汇编语言程序设计
实验报告

东北大学

实验名称	循环结构与子程序设计实验
班　　级	**软信-1503**
学　　号	**20155362**
姓　　名	**薛旗**
日　　期	**2016.11.06**
成　　绩	
评 阅 人	

软件学院

一、 实验目的与意义

实验四、实验五：循环程序设计实验

目的：

1.学会分析循环结构程序的组成部分和循环程序设计的分析思路

2.掌握循环程序的编程与上机调试的方法

3.学会循环控制指令、串操作指令的汇编格式与操作功能

4.继续学习 DEBUG 命令，验证指令的功能

意义：

可简化对处理过程相同，只是每次处理的数据不同而且数据的变化有规律的程序的书写，大大缩短了源程序输入的时间，减少机器代码占用的存储空间，提高了时空效率。

实验六：子程序及宏指令设计实验

目的：

1.掌握子程序的定义方法和参数传递

2.掌握宏定义和宏调用方法，熟悉宏展开方式

意义：

使用户更方便地利用汇编语言进行模块化程序设计，更方便地进行程序的调试，从而加快应用软件的开发。

二、 实验环境

操作系统：Windows 7

调试软件：Masm for Windows 集成实验环境

版本号：MASM 5.0

上机地点：信息 B405

三、 实验的预习内容

实验四：

1.循环控制指令：

①循环控制指令都隐含指定 CX 作为循环计数器

②循环控制指令与循环体入口指令的距离只能在-126~+129 个字节范围以内

2.数据串操作指令

①地址的变化简单而固定：对字节串，地址的变化为+1 或-1；对字串，地址的变化为+2 或-2；

②数据串元素的个数可以作为控制循环的计数

③LODSB 指令：取数据串

将 DS 段 SI 指出的字节数据加载到累加器 AL 中，然后修改 SI 的地址指针

AL ← (DS:SI)

如果 DF=0，则 SI←SI+DELTA

否则 SI←SI-DELTA

实验五：

多重循环的结构与程序设计

①由于使用的循环计数器或其他计数器较多，一般情况下最好不用 16 位寄存器计数，而用 8 位即可。此时不能用 LOOP 来完成各层循环的控制，而需用条件转移指令。

②在寄存器用做多种用途时，在跨越循环层次前后，可能需要利用堆栈保存或恢复寄存器的值。

实验六：

1.子程序

①子程序定义伪指令：PROC 和 ENDP

②定义格式

```
PN PROC [NEAR]/[FAR]
      :
PN   ENDP
```

③用 CALL 指令调用

2.宏指令

宏定义一般格式

```
MNAME  MACRO  [DUMPAR1][DUMPAR2]……
          :
          :
          :
ENDM
```

四、 实验的步骤与调试方法

实验四：

①将数据定义到数据段中，并将首地址送入 SI

②将数据个数送入 CX

③将得到的数据依次与 64H 进行比较

④如果存在相等数据，则显示 Y

⑤如果遍历完所有数据没有发现相等数据，则显示 N

附注：使用数据串操作指令实现时，使用 LODSB 指令

实验五：

①定义子程序，输出两数相乘结果并显示

②第一个乘数放入 BL，第二个乘数放入 BH

③置 BH,BL 初值为 1

④每一行 BL 固定不变，BH 依次加 1，直到 BH=BL

⑤换行，BL 加 1，BH 置为 1，循环步骤三、四，直到 BL>9

实验六：

①求绝对值：若 X 为正，则是其本身；若 X 为负，则取补(NEG 指令)

②求和：分别取 SUM 数据段和当前数据段的 1 字节的数进行相加，将得到的结果依次存到 SUM 中

附注：使用子程序实现时需要调用两次子程序；

使用宏指令时，简化了求和方法，只需要调用一次求和的宏指令

五、 实验数据与实验结果

实验四：

①源程序给出的数据中有 100（64H），所以输出’Y’

```
-d ds:0
0770:0000  30 45 53 89 64 42 06 00-00 00 00 00 00 00 00 00
0770:0010  00 00 00 00 00 00 00 00-00 00 30 00 00 00 1B 00
0770:0020  73 07 A6 01 00 00 00 00-00 00 00 00 00 00 00 00
0770:0030  B8 70 07 8E D8 B8 71 07-8E D0 BC 14 00 33 C0 8D
0770:0040  36 00 00 8B 0E 06 00 8A-04 3C 64 74 05 46 E2 F7
0770:0050  EB 08 B2 59 B4 02 CD 21-EB 08 B2 4E B4 02 CD 21
0770:0060  EB 00 B4 4C CD 21 00 00-00 00 00 00 00 00 00 00
0770:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-g
Y
Program terminated normally
```

②将 64H 改为 46H，结果输出’N’

```
-e ds:0000 30 45 53 89 46 42
-g
N
Program terminated normally
```

实验五：

输出九九乘法表

```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program: TEMPFILE
1*1=1
2*1=2 2*2=4
3*1=3 3*2=6 3*3=9
4*1=4 4*2=8 4*3=12 4*4=16
5*1=5 5*2=10 5*3=15 5*4=20 5*5=25
6*1=6 6*2=12 6*3=18 6*4=24 6*5=30 6*6=36
7*1=7 7*2=14 7*3=21 7*4=28 7*5=35 7*6=42 7*7=49
8*1=8 8*2=16 8*3=24 8*4=32 8*5=40 8*6=48 8*7=56 8*8=64
9*1=9 9*2=18 9*3=27 9*4=36 9*5=45 9*6=54 9*7=63 9*8=72 9*9=81
```

实验六：

①程序初始化，显示存在 DATA1 和 DATA2 单元的值（两个多字节数据都是正数）

```
-d ds:0000
0770:0000  11 12 13 04 20 03 34 15-04 00 00 00 00 00 00 00
0770:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0030  00 00 00 00 00 00 00 00-71 07 00 00 36 00 75 07
0770:0040  A6 01 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0050  8B 0E 08 00 8A 04 22 C0-78 02 79 04 F6 D8 88 04
0770:0060  46 E2 F1 C3 8B 0E 08 00-BF 0A 00 F8 8A 04 8A 1D
0770:0070  12 C3 88 05 46 47 E2 F4-C3 B8 70 07 8E D8 B8 71
```

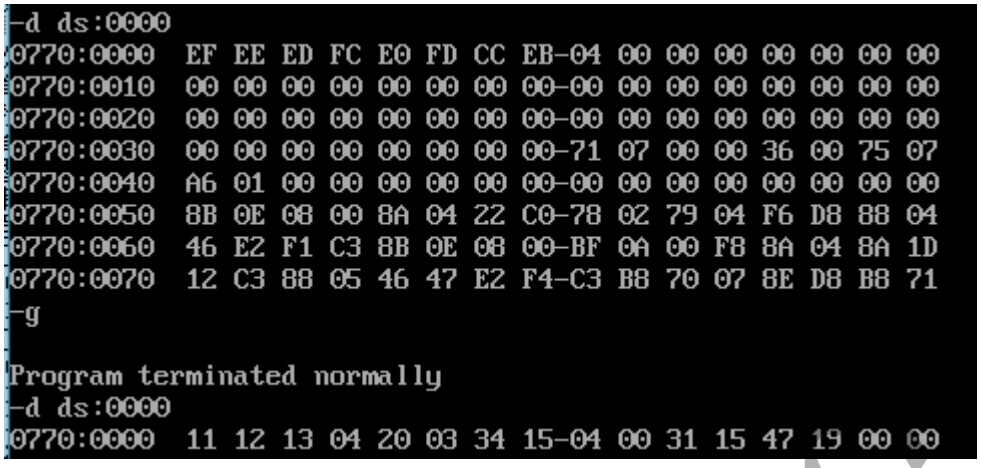
②执行 G 命令，使用 D 命令查看 SUM 中的值是否更改（两个数据的绝对值之和，存放在 SUM 开始的连续单元中）。发现 SUM 中的值已经修改。

```
-g

Program terminated normally
-d ds:0000
0770:0000  11 12 13 04 20 03 34 15-04 00 31 15 47 19 00 00
```

③修改 DATA1 中的数据，将其全改为负的，使得两个多字节数据一正一负。结果正确。

```
-d ds:0000
0770:0000  EF EE ED FC 20 03 34 15-04 00 00 00 00 00 00 00
0770:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0030  00 00 00 00 00 00 00 00-71 07 00 00 36 00 75 07
0770:0040  A6 01 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0770:0050  8B 0E 08 00 8A 04 22 C0-78 02 79 04 F6 D8 88 04
0770:0060  46 E2 F1 C3 8B 0E 08 00-BF 0A 00 F8 8A 04 8A 1D
0770:0070  12 C3 88 05 46 47 E2 F4-C3 B8 70 07 8E D8 B8 71
-g

Program terminated normally
-d ds:0000
0770:0000  11 12 13 04 20 03 34 15-04 00 31 15 47 19 00 00
```

④修改 DATA1 和 DATA2 中的数据，将其全改为负的，使得两个多字节数据都是负数。结果正确。



六、实验用程序清单（要有注释）

实验四：

1. 用一般的指令实现

```
DSEG   SEGMENT ;数据段开始

DATA   DB 30H,45H,53H,89H,64H,42H    ;原始数据，没有数据100

CNT    DW     06       ;CNT单元存放数据长度

DSEG   ENDS        ;数据段结束


SSEG   SEGMENT STACK ;堆栈段开始

SKTOP      DB 20 DUP(0)

SSEG   ENDS        ;堆栈段结束


CSEG   SEGMENT ;代码段开始

       ASSUME   CS:CSEG,DS:DSEG

       ASSUME   SS:SSEG

START: MOV   AX,DSEG   ;初始化数据段基址

       MOV   DS,AX
```

```
        MOV AX,SSEG ;初始化堆栈段基址
        MOV SS,AX
        MOV   SP,LENGTH SKTOP ;设置堆栈指针
        XOR AX,AX          ;将AX清零
        LEA SI,DATA    ;取原数据首地址
        MOV CX,CNT         ;将数据个数送入CX
AGAIN:MOV AL,[SI]    ;将第一个数据送入AL
        CMP AL,64H         ;将AL中的数据与64H进行比较
        JZ   DISP1         ;相等则转到DISP1
        INC     SI         ;不相等则取下一个数
        LOOP   AGAIN ;循环
        JMP DISP2    ;如果6个数据均没有100，则转到DISP2

DISP1: MOV DL,'Y'          ;输出Y
        MOV AH,02H
        INT 21H
        JMP     STOP

DISP2: MOV DL,'N'          ;输出N
        MOV AH,02H
        INT 21H
        JMP     STOP
```

```
STOP:  MOV AH,4CH
       INT 21H

CSEG   ENDS
       END   START
```

2.用数据串操作指令实现

```
DSEG   SEGMENT ;数据段开始
DATA   DB 30H,45H,53H,89H,94H,42H   ;原始数据，没有数据100
CNT    DW     06       ;CNT单元存放数据长度
DSEG   ENDS        ;数据段结束

SSEG   SEGMENT  STACK ;堆栈段开始
SKTOP     DB 20 DUP(0)
SSEG   ENDS        ;堆栈段结束

CSEG   SEGMENT ;代码段开始
       ASSUME   CS:CSEG,DS:DSEG
       ASSUME   SS:SSEG
START:     MOV   AX,DSEG   ;初始化数据段基址
       MOV   DS,AX
```

```
        MOV AX,SSEG      ;初始化堆栈段基址
        MOV SS,AX
        MOV   SP,LENGTH SKTOP ;设置堆栈指针
        XOR AX,AX        ;将AX清零
        LEA SI,DATA   ;取原数据首地址
        MOV CX,CNT       ;将数据个数送入CX
        CLD              ;将标志位DF清零,指针地址增1
AGAIN:LODSB              ;AL←(DS:SI)
        CMP AL,64H       ;将AL中的数据与64H进行比较
        JZ   DISP1       ;相等则转到DISP1
        INC     SI       ;不相等则取下一个数
        LOOP   AGAIN ;循环
        JMP DISP2     ;如果6个数据均没有100，则转到DISP2

DISP1:  MOV DL,'Y'       ;输出Y
        MOV AH,02H
        INT 21H
        JMP     STOP

DISP2:  MOV DL,'N'       ;输出N
        MOV AH,02H
        INT 21H
```

```
        JMP     STOP

STOP:   MOV AH,4CH
        INT 21H

CSEG    ENDS
        END     START
```

实验五

```
DSEG      SEGMENT    ;数据段开始
DSEG      ENDS       ;数据段结束
SSEG      SEGMENT STACK    ;堆栈段开始
SKTOP     DB         50 DUP (0)
SSEG      ENDS       ;堆栈段结束
CSEG      SEGMENT
          ASSUME     CS:CSEG,DS:DSEG
          ASSUME     SS:SSEG

DISP    PROC         ;子程序功能是输出两个数相乘的十进制显示结果
        PUSH   AX ;保存寄存器的值
        PUSH   BX
        PUSH   CX
        PUSH   DX
```

```
        XOR     CX,CX  ;计数器清零
        MOV     BX,10  ;除数
LOOP1:  XOR     DX,DX  ;被除数高为清零
        DIV     BX     ;得到余数和商
        INC     CX     ;位数加1
        PUSH    DX     ;保存余数
        CMP     AX,0   ;若AX=0，则结束，否则继续循环
        JNE     LOOP1
SAVE:   POP     DX     ;保存余数并显示
        ADD     DL,30H
        MOV     AH,02H
        INT     21H
        LOOP    SAVE
        POP     DX     ;恢复寄存器的值
        POP     CX
        POP     BX
        POP     AX
        RET            ;返回
DISP    ENDP

START:  MOV     AX,DSEG  ;初始化数据段基址
        MOV     DS,AX
```

```
        MOV     AX,SSEG   ;初始化堆栈段基址
        MOV     SS,AX
        MOV     SP,SIZE SKTOP    ;设置堆栈指针
        MOV   BL,1      ;被乘数初始化为1
        MOV   BH,1      ;乘数初始化为1
        MOV   CX,9      ;外循环循环9次

REPT2:  MOV     BH,1      ;乘数初始化为1
REPT1:  ADD     BL,30H    ;显示被乘数
        MOV     DL,BL
        MOV     AH,02H
        INT     21H
        SUB     BL,30H  ;将BL复原

        MOV   DL,'*'  ;显示乘号
        MOV     AH,02
        INT     21H

        ADD     BH,30H    ;显示乘数
        MOV     DL,BH
        MOV     AH,02
        INT     21H
```

```
SUB     BH,30H ;将BH复原

MOV     DL,'='  ;显示等号
MOV     AH,02
INT     21H

MOV     AL,BL  ;得到两数相乘结果，保存在AX中
MUL     BH

CALL    DISP     ;调用子程序，将16进制结果转化为10进制显示

MOV     DL,20H  ;输出一个空格
MOV     AH,02H
INT     21H

INC     BH      ;将乘数加1
CMP     BH,BL  ;如果乘数小于或者等于被乘数，则执行循环
JBE     REPT1

MOV     DL,0AH    ;若乘数不满足条件，则输出回车，进入下一行
MOV     AH,02
INT     21H
```

```
        MOV     DL,0DH
        MOV     AH,02
        INT     21H

        DEC     CL      ;计数器减1
        INC     BL      ;被乘数加1
        CMP     CL,0    ;若CL=0，则结束，否则执行循环
        JNE     REPT2

CSEG    ENDS            ;代码段结束
        END     START   ;源程序结束
```

实验六

1. 用子程序实现

```
DSEG    SEGMENT                 ;数据段开始
DATA1   DB      11H,12H,13H,4H  ;测试数据
DATA2   DB      20H,3H,34H,15H
LEN     DW      4               ;数据长度
SUM     DB      4 DUP(?)        ;和
DSEG    ENDS                    ;数据段结束
SSEG    SEGMENT STACK           ;堆栈段开始
SKTOP   DB      50 DUP (0)
SSEG    ENDS                    ;堆栈段结束
```

```
CSEG    SEGMENT
        ASSUME  CS:CSEG,DS:DSEG
        ASSUME  SS:SSEG

ABS     PROC                    ;绝对值子程序
        MOV     CX,LEN
AGAIN1: MOV     AL,[SI]         ;取值
        AND     AL,AL           ;置标志位
        JS      ABSL            ;X<0，转ABSL
        JNS     MOVE            ;X>=0，转MOVE

ABSL:   NEG     AL              ;取补
        MOV     [SI],AL         ;将这个数的值返回到[SI]中

MOVE:   INC     SI              ;取下个数据
        LOOP    AGAIN1          ;循环，直到将每个数都遍历
        RET
ABS     ENDP

SUM1    PROC                    ;数据求和
        MOV     CX,LEN          ;确定循环次数
        MOV     DI,OFFSET SUM   ;取SUM偏移地址
```

```
        CLC                         ;清标志CF
AGAIN2:MOV      AL,[SI]
        MOV     BL,[DI]
        ADC     AL,BL               ;两数相加求和
        MOV     [DI],AL             ;将结果存入SUM对应位置
        INC     SI                  ;取下个数据
        INC     DI
        LOOP    AGAIN2
        RET
SUM1    ENDP

START: MOV      AX,DSEG
        MOV     DS,AX
        MOV     AX,SSEG
        MOV     SS,AX
        MOV     SP,SIZE SKTOP       ;设置堆栈指针
        LEA     SI,DATA1            ;取地址
        CALL    ABS                 ;求绝对值
        LEA     SI,DATA2
        CALL    ABS
        LEA     SI,DATA1
        CALL    SUM1                ;求和
```

```
        LEA         SI,DATA2
        CALL    SUM1
        MOV         AH,4CH
        INT     21H
CSEG    ENDS
        END         START
```

2. 用宏指令实现

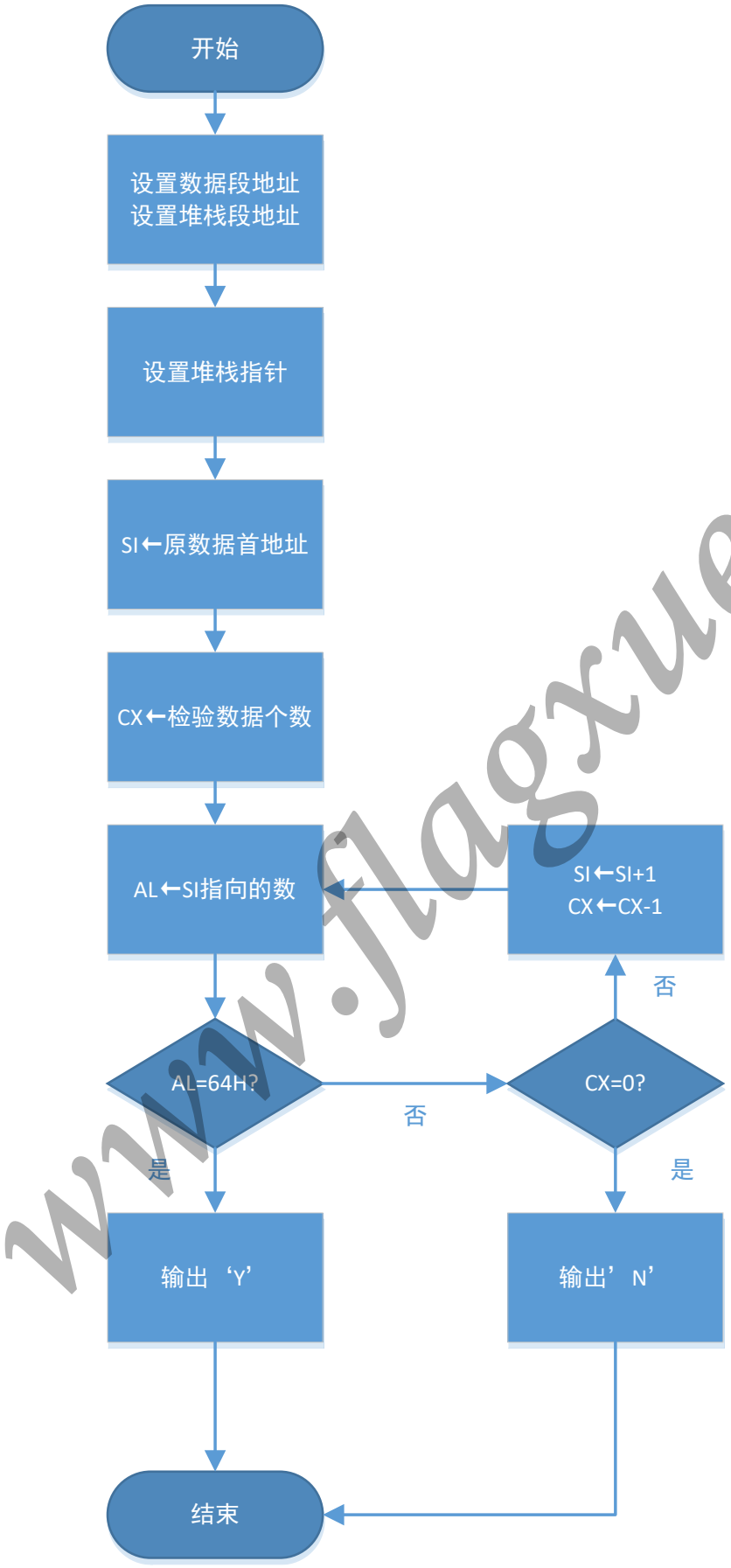
```
        DSEG    SEGMENT             ;数据段开始
DATA1 DB    11H,12H,13H,4H  ;测试数据
DATA2 DB    20H,3H,34H,15H
LEN         DW          4                   ;数据长度
SUM         DB      4 DUP(?)            ;和
DSEG    ENDS                        ;数据段结束
SSEG    SEGMENT STACK           ;堆栈段开始
SKTOP   DB      50 DUP (0)
SSEG    ENDS                        ;堆栈段结束
CSEG    SEGMENT
        ASSUME  CS:CSEG,DS:DSEG
        ASSUME  SS:SSEG

ABS         MACRO   DATA,LEN            ;宏指令求绝对值
        LOCAL AGAIN1,ABSL,MOVE  ;局部标号
```

```
        LEA     SI,DATA
        MOV     CX,LEN
AGAIN1:MOV      AL,[SI]         ;取值
        AND     AL,AL           ;置标志位
        JS      ABSL            ;X<0，转ABSL
        JNS     MOVE            ;X>=0，转MOVE

ABSL:   NEG     AL              ;取补
        MOV     [SI],AL         ;将这个数的值返回到[SI]中

MOVE: INC       SI              ;取下个数据
        LOOP    AGAIN1          ;循环，直到将每个数都遍历
        ENDM

SUM1  MACRO     SUM,SRC,DST,CNT     ;宏指令求和
        LOCAL AGAIN2            ;局部标号
        LEA     BX,SUM              ;取偏移地址
        LEA     DI,DST
        LEA     SI,SRC
        MOV     CX,LEN          ;确定循环次数
        CLC                     ;清标志CF
AGAIN2:MOV      AL,[SI]
```

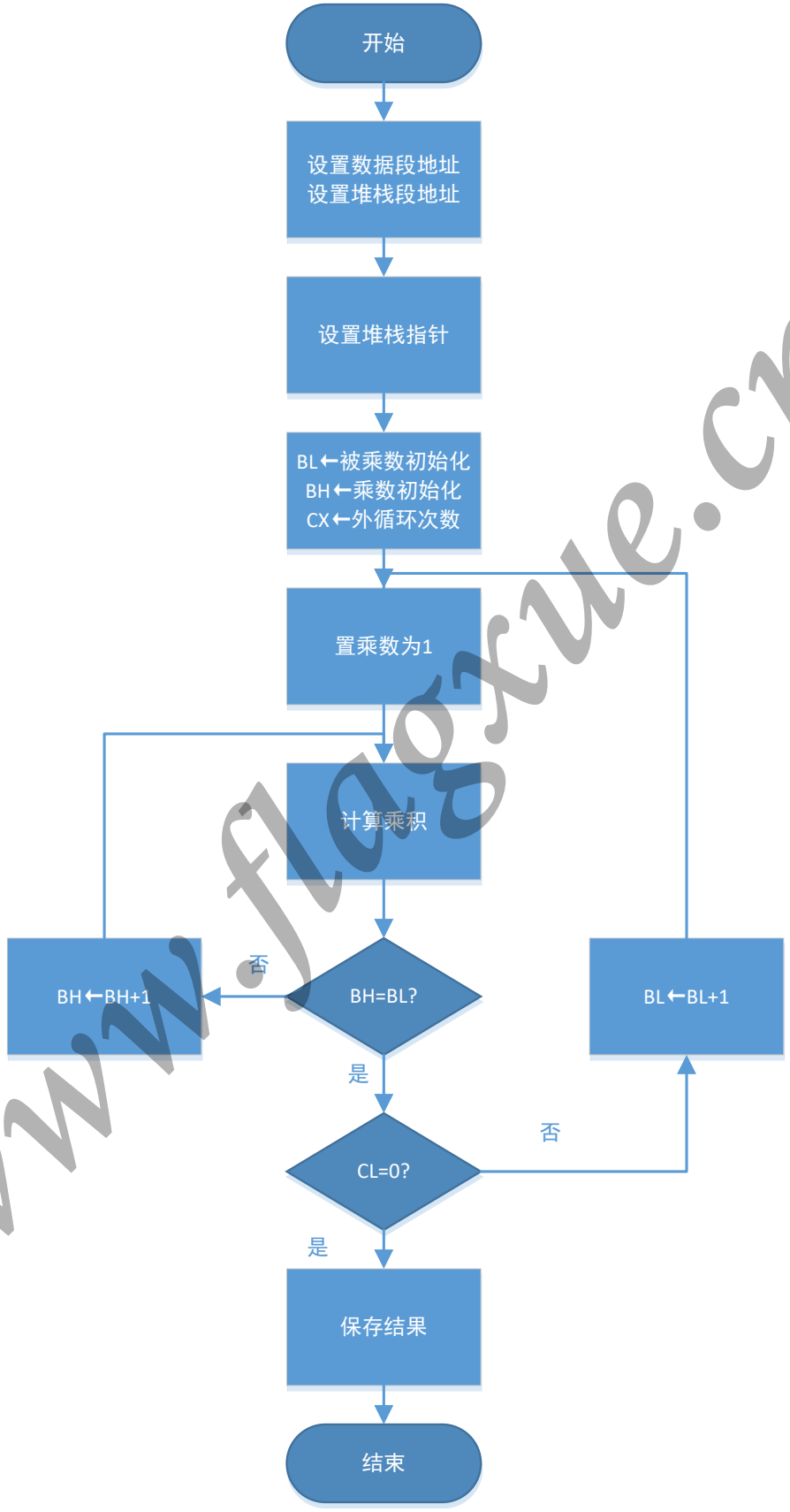
```
        MOV     BL,[DI]
        ADC     AL,BL               ;两数相加求和
        MOV     [DI],AL             ;将结果存入SUM对应位置
        INC     SI                  ;取下个数据
        INC     DI
        INC     BX
        LOOP    AGAIN2
        ENDM

START:  MOV     AX,DSEG
        MOV     DS,AX
        MOV     AX,SSEG
        MOV     SS,AX
        MOV     SP,SIZE SKTOP       ;设置堆栈指针
        ABS     DATA1,LEN           ;求绝对值
        ABS     DATA2,LEN
        SUM1    SUM,DATA1,DATA2,LEN ;求和
        MOV     AH,4CH
        INT     21H
CSEG    ENDS                        ;代码段结束
        END     START               ;源程序结束
```

七、画出每个实验程序的流程图

实验四：



实验五：



实验六：

八、写出子程序说明文件

ABS 子程序说明文件

子程序名：ABS

子程序功能：求多字节带符号数据的绝对值

入口条件：数据地址放在 SI 中

出口条件：无

受影响寄存器：SF,OF

SUM1 子程序说明文件

子程序名：SUM1

子程序功能：求两个多字节数据的和

入口条件：数据地址放在 SI 中

出口条件：和放在 SUM 开始的连续单元中

九、思考题（必需回答）

1.编写循环程序应注意哪些问题？

①有循环出口条件且一定能被执行

②循环控制指令与循环体入口指令的距离只能在-126~+129 个字节范围以内

③多重循环的结构与程序设计中，在寄存器用做多种用途时，在跨越循环层次前后，应注意可能需要利用堆栈保存或恢复寄存器的值。

2.REP 前缀的作用是什么？能用 REP LODSB 指令读取 DS：SI 所指内存中的每个字符吗？为什么？

REP 前缀作用：根据 CX 值得情况重复执行后面的基本串操作指令，每执行一次则 CX 减 1，直到 CX=0 为止。

不能。加入重复前缀之后，累加器中载入的内容不断取代原来的结果，只留下最后一个加载的元素。

3.简述 CALL 指令和 JMP 指令的主要区别，它们可以互相代替吗？

CA11 指令是将子程序调用进入程序段，在执行到 RET 指令后，会返回到 CALL 后继续执行。JMP 指令用作直接跳转，是用在分支结构中将程序跳转至任意位置

的指令二者区别很大，不可替换。

4.简述宏与子程序的区别和联系，什么样的程序块适合于编写成子程序？

区别：宏在汇编期间展开，占用内存空间与调用次数有关，每调用一次就把宏定义展开一次，占用内存空间多，但运行速度快，十分灵活；子程序在程序执行期间由主程序调用，占用内存空间少，只占用自身大小的空间，但运行速度较低，不够灵活。

联系：*宏和*子程序都可用来简化程序,并可使程序多次对它们调用。

一般来说，要求运行速度快，内存容量较大时，采用宏的办法比较灵活、有效。

十、结束语

实验教学与理论教学相辅相成，提高了我分析问题和解决问题的能力，加深了我对循环结构、子程序和宏指令的理解，使我对汇编语言有了更加深刻地认识，获益匪浅。

十一、参考文献

1.《汇编语言程序设计》(第四版)，高福祥 齐志儒主编，东北大学出版社

2.《汇编语言实验教程》，张坤编著，清华大学出版社

3.《汇编语言及其应用教程》，李浪，熊江，齐忠主编，华中科技大学出版社

实验成绩

考查内容	分数	得分
做好实验内容的预习，写出预习报告	10	
了解实验题目的调试方法	10	
按实验要求预先设计好程序	10	
认真记录实验数据并分析实验结果	10	
实验后按要求书写实验报告，记录实验用数据及运行结果	30	
创新能力强，在实验中设计的程序有一定的通用性，算法优化	20	
实验过程中，具有严谨的学习态度，认真、踏实、一丝不苟的科学作风	10	