

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

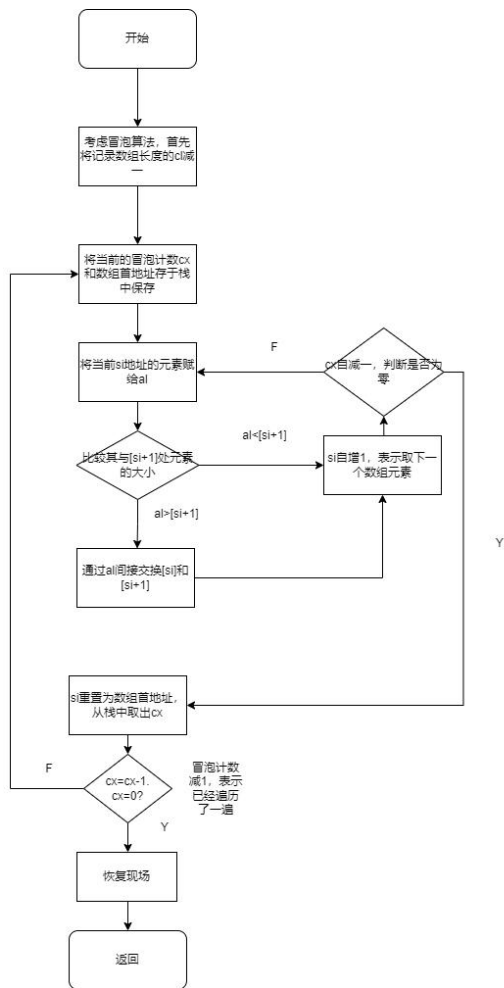
解题思路：

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：

```
1  DATAS SEGMENT
2      num1 db 10,6,24,39,33,77,18,100,36,4 ;数组
3      n1 db 10 ;数字个数
4      ;num1 db 10,6
5      ;n1 db 2
6      num2 db 45,12,30,22,100,79 ;数组
7      n2 db 6 ;数字个数
8      result1 dw ? ;数组和
9      result2 dw ? ;数组和
10     string db 'After Sort the num is:',0ah,0dh,'$','$终止符
11     string1 db 'sum:$'
12     ;此处输入数据段代码
13 DATAS ENDS
14
15 STACKS SEGMENT
16     db 40 dup(0) ;此处输入堆栈段代码
17 STACKS ENDS
18
19 CODES SEGMENT
20 main proc far
21     ASSUME CS:CODES,DS:DATAS,SS:STACKS
22     MOV AX,DATAS
23     MOV DS,AX
24     ;此处输入代码段代码
25     mov cl,n1 ;传入数组数量参数
26     mov si,offset num1 ;传入数组初始地址
27     call sort ;调用子程序
28     call print
29     call sum
30     mov result1,bx
31
32
33
34     mov cl,n2
35     mov si,offset num2
36     call sort
37     call print
38     call sum
39     mov result2,bx
40
41
42     MOV AH,4CH
```

```
43     INT 21H
44     ret
45 main endp
46
47 ;-----
48 ;子程序名: sort
49 ;功能: 通过冒泡排序对给定的数组进行排序
50 ;入口参数: 数组偏移地址、数组元素个数
51 ;出口参数: 重新排序的数字存于原来的内存地址
52 ;占用寄存器: AX, BX, CX, DX, SI, FLAG
53 ;-----
54
55 sort proc near
56     mov ch,0
57     push cx
58     dec cl
59 OUTER:
60     push cx ;保存当前冒泡的计数
61     push si ;存入数组首地址
62     INNER:
63         mov al,[si]
64         cmp al,[si+1] ; 判断IF (NUM[SI] > NUM[SI+1])
65         JLE UPDATE ; 不是跳转
66         XCHG al,[si+1] ; 是交换,SWAP (NUM[SI], NUM[SI+1])
67         XCHG al,[si]
68     UPDATE:
69         inc si
70         LOOP INNER ; CONTINUE
71     pop si
72     pop cx
73     LOOP OUTER
74     pop cx
75     ret
76 sort endp
77
78
```

```

79 ;-----
80 ;子程序名: sum
81 ;功能: 对数组求和并输出结果至屏幕上
82 ;入口参数: 数组偏移地址、数组元素个数
83 ;出口参数: 计算结果存于BX
84 ;占用寄存器: AX, BX, CX, DX, SI, FLAG
85 ;-----
86
87 sum proc near
88     mov ch,0
89     push cx
90     mov bx,0
91 LOOP1:
92     add bl,[si]
93     adc bh,0
94     inc si
95     LOOP LOOP1
96     push bx;保存结果
97     ;****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
98     mov dx,offset string1;打印结果提示
99     mov ah,9
100    int 21h
101    mov ax,bx
102    mov bx,10
103 DIV_:
104    mov dx,0
105    div bx ;商在ax,余在dx
106    push dx
107    inc cx
108    cmp ax,0
109    jnz DIV_
110
111 do:
112    pop dx
113    or dx,30h
114    mov ah,2
115    int 21h
116    loop do
117    call newline
118    pop bx
119    pop cx
120    ret

```

```

121     sum endp
122 ;-----
123 ;子程序名: print
124 ;功能: 将数组元素输出到屏幕上
125 ;入口参数: 数组偏移地址、数组元素个数
126 ;占用寄存器: AX, BX, CX, DX, SI, FLAG
127 ;-----
128
129 print proc near
130     push cx
131     push ax
132     push bx
133     push dx,
134     push si;
135     mov dx,offset string;打印数组提示
136     mov ah,9
137     int 21h
138     mov bl,10;设置被除数
139 initial:
140     mov dx,0 ;设置被除次数计数
141     mov al,[si];设置被除的数
142     mov ah,0
143 first: ;单个数字输出, 每次除10保留余数, 每次输出一个余数
144     div bl; 余在ah,商在al
145     push ax
146     inc dx
147     cmp al,0 ;判断是否除尽
148     mov ah,0
149     jnz first
150
151 second:
152     pop ax
153     or ah,30H
154     dec dx
155     push dx ;压入保存
156     mov dl,ah ;赋值准备打印
157     mov ah,2
158     int 21h
159     pop dx ;弹出
160     cmp dx,0 ;判断当前的数字位数是否打完
161     jnz second
162

```

```

164 next:
165     call space
166     inc si;
167     loop initial
168
169     call newline
170     pop si
171     pop dx
172     pop bx
173     pop ax
174     pop cx
175     ret
176 print endp
177
178 ;-----
179 ;子程序名: space
180 ;功能: 打印空格
181 ;占用寄存器: ah, dl
182 ;-----
183 space proc near
184     push ax
185     push dx
186     mov dl,20h
187     mov ah,2
188     int 21h
189     pop dx
190     pop ax
191     ret
192 space endp

```

```

193 ;-----
194 ;子程序名: newline
195 ;功能: 打印换行符
196 ;占用寄存器: ah, dl
197 ;-----
198 newline proc near
199     push ax
200     push dx
201     mov dl,0Ah
202     mov ah,2
203     int 21h
204     pop dx
205     pop ax
206     ret
207 newline endp
208
209 CODES ENDS
210 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。

```
-g
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

Program terminated normally
-d0
0770:0000 04 06 0A 12 18 21 24 27-4D 64 0A 0C 16 1E 2D 4F .....!$'Md....-0
0770:0010 64 06 5B 01 20 01 41 66-74 65 72 20 53 6F 72 74 d.l. .After Sort
0770:0020 20 74 68 65 20 6E 75 6D-20 69 73 3A 0A 0D 24 73 the num is:..$s
0770:0030 75 6D 3A 24 00 00 00 00-00 00 00 00 00 00 00 00 um:$.....
0770:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0070 B8 70 07 8E D8 8A 0E 0A-00 BE 00 00 E8 23 00 E8 .p.....#..
```

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。

```
-g
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

Program terminated normally
-d0
0770:0000 04 06 0A 12 18 21 24 27-4D 64 0A 0C 16 1E 2D 4F .....!$'Md....-0
0770:0010 64 06 5B 01 20 01 41 66-74 65 72 20 53 6F 72 74 d.l. .After Sort
0770:0020 20 74 68 65 20 6E 75 6D-20 69 73 3A 0A 0D 24 73 the num is:..$s
0770:0030 75 6D 3A 24 00 00 00 00-00 00 00 00 00 00 00 00 um:$.....
0770:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0770:0070 B8 70 07 8E D8 8A 0E 0A-00 BE 00 00 E8 23 00 E8 .p.....#..
```

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：

```
1  DATAS SEGMENT
2      string_1 db 'Please input the number(0-65535):','$';此处输入数据段代码
3      string_2 db 'Wrong Input!Please input again!','0ah,0dh','$'
4      string_3 db 'binary:$'
5      string_4 db 'octal:$'
6      newline1 db 0ah,0dh,'$';用内存段实现换行也会很方便
7  DATAS ENDS
8
9  STACKS SEGMENT
10     dw 40 dup(?);此处输入堆栈段代码
11 STACKS ENDS
12
13 CODES SEGMENT
14     ASSUME CS:CODES,DS:DATAS,SS:STACKS
15 START:
16     MOV AX,DATAS
17     MOV DS,AX
18     ;此处输入代码段代码
19     mov dx,offset string_1
20     mov ah,9
21     int 21h
22     call input
23     mov dx,offset string_3
24     int 21h
25     call DEC2BIN
26     mov dx,offset string_4
27     int 21h
28     call DEC2OCT
29     MOV AH,4CH
30     INT 21H
```

```

31 ;-----
32 ;子程序名: input
33 ;功能: 从键盘录入数据
34 ;出口参数: 录入的数存于BX
35 ;占用寄存器: AX, BX, CX, DX, FLAG
36 ;-----
37
38 input proc near
39     push ax
40     ;push bx
41     push cx
42     push dx
43     mov bx, 0
44     mov cl, 10
45     mov ch, 0
46     mov dx, 0
47 LOOP0:
48
49     mov ah, 01
50     int 21h
51     cmp al, 0Dh
52     je End_
53
54     sub al, 30h
55     jl error_
56     cmp al, 9
57     jg error_
58
59     ;拼接
60     mov ah, 0; 高位置零
61     xchg ax, bx ;          a1 (新的数) + 10 * 原来的ax
62     mul cx ;          bx一开始为0, 故第一次原来的ax为0
63     jc error_
64     add bx, ax ;          将ax的计算结果保护在bx中通过xchg取回乘10
65     jc error_ ;          判断是否溢出
66     jmp LOOP0
67
68 error_:
69     mov dx, 0
70     mov bx, 0
71     call newline ; 换行
72     call ERROR ; 输出错误提示

```

```

73     jmp LOOP0
74
75 End_:
76     pop dx
77     pop cx
78     ;pop bx
79     pop ax
80     ret
81 input endp
82

```

```

83 ;采用循环移位的方式，将最高位不停的传到最低位，输出
84 ;也可以类似19进制输出，除2输出余数
85 ;-----
86 ;子程序名: input
87 ;功能: 从键盘录入数据
88 ;出口参数: 录入的数存于BX
89 ;占用寄存器: AX, BX, CX, DX, FLAG
90 ;-----
91
92 DEC2BIN proc near
93     push bx ;保存之前的结果
94     push ax
95     push cx
96     push dx
97     mov ch,0
98     mov cl,16;只用循环16次
99
100
101     loop1:
102         rol bx,1
103         mov dx,bx
104         and dx,1
105         or  dx,30H
106         mov ah,2
107         int 21h
108         loop loop1
109         call newline
110         pop dx
111         pop cx
112         pop ax
113         pop bx
114         ret
115 DEC2BIN endp

```

```

117 ;循环移位，每次3位,余最高位单独处理，首先左移一位
118 DEC2OCT proc near
119     push bx
120     push ax
121     push cx
122     push dx
123     ;先输出首位
124     rol bx,1
125     mov dx,bx
126     and dx,1
127     or  dx,30H
128     mov ah,2
129     int 21h
130
131     mov ch,0
132     mov cl,5
133     loop1:
134         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
135         mov cl,3
136         rol bx,cl
137         mov dx,bx
138         and dx,07h
139         or  dx,30h
140         mov ah,2
141         int 21h
142         pop cx ;取出移位次数
143         loop loop1
144
145
146     pop dx
147     pop cx
148     pop ax
149     pop bx
150     ret
151
152 DEC2OCT endp

```

```

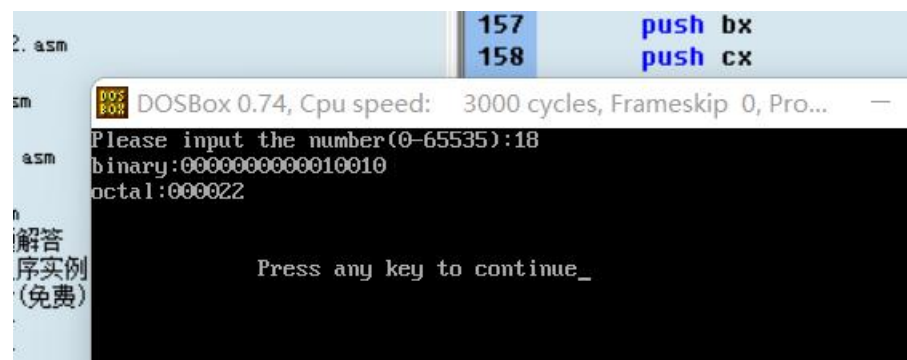
154 ;简单的报错子程序
155 ERROR proc near
156     push ax
157     push bx
158     push cx
159     push dx
160     mov dx,offset string_2
161     mov ah,9
162     int 21h
163     mov dx,offset string_1
164     mov ah,9
165     int 21h
166     pop dx
167     pop cx
168     pop bx
169     pop ax
170     ret
171 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

```
DOS FOR DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro.
Please input the number(0-65535):65535
binary:1111111111111111
octal:177777

Press any key to continue_
```

错误处理:

输入 65536; 99999, 输入 8u

```
DOS FOR DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro.
Please input the number(0-65535):65536
Wrong Input!Please input again!
Please input the number(0-65535):99999
Wrong Input!Please input again!
Please input the number(0-65535):8u
Wrong Input!Please input again!
Please input the number(0-65535):9999
binary:0010011100001111
octal:023417

Press any key to continue
```

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

```
077B:005E C3 RET
-t
AX=0970 BX=0012 CX=0180 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=077B IP=005E NU UP EI PL ZR NA PE NC
077B:005E C3 RET
-t
AX=0970 BX=0012 CX=0180 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=077B IP=000F NU UP EI PL ZR NA PE NC
077B:000F BA4400 MOV DX,0044
-g
binary:0000000000010010
octal:000022
Program terminated normally
-d0
0770:0000 50 6C 65 61 73 65 20 69-6E 70 75 74 20 74 68 65 Please input the
0770:0010 20 6E 75 6D 62 65 72 28-30 2D 36 35 35 33 35 29 number(0-65535)
0770:0020 3A 24 57 72 6F 6E 67 20-49 6E 70 75 74 21 50 6C :Wrong Input!Pl
0770:0030 65 61 73 65 20 69 6E 70-75 74 20 61 67 61 69 6E ease input again
0770:0040 21 0A 0D 24 62 69 6E 61-72 79 3A 24 6F 63 74 61 !..$binary:$octa
0770:0050 6C 3A 24 0A 0D 24 00 00-00 00 00 00 00 00 00 l:$..$.
0770:0060 00 00 00 00 00 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 .....
```

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。