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**

实验七 循环程序设计

一、实验目的

1. 掌握循环程序的结构。
2. 掌握循环程序的设计、编写及调试方法。

二、实验内容

1. 复习教材中循环结构程序设计的相关内容。
2. 编写程序 1 计算 1~100 的和。
3. 编写程序 2 计算 S=1+2×3+3×4+…+N×(N+1)，直到 S>200 为止。
4. 编写程序（程序 3）实现下列功能：已知数组 A 中包含 15 个互不相等的整数，数组 B 中包含 20 个互不相等的整数，试编一程序，将既在 A 中出现又在 B 中出现的整数存放于数组 C 中。

三、实验结果（截图）

1. 程序 1

求和的结果存在 076C:0002 处，可以看到是 13BA，即十进制下的 5050，是正确的。

```
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
077D:0041 7461          JZ      00A4
-g 25

AX=0064  BX=0000  CX=013A  DX=0000  SP=0100  BP=0000  SI=0000  DI=0000
DS=076C  ES=075C  SS=076D  CS=077D  IP=0025   NV UP EI PL NZ NA PO NC
077D:0025 B8004C        MOV     AX,4C00
-d 0
076C:0000  65 00 BA 13 00 00 00 00-00 00 00 00 00 00 00 00   e...............
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   ................
-
```

2. 程序 2

076C:0002 内存单元处的内容就是 S 最终的结果，为 00EF，即 239。

1+2*3+3*4+4*5+5*6+6*7+7*8=167

1+2*3+3*4+4*5+5*6+6*7+7*8+8*9=239

结果正确。

```
077D:0025 01060200      ADD     [0002],AX
077D:0029 FF060000      INC     WORD PTR [0000]
077D:002D EBE5          JMP     0014
077D:002F B8004C        MOV     AX,4C00
077D:0032 CD21          INT     21
077D:0034 BE0550        MOV     SI,5005
077D:0037 E8C371        CALL    71FD
077D:003A 83C402        ADD     SP,+02
077D:003D 8BF0          MOV     SI,AX
077D:003F 0BF6          OR      SI,SI
-g 2f

AX=0048  BX=0000  CX=0144  DX=0000  SP=0100  BP=0000  SI=0000  DI=0000
DS=076C  ES=075C  SS=076D  CS=077D  IP=002F   NV UP EI PL NZ NA PE NC
077D:002F B8004C        MOV     AX,4C00
-d 0
076C:0000  09 00 EF 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   ................
-_
```

3. 程序 3

076C:0046 内存单元处的内容就是数组 C 最终的结果，为 1000、0700、1100、1300、0C00、0600、FFFF、0F00，即 16、7、17、19、12、6、-1、15。

076C:0064 内存单元处的内容为 COUNT 最终的结果，记录了 A、B 数组中相同元素的个数，结果为 8，是正确的。

```
0783:004A EB07          JMP     0053
0783:004C 90            NOP
0783:004D FF066800      INC     WORD PTR [0068]
0783:0051 EBCE          JMP     0021
0783:0053 FF066600      INC     WORD PTR [0066]
0783:0057 EBBB          JMP     0014
0783:0059 B8004C        MOV     AX,4C00
0783:005C CD21          INT     21
0783:005E 867AFF        XCHG    BH,[BP+SI-01]
0783:0061 72E9          JB      004C
-g 59

AX=0014  BX=0010  CX=01CE  DX=0000  SP=0100  BP=0000  SI=0026  DI=0000
DS=076C  ES=075C  SS=0773  CS=0783  IP=0059   NV UP EI PL ZR NA PE NC
0783:0059 B8004C        MOV     AX,4C00
-d 0
076C:0000  00 00 10 00 07 00 11 00-13 00 02 00 0C 00 06 00   ................
076C:0010  0E 00 FC FF 0D 00 FE FF-FF FF 0F 00 14 00 11 00   ................
076C:0020  18 00 15 00 FA FF 13 00-05 00 0B 00 FF FF 0C 00   ................
076C:0030  FD FF 01 00 10 00 04 00-16 00 03 00 07 00 06 00   ................
076C:0040  FB FF 0F 00 19 00 10 00-07 00 11 00 13 00 0C 00   ................
076C:0050  06 00 FF FF 0F 00 00 00-00 00 00 00 00 00 00 00   ................
076C:0060  00 00 00 00 08 00 0F 00-14 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   ................
-
```

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**

四、实验报告要求（习题）

1. **程序 1——计算 1~100 的和**

这段汇编代码相当于如下的高级语言代码

```
sum = 0;
for (int i = 1; i <= 100; i++) {
    sum += i;
}
```

```
DATA SEGMENT
        I DW 1
        SUM DW 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
        ; 初始化变量
        ; for (i=1)
        mov I, 1
        mov SUM, 0
Iloop: ; i 循环
        ; for(i<=100)
        cmp I, 100
        ja EXIT
        ; sum+=i
        mov ax, I
        add SUM, ax
Iinc: ; 循环变量 i 自增
        inc I
```

```
        jmp Iloop
EXIT:  ; 退出代码
    mov ax, 4C00H
    int 21H
CODE ENDS
END START
```

2. 程序 2——计算和式直到 S 超过 200

相当于如下的高级语言代码

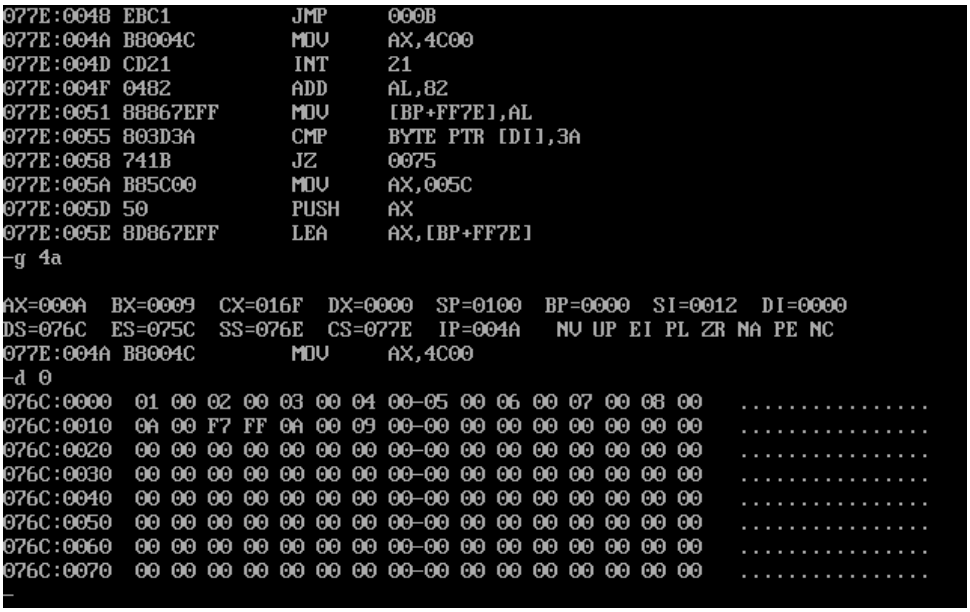
```
sum = 1, n = 2, ax = 0;
while (sum <= 200) {
    ax = n*n + n;
    sum += ax;
    n++;
}
```

```
DATA SEGMENT
        N DW 2
        SUM DW 1
DATA ENDS

STACK SEGMENT STACK
        DW 128 DUP(0)
STACK ENDS

CODE SEGMENT
        ASSUME cs:CODE, ds:DATA, ss:STACK
START:
        mov ax, DATA
        mov ds, ax
        ; 初始化变量
        mov SUM, 1
        mov N, 2
        mov ax, 0
AGAIN:
        ; while(sum <= 200)
        cmp sum, 200
        jg EXIT
```

```
        ; 减少指令，将 n*(n+1)转换为 n^2+n
        ; n^2+n 存在 ax 中
        ; ax = n*n+n
        mov ax, N
        imul ax
        add ax, N
        ; 往 sum 上加
        ; sum += ax
        add SUM, ax
        ; n++
        inc N
        jmp AGAIN
EXIT:  ; 退出代码
    mov ax, 4C00H
    int 21H

CODE ENDS
END START
```

3. 程序 3——找到两个数组中的相同元素

采用枚举法，从 A 数组中的元素出发，在 B 中查看是否有相同的，有相同的则记录在 C 数组中，并将 COUNT 自加 1。



相当于下面的高级语言程序代码

```
int A[15] = {0, 16, 7, 17, 19, 2, 12, 6, 14, -4, 13, -2, -1,
```

```
15, 20};
int B[20] = {17, 24, 21, -6, 19, 5, 11, -1, 12, -3, 1, 16, 4,
22, 3, 7, 6, -5, 15, 25};
int C[15];
int COUNT=0;

int main() {
        int bx=0;
        int i, j;

        for (i = 0; i < 15; i++) {
          for (j = 0; j < 20; j++) {
             if (A[i] == B[j]) {
                COUNT += 1;
                C[bx] = A[i];
                bx += 2;
                break;
             }
          }
        }

        return 0;
}
```

```
DATA SEGMENT
        A DW 0, 16, 7, 17, 19, 2, 12, 6, 14, -4, 13, -2, -1,
15, 20
        B DW 17, 24, 21, -6, 19, 5, 11, -1, 12, -3, 1, 16, 4,
22, 3, 7, 6, -5, 15, 25
        C DW 15 DUP(0)
        ; 记录相同元素的个数
        COUNT DW 0
        ; 循环变量
        I DW 0
        J DW 0
DATA ENDS
```

```
; 清零栈段空间
STACK SEGMENT STACK
        DW 128 DUP(0)
STACK ENDS

CODE SEGMENT
        ; 设置段寄存器
        ASSUME cs:CODE, ds:DATA, ss:STACK
START:
        mov ax, DATA
        mov ds, ax
        ; 初始化变量
        ; BX 记录 C 数组中当前可以存数的位置，每次要加 2
        mov bx, 0
        mov I, 0
        mov J, 0
Iloop: ; 外层 i 循环
        cmp I, 15
        jae EXIT
        ; for (j=0)
        mov J, 0
Jloop: ; 内层 j 循环
        cmp J, 20
        jae Iinc
        ; if (A[i] == B[j])
        mov si, I
        ; 注意将下标乘 2 转为字节位置
        shl si, 1
        mov ax, A[si]
        mov si, J
        shl si, 1
        cmp ax, B[si]
        jne Jinc
        ; count++
        add COUNT, 1
        ; C[bi] = A[i]
        mov C[bx], ax
```

```
        ; bx+=2
        add bx, 2
        ; break
        jmp Iinc
Jinc: ; 循环变量 j 自增
        ; for(j++)
        inc J
        jmp Jloop
Iinc:  ; 循环变量 i 自增
        ; for(i++)
        inc I
        jmp Iloop
EXIT:  ; 退出代码
    mov ax, 4C00H
    int 21H

CODE ENDS
END START
```

4. 寄存器的使用

ax 寄存器存储中间变量，用于存放无法使用两个内存中的操作数指令中的某个操作数，bx 用于数组的定位，si 用作偏移地址访问数组。

5. 循环与条件控制程序的总结

循环与条件控制语句中，对于数值条件的判断，由于需要进行跳转，往往设计为达到相反的条件时（即不满足循环/分支条件时）跳转到逻辑上的下一个程序段即可。对于循环结构的程序，一般在循环体开始与循环变量变化段设置两个 label 以进行跳转。

6. 使用 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

7. 求取偶数和或奇数和

只要将循环变量变化处的代码从每次加一改为加二即可，需要注意循环变量初始值。下图分别为求得偶数和 09F6H=2550 和奇数和 09C4H=2500 的结果

```
-d 26
075C:0020                   FF FF-FF FF FF FF 51 07 4C 01        ......Q.L.
075C:0030  64 06 14 00 18 00 5C 07-FF FF FF FF 00 00 00 00   d.....\.........
075C:0040  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
075C:0050  CD 21 CB 00 00 00 00 00-00 00 00 00 00 00 00 00   .!..............
075C:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
075C:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
075C:0080  00 0D 50 52 4F 4A 45 43-54 2E 45 58 45 0D 00 00   ..PROJECT.EXE...
075C:0090  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
075C:00A0  00 00 00 00 00 00                                 ......
-g 26

AX=0064  BX=0000  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 PL NZ NA PO NC
077D:0026 B8004C        MOV     AX,4C00
-d 0
076C:0000  66 00 F6 09 00 00 00 00-00 00 00 00 00 00 00 00   f...............
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   ................
-
```

```
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
077D:0041 7461          JZ      00A4
077D:0043 8D867EFF      LEA     AX,[BP+FF7E]
-g 26

AX=0063  BX=0000  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 PL NZ NA PO NC
077D:0026 B8004C        MOV     AX,4C00
-d 0
076C:0000  65 00 C4 09 00 00 00 00-00 00 00 00 00 00 00 00   e...............
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   ................
-
```

8. 修改程序 2 实现不同的和式计算

要计算(N-1)*(N+1)的值，为了编程方便，将其转化为 N^2-1，循环应从 2 开始，递增 1 即可。

1×3+2×4+3×5+4×6+5×7+6×8+7×9=196

1×3+2×4+3×5+4×6+5×7+6×8+7×9+8×10=276

结果中 N 应为 9，循环完成后应为 10，S 应为 276。下图中 076C:0000 处为 N 的值 000AH，076C:0002 处的值为 0114H，结果正确。

```
077D:0028 FF060000      INC     WORD PTR [0000]
077D:002C EBE6          JMP     0014
077D:002E B8004C        MOV     AX,4C00
077D:0031 CD21          INT     21
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 2e

AX=0050  BX=0000  CX=0143  DX=0000  SP=0100  BP=0000  SI=0000  DI=0000
DS=076C  ES=075C  SS=076D  CS=077D  IP=002E   NV UP EI PL NZ AC PO NC
077D:002E B8004C        MOV     AX,4C00
-d 0
076C:0000  0A 00 14 01 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   ................
-
```

9. 遇到的问题及解决

在代码编写过程中，和上一份实验一样，遇到了数组下标与字节位置不对应的问题。在汇编语言中，汇编程序并不会自动帮我们处理数组下标与字节偏移量的对应关系，即不会将数组下标乘上数组中元素的字节大小，需要自己完成。可以保持记录下标位置的变量仍然记录下标，由于字长是 2 的倍数，在取数组中元素时再通过 SHL 逻辑移位指令完成数组下标到字节偏移量的转换即可。

五、个人体会与总结

与前几次实验相同，先使用 C 代码快速完成逻辑上的代码编写，再人工将其“编译”为汇编语言代码，这样子只需要按照规律将各结构的语句转换为汇编代码即可，逻辑上不会混乱，编写起来还更快。在观察真正的 C 编译器的编译结果中，发现其将各个变量使用 rbp-偏移地址进行间接寻址，统一管理内存。这样的方法可以使得整体变量存放更加有序合理。