dx,input ;给出输入提示

mov ah,9

int 21h

lea dx,numin ;从键盘接收输入数值放入numin缓冲区

mov ah,0AH

int 21h

enterline ;回车换行

PRINT:

mov ax,0

mov bx,0

mov si,0

LEA SI,INPUTSTR

LEA DI,NUMIN

i0: mov cx,0

MOV AL,[SI+2]

CMP AL,'$'

JE NORMAL

CMP AL,[DI+2]

jne I2

cmp AL,65

JAE I1

JMP I2

I1: cmp AL,90

JBE JIA

CMP AL,97

JB I2

CMP AL,122

JA I2

JIAN:

SUB AL,20H

JMP SHUCHU

JIA:

add AL,20H

;输出

SHUCHU:

mov dl,AL

mov ah,2

int 21h

;sub [si+2],bl

JMP I3

I2:

mov dl,[si+2]

mov ah,2

int 21h

I3:

INC SI

JMP I0

E:MOV DX,OFFSET ERROR

MOV AH,09H

INT 21H

jmp beginint ;如出错则返回起始点重新输入

enterline

NORMAL:

RET

CHANGE ENDP

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DELETE PROC NEAR

BEGINDEL:

mov ax,0

mov bx,0

mov si,0

LEA SI,INPUTSTR

LEA DI,NUMIN

D1:

MOV AL,[SI+2]

CMP AL,'$'

JE NORMAL

CMP AL,[DI+2]

jne D2

JMP D3

D2:

mov dl,[si+2]

mov ah,2

int 21h

D3:

INC SI

JMP D1

NORMAL:

RET

DELETE ENDP

;===============================================================

CODES ENDS

END START