哈尔滨工业大学（深圳）

汇编语言与接口技术 实验报告

实验二：程序流控制和子程序

学号：200111325

姓名：杨行

日期：2022.11.26

目录

必做题 1
1 问题描述 1
2 解决方案和代码 1
3 调试过程与结果 5
选做题 6
1 问题描述 6
2 解决方案及代码 7
3 调试过程与结果 10
总结 11

必做题

1 问题描述

给定一个字节类型的无序数组，请将该数组按从小到大的顺序排序，并计算数组之和。要求：

排序用子程序实现；

数据段定义如指导书；

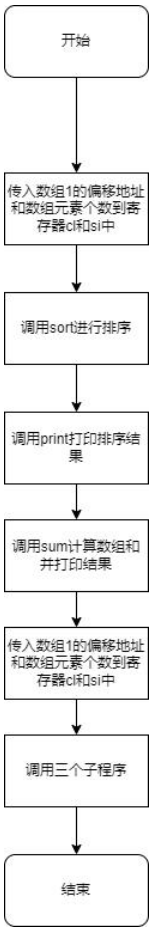
单步调试，查看结果，或者用 dos 功能调用输出排序结果。

2 解决方案和代码

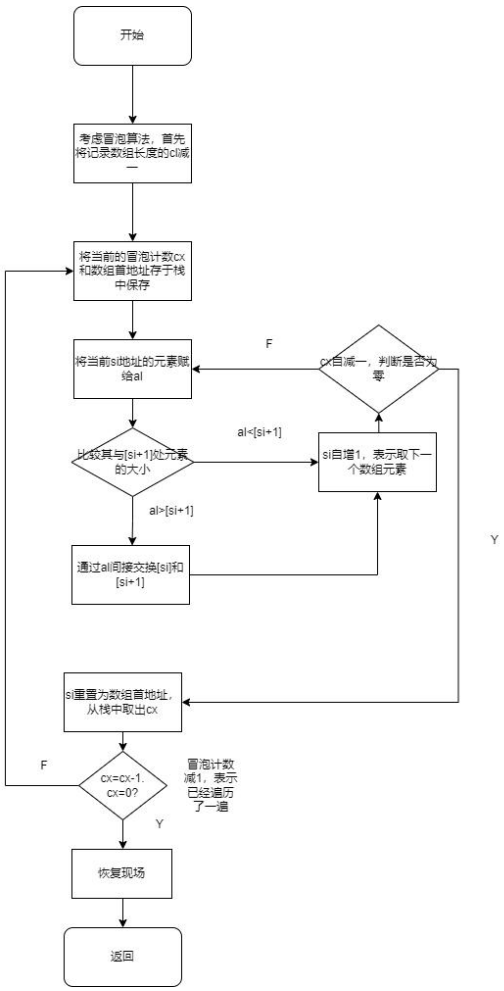
解题思路:

首先考虑采用冒泡排序的方法将数组按从小到大的顺序排序。通过对排序后的数组进行累加，得到数组之和的结果。由于题目要求排序用子程序完成，所以排序和数组求和的功能都由子程序实现，通过通过设计一个 print 子程序输出排序后的数组。

主程序实际流程图:



sort 子程序的流程设计图:



Print 的设计思路既是通过每次除以 10 取余数的方式将数字按照每位输出。

代码:

```
DATAS SEGMENT
    num1  db 10,6,24,39,33,77,18,100,36,4   ;数组
    n1 db 10  ;数字个数
    ;num1 db 10,6
    ;n1 db 2
    num2  db 45,12, 30, 22, 100, 79 ;数组
    n2 db 6  ;数字个数
    result1 dw ? ;数组和
    result2 dw ? ;数组和
    string  db 'After Sort the num is:',0ah,0dh,'$';$终止符
    string1  db  'sum:$'
    ;此处输入数据段代码
DATAS ENDS

STACKS SEGMENT
    db 40 dup(0) ;此处输入堆栈段代码
STACKS ENDS

CODES SEGMENT
main proc far
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
    MOV AX,DATAS
    MOV DS,AX
    ;此处输入代码段代码
    mov c1,n1   ;传入数组数量参数
    mov si, offset num1  ;传入数组初始地址
    call sort        ;调用子程序
    call print
    call sum
    mov result1,bx



    mov c1,n2
    mov si,offset num2
    call sort
    call print
    call sum
    mov result2,bx


    MOV AH,4CH
    INT 21H
    ret
main endp

;-------------------------------------------------
;子程序名：sort
;功能：通过冒泡排序对给定的数组进行排序
;入口参数：数组偏移地址、数组元素个数
;出口参数：重新排序的数字存于原来的内存地址
;占用寄存器：AX, BX, CX, DX, SI,FLAG
;-------------------------------------------------

sort proc near
    mov ch, 0
    push cx
    dec c1
OUTER:
    push cx  ;保存当前冒泡的计数
    push si  ;存入数组首地址
INNER:
        mov a1,[si]
        cmp a1, [si+1]              ; 判断IF (NUM[SI] > NUM[SI+1])
        JLE UPDATE                  ; 不是跳转
        XCHG a1,[si+1]              ; 是交换,SWAP (NUM[SI], NUM[SI+1])
        XCHG a1,[si]

UPDATE:
        inc si
        LOOP INNER              ; CONTINUE
        pop si
        pop cx
        LOOP OUTER
        pop cx
        ret
sort endp

```

```
;-----------------------------------------------------
;子程序名：sum
;功能：对数组求和并输出结果至屏幕上
;入口参数：数组偏移地址、数组元素个数
;出口参数：计算结果存于BX
;占用寄存器：AX, BX, CX, DX, SI,FLAG
;-----------------------------------------------------

sum  proc near
     mov ch,0
     push cx
     mov bx,0
LOOP1:
     add  bl,[si]
     adc  bh,0
     inc si
     LOOP LOOP1
     push bx;保存结果
     ;*****打印结果****经过前一个子程序恰好cx为0，这里就没有初始化
     mov dx,offset string1;打印结果提示
     mov ah,9
     int 21h
     mov ax,bx
     mov bx,10
DIV_:
     mov dx,0
     div bx ;商在ax,余在dx
     push dx
     inc cx
     cmp ax,0
     jnz DIV_

do:
     pop dx
     or dx,30h
     mov ah,2
     int 21h
     loop do
     call newline
     pop bx
     pop cx
     ret
```

```
     sum endp
;-----------------------------------------------------
;子程序名：print
;功能：将数组元素输出到屏幕上
;入口参数：数组偏移地址、数组元素个数
;占用寄存器：AX, BX, CX, DX, SI, FLAG
;-----------------------------------------------------

print proc near
     push cx
     push ax
     push bx
     push dx;
     push si;
     mov  dx,offset string;打印数组提示
     mov ah,9
     int 21h
     mov bl,10;设置被除数
initial:
     mov dx,0  ;设置被除次数计数
     mov al,[si];设置被除的数
     mov ah,0
first:  ;单个数字输出，每次除10保留余数，每次输出一个余数
     div bl; 余在ah,商在al
     push ax
     inc dx
     cmp al,0  ;判断是否除尽
     mov ah,0
     jnz first

second:
     pop ax
     or ah,30H
     dec dx
     push dx  ;压入保存
     mov dl,ah  ;赋值准备打印
     mov ah,2
     int 21h
     pop dx  ;弹出
     cmp dx,0 ;判断当前的数字位数是否打完
     jnz second

```

```
next:
    call space
    inc si;
    loop initial

    call newline
    pop si
    pop dx
    pop bx
    pop ax
    pop cx
    ret
print endp

;-------------------------------------------------
;子程序名：space
;功能：打印空格
;占用寄存器：ah, dl
;-------------------------------------------------
space proc near
     push ax
     push dx
     mov  dl,20h
     mov ah,2
     int 21h
     pop dx
     pop ax
   ret
space endp
```

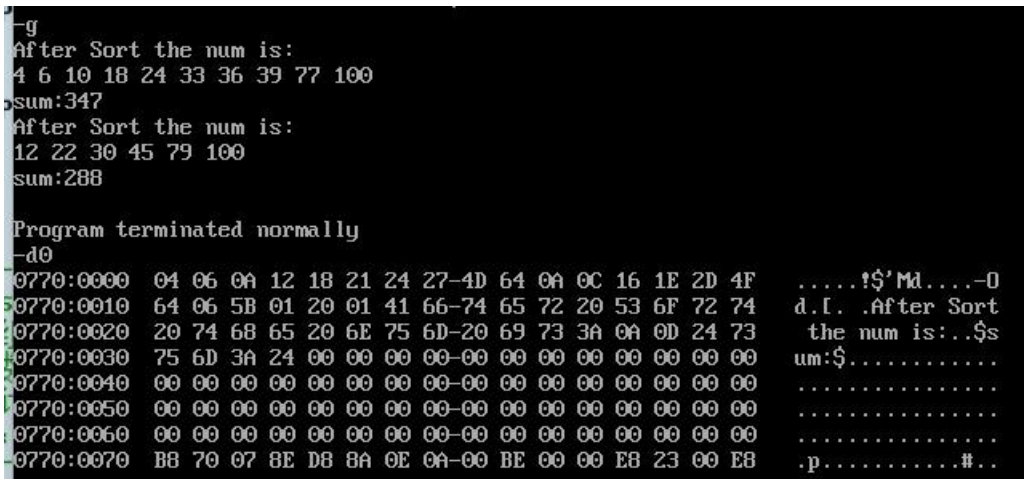
```
;-------------------------------------------------
;子程序名：newline
;功能：打印换行符
;占用寄存器：ah, dl
;-------------------------------------------------
newline proc near
      push ax
      push dx
      mov dl,0Ah
      mov ah,2
      int 21h
      pop dx
      pop ax
     ret
newline endp

CODES ENDS
END MAIN
```

3 调试过程与结果

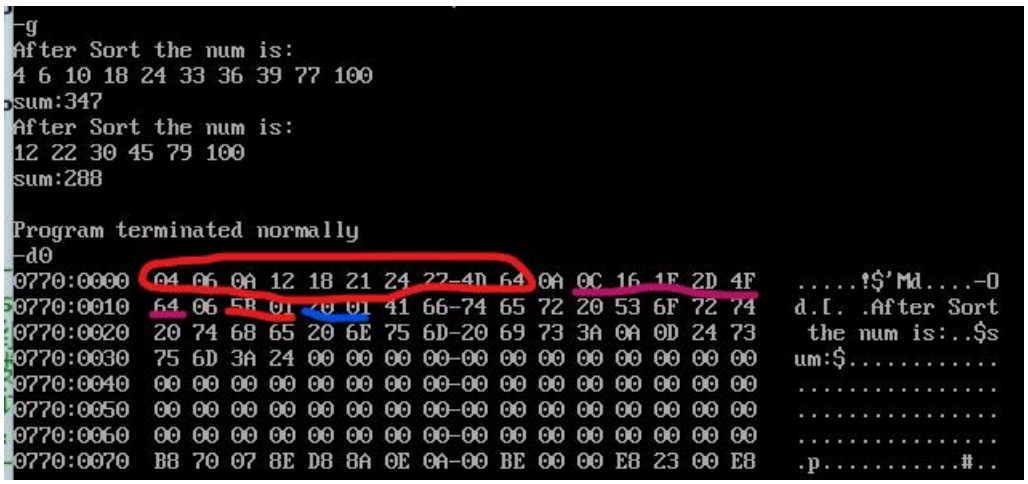
通过图文方式展示调试过程及结果。
运行结果：

```
After Sort the num is:
4 6 10 18 24 33 36 39 77 100
sum:347
After Sort the num is:
12 22 30 45 79 100
sum:288



            Press any key to continue
```

调试过程：
执行程序后，查看地址数据。



红色圈出的是数组 1 重新排序后的存储情况。紫色划出的是数组 2 重新排序后的存储情况。
红色划出的是 result1，十六进制为 0158H
蓝色划出的是 result2，十六进制为 0120H
结果与输出一致。



选做题

1 问题描述

编写进制转换子程序 DEC2BIN 和 DEC2OCT，把输入的一个字类型的正整数分别用二进制和八进制显示出来。
要求：

从屏幕输入十进制数（最大 65535），输入回车键开始转换，调用进制转换子程序，并将转换后的二进制和八进制数显示在屏幕上。

2 解决方案及代码

解题思路:

首先考虑得到从键盘输入的数字，考虑用 dos 调用的 1 号功能单个字符的录入，然后将输入的字符链接成一个整数并判断是否是否溢出以及为非数字符号。出错则报错并让其重新输入。通过键入回车判断输入结束。

二进制和八进制在除后取余和移位中选择后者实现，即二进制输出每次将数字循环移位一位，通过每次取最后一位输出，实现。八进制同理，只是 16 位首先需要移位一位将首位输出，剩下 15 位，每次循环移位三位，输出末尾三位实现。

代码:

```
DATAS SEGMENT
    string_1 db 'Please input the number(0-65535):','$';此处输入数据段代码
    string_2 db 'Wrong Input!Please input again!',0ah,0dh,'$'
    string_3 db 'binary:$'
    string_4 db 'octal:$'
    newline1 db 0ah,0dh,'$';用内存段实现换行也会很方便
DATAS ENDS

STACKS SEGMENT
    dw 40 dup(?);此处输入堆栈段代码
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
    mov dx ,offset string_1
    mov ah,9
    int 21h
    call input
    mov dx,offset string_3
    int 21h
    call DEC2BIN
    mov dx,offset string_4
    int 21h
    call DEC2OCT
    MOV AH,4CH
    INT 21H
```

```
;-------------------------------------------------
;子程序名：input
;功能：从键盘录入数据
;出口参数：录入的数存于BX
;占用寄存器：AX, BX, CX, DX, FLAG
;-------------------------------------------------

input proc near
      push ax
      ;push bx
      push cx
      push dx
      mov bx,0
      mov cl,10
      mov ch,0
      mov dx,0
LOOP0:

     mov ah,01
     int 21h
     cmp al,0Dh
     je End_

     sub al,30h
     jl error_
     cmp al,9
     jg error_

     ;拼接
     mov ah,0; 高位置零
     xchg ax,bx ;            al（新的数）+10*原来的ax
     mul  cx    ;            bx一开始为0，故第一次原来的ax为0
     jc   error_
     add  bx,ax ;            将ax的计算结果保护在bx中通过xchg取回乘10
     jc   error_;            判断是否字溢出
     jmp LOOP0

error_:
     mov dx, 0
     mov bx, 0
     call newline    ; 换行
     call ERROR  ; 输出错误提示
```

```
      jmp  LOOP0

End_:
      pop dx
      pop cx
      ;pop bx
      pop ax
      ret
input endp
```

```
;采用循环移位的方式，将最高位不停的传到最低位，输出
;也可以类似19进制输出，除2输出余数
;-------------------------------------------------
;子程序名：input
;功能：从键盘录入数据
;出口参数：录入的数存于BX
;占用寄存器：AX, BX, CX, DX, FLAG
;-------------------------------------------------

DEC2BIN proc near
      push bx ;保存之前的结果
      push ax
      push cx
      push dx
      mov ch,0
      mov cl,16;只用循环16次


loop1:
      rol bx,1
      mov dx,bx
      and dx,1
      or  dx,30H
      mov ah,2
      int 21h
      loop loop1
      call newline
      pop dx
      pop cx
      pop ax
      pop bx
      ret
DEC2BIN endp
```

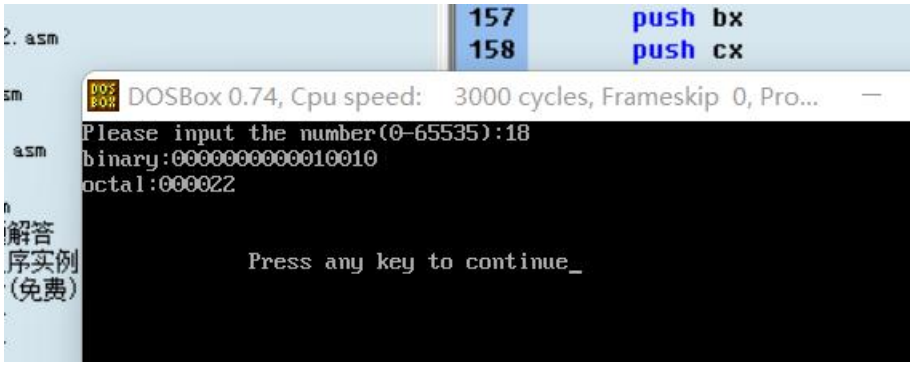
```
;循环移位，每次3位,余最高位单独处理，首先左移一位
DEC2OCT proc near
      push bx
      push ax
      push cx
      push dx
      ;先输出首位
      rol bx,1
      mov dx,bx
      and dx,1
      or  dx,30H
      mov ah,2
      int 21h

      mov ch,0
      mov cl,5
loop1:
      push cx   ;由于loop需要cx，且rol也要使用cl,先将移位次数存于栈中
      mov cl,3
      rol bx,cl
      mov dx,bx
      and dx,07h
      or  dx,30h
      mov ah,2
      int 21h
      pop cx   ;取出移位次数
      loop loop1


      pop dx
      pop cx
      pop ax
      pop bx
      ret

DEC2OCT endp
```

```
154   ;简单的报错子程序
155   ERROR proc near
156         push ax
157         push bx
158         push cx
159         push dx
160         mov dx,offset string_2
161         mov ah,9
162         int 21h
163         mov dx ,offset string_1
164         mov ah,9
165         int 21h
166         pop dx
167         pop cx
168         pop bx
169         pop ax
170         ret
171   ERROR endp
172   ;换行子程序
173   newline proc near
174         push ax
175         push dx
176         mov dl,0Ah
177         mov ah,2
178         int 21h
179         pop dx
180         pop ax
181        ret
182   newline endp
183
184
185
186   CODES ENDS
187       END START
```

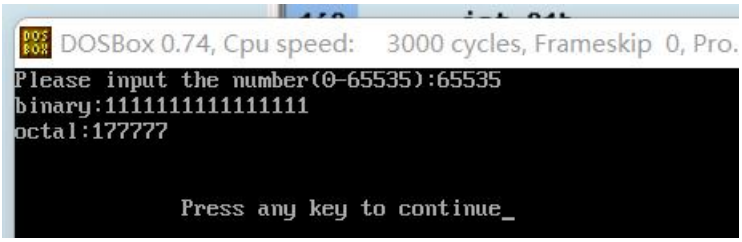
3 调试过程与结果

结果:

输入 18

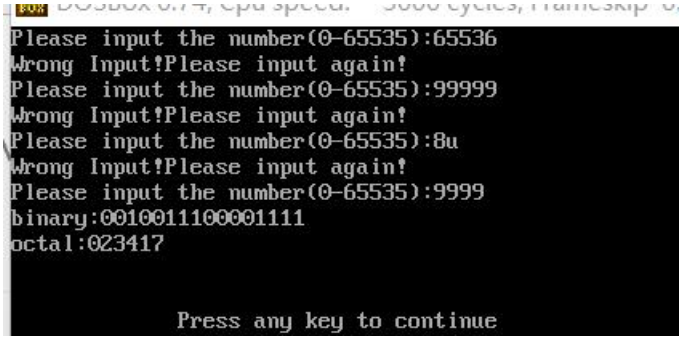


输入 65535



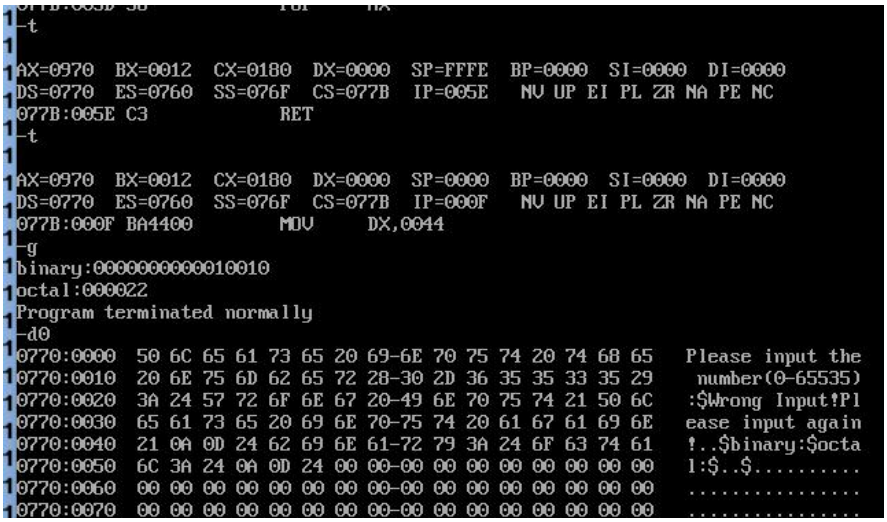
错误处理：

输入 65536；99999，输入 8u



调试过程：

选用 18 时的调试过程，由于没有在内存中存入数据，数据段没有变化



总结

实验总结与心得体会：

在本次实验中，我们主要学习了对子程序的设计，这次实验尤其让我对用汇编输出至屏幕印象深刻，由于寄存器数量的限制，让一个简单的数组输出的语句都尤为长，甚至超过了排序；然后汇编中一定要注意子程序对寄存器的占用，这会在堆栈使用等各方面影响程序。本次实验大大提高了我对子程序设计和一些常用汇编指令的理解，受益匪浅。