学院：**信息学院**　学期：**2022 年秋季学期**　课程名称：**汇编语言程序设计**　指导教师：**周丽华**

专业：**计算机科学与技术**　年级：**2020**　姓名：**胡诚皓**　学号：**20201060330**

实验五 分支程序设计

一、实验目的

1. 掌握分支程序的结构。
2. 掌握分支程序的设计、编写及调试方法。

二、实验内容

1. 复习教材中分支结构程序设计的相关内容。
2. 分析下面程序（程序 1）的功能。。

```
DATA    SEGMENT
BUF     DB 38H,73H,1FH
MAX     DB ?
DATA    ENDS
CODE    SEGMENT
        ASSUME   CS:CODE,DS:DATA
START:  MOV   AX,DATA
        MOV   DS,AX
        MOV   AL,BUF
        CMP   AL,BUF+1
        JNB   LP1
        MOV   AL,BUF+1
LP1:    CMP   AL,BUF+2
        JAE   LP2
        MOV   AL,BUF+2
LP2:    MOV   MAX,AL
        MOV   AH,4CH
        INT   21H
CODE    ENDS
        END START
```

3. 编写程序 2：判断 BUF 单元存放的带符号字节数的正负，并在屏幕上给出正负信息。

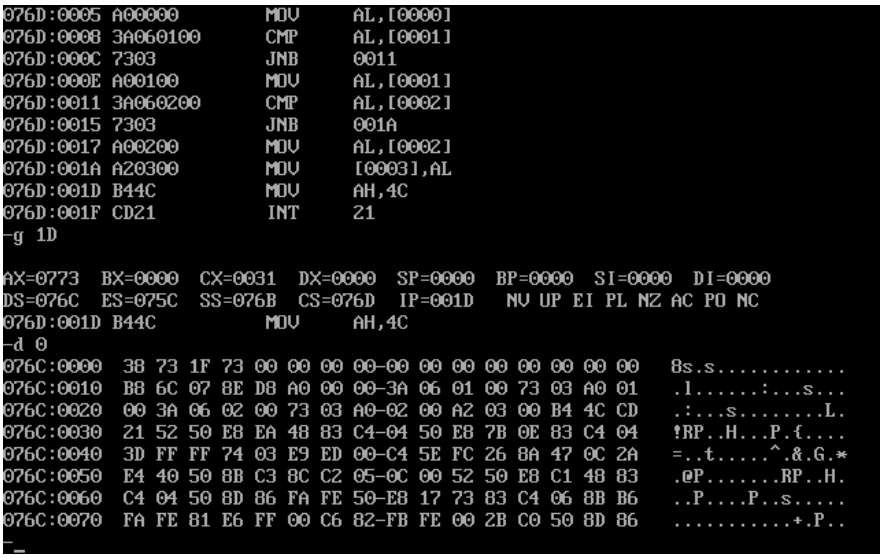
4. 编写程序 3：实现任意给定 X 值（-128≤X≤127），求符号函数 Y 的值，存放于内存单元。

$$Y=\begin{cases}1, X>0\\0, X=0\\-1, X<0\end{cases}$$

三、实验结果（截图）

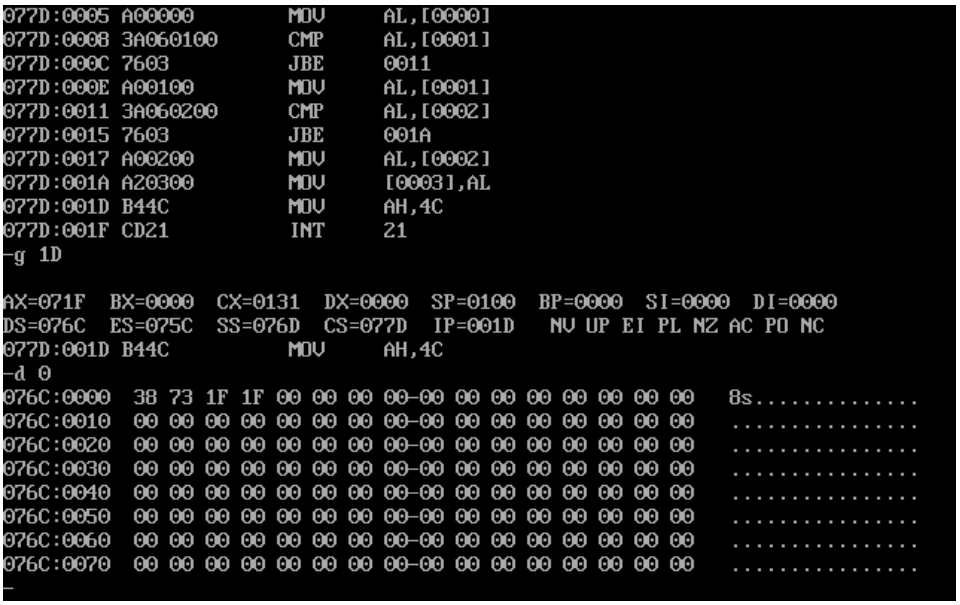
1. 程序 1 的理解

程序 1 的这段代码取出字节类型、长度为 3 的数组 BUF 中最大的数。显然最后 MAX 单元中的内容为 BUF 数组中最大的数，即 73H。从下图中可以看出 076C:0003 内存单元中 MAX 的值为 73H。



2. 修改程序 1 实现取得最小数存入 MIN

很显然，将原先代码中的条件跳转指令做对应修改即可。把原来的不小于 JNB 改为不大于 JNA、原来的大于等于 JAE 改为小于等于 JBE。从下图中可以看出 076C:0003 内存单元中 MIN 的值为 1FH。



3. 程序 2

使用分支语句可以很轻松的完成要求，需要注意在分支语句块中的每个分支的最后进行跳转，防止进入同一语句块的其他分支再次执行。

```
07BB:0040 E81400        CALL    0057
-u
07BB:0043 EB01          JMP     0046
07BB:0045 90            NOP
07BB:0046 8306E80301    ADD     WORD PTR [03E8],+01
07BB:004B 833EE80303    CMP     WORD PTR [03E8],+03
07BB:0050 72BC          JB      000E
07BB:0052 B8004C        MOV     AX,4C00
07BB:0055 CD21          INT     21
07BB:0057 50            PUSH    AX
07BB:0058 B409          MOV     AH,09
07BB:005A CD21          INT     21
07BB:005C 58            POP     AX
07BB:005D C3            RET
07BB:005E FE8BE55D      DEC     BYTE PTR [BP+DI+5DE5]
07BB:0062 C3            RET
-g 52
Positive
Positive
Positive

AX=001F  BX=0000  CX=054E  DX=000D  SP=0100  BP=0000  SI=0002  DI=0000
DS=076C  ES=075C  SS=07AB  CS=07BB  IP=0052   NV UP EI PL ZR NA PE NC
07BB:0052 B8004C        MOV     AX,4C00
-_
```

3. 程序 3

条件测试方法与程序 2 类似，结果不需要输出，保存在 Y 的内存单元中即可,此处使用的 X 为-5，在 DS:0002 处为 Y 的值，为 FF，即-1。

```
077D:002B 8BEC          MOV     BP,SP
077D:002D 81EC8600      SUB     SP,0086
077D:0031 57            PUSH    DI
077D:0032 56            PUSH    SI
077D:0033 B8BE05        MOV     AX,05BE
077D:0036 50            PUSH    AX
077D:0037 E8C371        CALL    71FD
077D:003A 83C402        ADD     SP,+02
077D:003D 8BF0          MOV     SI,AX
077D:003F 0BF6          OR      SI,SI
-g 26

AX=076C  BX=00FB  CX=013B  DX=0000  SP=0100  BP=0000  SI=0000  DI=0000
DS=076C  ES=075C  SS=076D  CS=077D  IP=0026   NV UP EI NG NZ NA PO NC
077D:0026 B8004C        MOV     AX,4C00
-d 0
076C:0000  FB FF 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

四、实验报告要求（习题）

1. 程序 1

这段汇编代码相当于如下的高级语言代码

```
AL = BUF[0];
if (AL < BUF[1]) {
    AL = BUF[1];
} else if (AL < BUF[2]) {
    AL = BUF[2];
}

MAX = AL;
```

```
DATA    SEGMENT
    ; 数组 BUF
    BUF DB  38H,73H,1FH
    ; 用于存放最大值
    MAX DB  0
DATA    ENDS

STACK SEGMENT STACK
    DW 128 DUP(0)
STACK ENDS

CODE    SEGMENT
    ASSUME  CS:CODE,DS:DATA
START:
    ; 定义数据段
    MOV AX,DATA
    MOV DS,AX
    ; AL 初始值为 BUF[0]
    MOV AL,BUF
    ; if (AL < BUF[1])
    CMP AL,BUF+1
    JNB LP1
    MOV AL,BUF+1
LP1:
    ; else if (AL < BUF[2])
    CMP AL,BUF+2
    JAE LP2
```

```
    MOV AL,BUF+2
LP2:
    MOV MIN,AL
    MOV AH,4CH
    INT 21H
CODE    ENDS
    END START
```

2. 程序 **1** 改为找三者最小

相当于如下的高级语言代码

```
AL = BUF[0];
if (AL > BUF[1]) {
    AL = BUF[1];
} else if (AL > BUF[2]) {
    AL = BUF[2];
}
MIN = AL;
```

```
DATA    SEGMENT
    ; 数组 BUF
    BUF DB  38H,73H,1FH
    ; 用于存放最大值
    MAX DB  0
DATA    ENDS

STACK SEGMENT STACK
    DW 128 DUP(0)
STACK ENDS

CODE    SEGMENT
    ASSUME  CS:CODE,DS:DATA
START:
    ; 定义数据段
    MOV AX,DATA
    MOV DS,AX
    ; AL 初始值为 BUF[0]
   MOV AL,BUF
    ; if (AL > BUF[1])
```

```
    CMP AL,BUF+1
    JNA LP1
    MOV AL,BUF+1
   LP1:
    ; else if (AL > BUF[2])
    CMP AL,BUF+2
    JBE LP2
    MOV AL,BUF+2
LP2:
    MOV MIN,AL
    MOV AH,4CH
    INT 21H
CODE    ENDS
       END START
```

3. 程序 2

本程序相当于下面的高级语言程序

```
int main() {
    int i;
    for (int i = 0; i < 3; i++) {
        if (BUF[i] < 0) {
            printf("Negative");
        } else if (BUF[i] > 0) {
            printf("Positive");
        } else {
            printf("Zero");
        }
    }

    return 0;
}
```

创建了个子过程 printDx 用于输出，将要输出内容内存段的首地址放到 DX 中后直接调用该过程即可输出。

```
DATA    SEGMENT
    ; 数组 BUF
    BUF DB  38H,73H,1FH
    ; 字符串常量段
```

```
    NEG_STR DB 'Negative', 10, '$'
    POS_STR DB 'Positive', 10, '$'
    ZERO_STR DB 'Zero', 10, '$'
    ; 变量段
    ORG 1000
    I DW 0

DATA    ENDS

STACK SEGMENT STACK
    DW 128 DUP(0)
STACK ENDS

CODE    SEGMENT

main PROC FAR
    ASSUME  CS:CODE,DS:DATA,SS:STACK
START:
    ; 定义数据段
    MOV AX,DATA
    MOV DS,AX
    ; 循环变量初始化
    MOV I, 0
    jmp LOOP_JUDGE
L1:  ; BUF[i]<0
    mov si, I
    mov al, BUF[si]
    cbw
    test ax, ax
    jns L2
    lea dx, NEG_STR
    call printDx
    jmp LOOP_INC
L2:  ; BUF[i]>0
    mov si, I
    mov al, BUF[si]
    cbw
```

```
    test ax, ax
    jle L3
    lea dx, POS_STR
    call printDx
    jmp LOOP_INC
L3:  ; else，即 BUF[i]==0
    lea dx, ZERO_STR
    call printDx
    jmp LOOP_INC
LOOP_INC:  ; 循环变量+1
    add I, 1
LOOP_JUDGE:  ; 循环判断
    cmp I, 3
    jb L1
EXIT:  ; 退出段
    mov ax, 4C00H
    int 21H
main ENDP

; 输出 DX 处的内容
printDx PROC NEAR:
    push ax
    mov ah, 9
    int 21h
    pop ax
    ret
printDx ENDP

CODE    ENDS
    END START
```

4. 程序 3

相当于下面的 C 语言代码

```
if (X > 0)
    Y = 1
else if (X == 0)
    Y = 0
else
```

```
    Y = -1
```

```
DATA SEGMENT
    X DB -5
    Y DB 0
DATA ENDS

STACK SEGMENT STACK
    DW 128 DUP(0)
STACK ENDS

CODE SEGMENT
    ASSUME cs:CODE, ds:DATA, ss:STACK
START:
    ; 设置段寄存器
    mov ax, DATA
    mov ds, ax
    mov bl, X
L1:  ; if X>0
    test bl, bl
    jle L2
    mov Y, 1
    jmp EXIT
L2:  ; else if (X == 0)
    test bl, bl
    jnz L3
    mov Y, 0
    jmp EXIT
L3:  ; else
    mov Y, -1
EXIT:  ; 退出段
    mov ax, 4C00H
    int 21H
CODE ENDS
END START
```

5. 如何用 debug 查看程序运行结果

下面叙述本次报告使用 debug 查看运行结果的过程

① 使用 debug [exe 程序名]，进入程序的调试

② 使用 g [断点]运行程序至断点处，断点可以使用相对地址指定，需要注意的是，指定的断点位置的语句并不会运行

③ 使用 r 命令观察寄存器，计算过程中把结果存在了寄存器或相应内存单元中

④ 使用 d [内存地址]查看指定内存单元以后一段的内容，该命令中的地址也可以使用相对地址进行指定，默认查看的是 DS:指定偏移地址的位置

6. 遇到的问题及如何解决

本次实验主要对分支程序进行学习，在实验过程中，若是直接写汇编，当分支数多了之后，逻辑关系很难理清，比较耗费时间。在实践过后，发现可以先用高级语言写出要完成的内容，再将其翻译为汇编代码，这样不容易出错。在这过程中,需要注意在进入每个分支语句结束后,要跳转到分支语句块结束后的位置。

五、个人体会与总结

分支语句作为程序中几乎必用的语句，非常重要。在使用条件跳转指令时，注意要和实际逻辑判断相反，即当不满足跳转条件时正常执行该分支内的内容，执行完某分支内的内容后，使用无条件跳转指令 JMP 转出该块分支语句。