

北京科技大学实验报告

学院：计通学院

专业：计算机科学与技术

班级：计 173

姓名：张宝丰

学号：41724081

实验日期：2019 年 12 月 6 日

实验名称：汇编预习作业

实验目的：通过自学汇编语言编程，完成预习报告，为实验课奠定基础。

实验要求：通过 CG 系统抽取预习题目，在到实验室上课前完成预习报告。

实验内容：

1、简答题

题目 1：请分别用三种不同的汇编编程语句，每种仅使用一条语句，实现以下功能：

使 AX=0

答：

1. mov ax,0

2. xor ax,ax

3. sub ax,ax

2、编程题

要求：按题目中的要求完成 2 道编程练习题。

所用编程工具：vscode、dosbox、emu8086

编程题 1：用汇编语言编写程序，计算 1-150 的和，并将结果以十进制的表示显示在屏幕上。

1) 程序源代码

```
; 将 data 段中的数据以十进制的形式显示出来
; 首先计算 1-150 的和，放入 ax
; 调用 dtoc，将 ax 转化十进制数的 ASCII 码，放入 buffer
; 调用 show_str，将结果以绿色字体输出
assume cs:code, ds:data, ss:stack
data segment
    buffer dw 10 dup (0)
data ends
```

```

stack segment
    dw 256 dup (0)
stack ends

code segment
start:
    ; 初始化
    mov bx,data
    mov ds,bx
    mov si,0
    mov bx,stack
    mov ss,bx
    mov cx,150
    ; 计算 1-150 的和, 结果放入 ax
cal_ax:
    add ax,cx
    loop cal_ax
    call dtoc

    mov dh,8
    mov dl,3
    mov cl,2
    mov si,0
    call show_str

    mov ax,4c00h
    int 21h

dtoc:
    ; 逐次取余数, 并入栈, 最后再倒序拿出来, 在 buffer 最后放 0
    push ax
    push di
    push cx
    push dx
    push si
    mov di,0          ; 记录入栈次数

dtoc_s1:
    mov cx,10
    mov dx,0
    div cx

    mov cx,ax          ; 如果商为 0, 则求值完成
    jcxz dtoc_s2

```

```

    add dx,30h
    push dx          ; 求得的 ASCII 码入栈
    inc di
    jmp dtoc_s1

dtoc_s2:
    add dx,30h
    push dx
    inc di
    mov cx,di        ; cx 为转化后的字符串长度
dtoc_s3:
    pop ax
    mov [si],al
    inc si
    loop dtoc_s3     ; 将 ASCII 码出栈
    mov al,0
    mov [si],al      ; 最后一位放 0（虽然本来初始化的时候就已经是 0 了）

    pop si
    pop dx
    pop cx
    pop di
    pop ax
    ret

show_str:
    push cx
    push dx
    mov ax,0b800h
    mov es,ax        ; 用 es 储存显存段地址
    ; 计算行偏移
    sub dh,1         ; dh-1
    mov bl,10
    mov al,dh
    mul bl            ; 计算出起始行位置，存在 ax 中
    mov bl,16
    mul bl
    mov bp,ax        ; 行地址暂存在 bp 中
    ; 计算列偏移
    sub dl,1
    mov al,2
    mul dl
    add ax,bp
    mov di,ax        ; 显示偏移地址放到 di 中

```

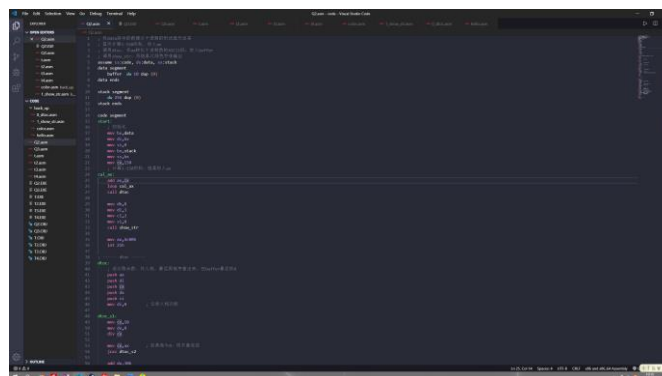
```

    mov bp,0      ; 清空 bp
    mov bl,cl      ; 字体颜色放到 bl 中
show:
    mov al,ds:[si] ; 把字符串中的一个字节放到 al
    mov ah,0
    mov cx,ax      ; 把每个数据都放到 cx 中检验是否为 0
    jcxz ok
    mov byte ptr es:[di],al ; 显示字符
    mov byte ptr es:[di+1],bl ; 显示颜色
    add di,2      ; 显存指针每次+2
    inc si        ; 字符串指针每次+1
    jmp show
ok:
    pop dx
    pop cx
    ret

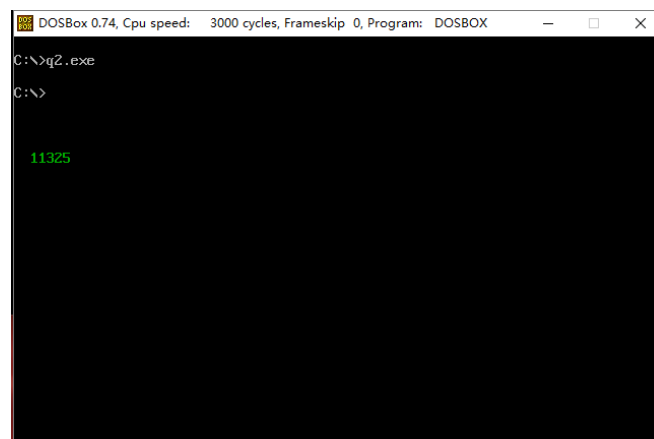
code ends
end start

```

2) 程序源码的屏幕截图



3) 程序运行结果的屏幕截图



编程题 2: 用汇编语言编写程序，实现 10 个无符号数自大到小排序的功能，即，从屏幕上输入 10 个无符号数，将排序结果显示在屏幕上。（注：我扩展了程序，首先输入一个 N，指明要对多少个数进行排序）

1) 程序源代码

```
; 第一行输入 N
; 第二行开始的 N 行，每行输入一个小于 256 的正整数
; 排序后输出结果
assume cs:code,ds:data,ss:stack
data segment
    buf      db      200 dup(0)
    n        db      0
    outbuf   db      200 dup(0)
    hello    db      "hello world!$"
data ends

stack segment
    dw 128 dup (0)
stack ends

code segment
; TODO: 排序，输出
start:
    ; 初始化
    mov ax,data
    mov ds,ax
    mov sp,128
    call getInt      ; 读入 N
    mov n,al
    mov bx,0
    mov ch,0
    mov cl,al
read_n_nums:
    ; 读入 n 个数
    push cx
    call getInt
    mov buf[bx],al
    inc bx
    pop cx
    loop read_n_nums
    mov bx,0
```

```

lp1:
    mov di,bx
    inc di
lp2:
    mov al,buf[bx]
    mov cl,buf[di]
    cmp al,cl
        jae els
    mov buf[bx],cl
    mov buf[di],al
els:
    inc di
    mov ax,di
    mov ah,n
    cmp ah,al
    jae lp2
    inc bx
    cmp bl,n
    jb lp1

    mov ch,0
    mov cl,n
    mov bx,0

output:
    mov ah,0
    mov al,buf[bx]
    call putInt
    mov al,' '
    call putchar
    inc bx
    cmp bx,cx
    jb output

    mov ah,4ch
    int 21h

putchar:
    ; 把 dx 地址中的字符输出
    push dx
    mov dx,ax
    mov ah,02h
    int 21h

```

```
pop dx
ret
```

```
getInt:
```

```
    push bx
    push cx
    push dx
    mov ax,0
    mov bx,0
    mov cx,0
    mov dx,0
```

```
getInt_getchar:
```

```
    ; 读入一个字符
    mov ah,01h
    int 21h
    cmp al,0dh      ; 判断是否为回车符
    jz  getInt_ret  ; 判断是回车符，跳转出
```

```
    cmp al,30h      ; <'0'
    jb getInt_getchar
    cmp al,39h      ; >'0'<='9'
    jbe getInt_SumToAx
    jmp getInt_getchar
```

```
getInt_SumToAx:
```

```
    mov ah,0
    push ax

    mov ax,cx      ; ax = ax*10 + al
    mov bx,10
    mul bx
    mov dx,0
    mov cx,ax
    pop ax
```

```
    sub al,30h     ; ascii 码需要-30h 才是数字
    add cx,ax
    jmp getInt_getchar
```

```
getInt_ret:
```

```
    mov ax,cx
    pop dx
    pop cx
    pop bx
    ret
```

putInt:

```
    push si
    push dx
    mov dx,0
    mov si,offset outbuf
    call dtoc
    mov dx,offset outbuf
    mov ah,9
    int 21h
    pop dx
    pop si
    ret
```

dtoc:

```
    ; 逐次取余数，并入栈，最后再倒序拿出来，在 buffer 最后放 0
    push ax
    push di
    push cx
    push dx
    push si
    mov di,0          ; 记录入栈次数
```

dtoc_s1:

```
    mov cx,10
    mov dx,0
    div cx

    mov cx,ax          ; 如果商为 0，则求值完成
    jcxz dtoc_s2

    add dx,30h
    push dx             ; 求得的 ASCII 码入栈
    inc di
    jmp dtoc_s1
```

dtoc_s2:

```
    add dx,30h
    push dx
    inc di
    mov cx,di          ; cx 为转化后的字符串长度
```

dtoc_s3:

```
    pop ax
    mov [si],al
```



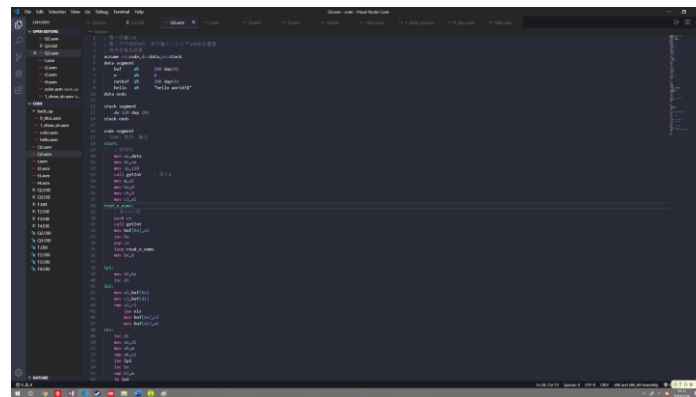
```

inc si
    loop dtoc_s3    ; 将 ASCII 码出栈
    mov al,"$"
    mov [si],al     ; 最后一位放$（虽然本来初始化的时候就已经是 0 了）
    pop si
    pop dx
    pop cx
    pop di
    pop ax
    ret

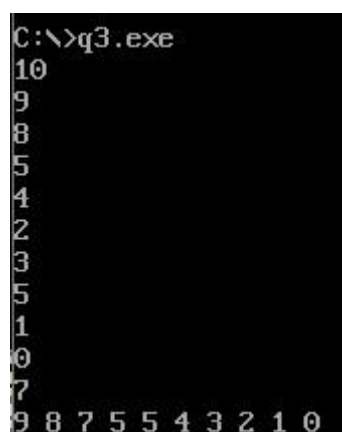
code ends
end start

```

2) 程序源码的屏幕截图



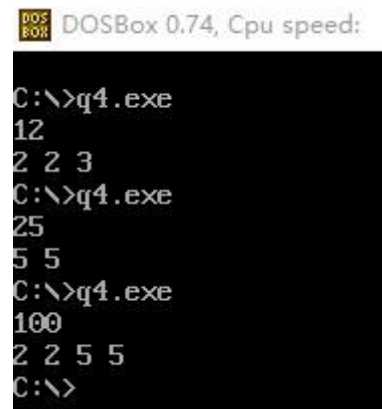
3) 程序运行结果的屏幕截图



学长挑战题：从键盘输入一个整数（不超过 127），要求对它进行质因数分解，数字间用空格隔开。如输入 12，则输出应为： 2 2 3

考点：键盘输入、屏幕输出、整数-字符串转换、除法、循环、子程序

运行结果：



```
DOS
BOX
DOSBox 0.74, Cpu speed:
C:\>q4.exe
12
2 2 3
C:\>q4.exe
25
5 5
C:\>q4.exe
100
2 2 5 5
C:\>
```

代码：

```
assume cs:code,ds:data,ss:stack
data segment
    n        db        0
    buf       db        200 dup(0)
    outbuf    db        200 dup(0)
data ends

stack segment
    dw 128 dup (0)
stack ends

code segment
start:
    mov bx,data
    mov ds,bx
    mov si,0
    mov bx,stack
    mov ss,bx
    mov bx,0
    call getInt    ; 读入 N 到 al 中
    mov n,al
    mov ah,0
    mov cx,2        ; 质因数从 2 开始
calNext:            ; 计算 ax 的下一个质因数
    push ax
    cmp ax,1
```

jbe calNext_end ; 若 ax<=1, 则直接返回

```
div cl          ; 商在 al 中, 余数在 ah 中, 若余数为 0, 说明是质因数
cmp ah,0
jne calNext_else
```

calNext_if: ; if ah==0

```
mov ax,cx
call putInt
mov ah,0
mov al,' '
call putchar
```

```
pop ax
div cl          ; ax = ax/cl
mov ah,0
jmp calNext
```

calNext_else: ; 找到下一个质数, 放到 cl 中

```
inc cx
mov ax,cx
call isPrime
cmp ax,0
je calNext_else
```

```
pop ax
jmp calNext
```

calNext_end:

```
mov ax,4c00h
int 21h
```

isPrime: ; 检验 ax 中的数值是否为质数, 如果是质数, 返回 ax=1

```
push bx
push cx
push dx
mov bx,2
```

isPrime_loop:

```
push ax
div bl
cmp ah,0 ; 若余数为 0, 说明整除, 返回 ax = 0
je isPrime_falseRet
inc bx
pop ax
cmp ax,bx
je isPrime_trueRet
```

```
jmp isPrime_loop
isPrime_falseRet:
    pop ax
    mov ax,0
    jmp isPrime_ret
isPrime_trueRet:
    mov ax,1
    jmp isPrime_ret
isPrime_ret:
    pop dx
    pop cx
    pop bx
    ret
```

```
putchar:                ; 把 dl 中的字符输出
    push ax
    push dx
    mov dx,ax
    mov ah,02h
    int 21h
    pop dx
    pop ax
    ret
```

```
putInt:                ; 将 ax 中的 16 位整数输出到屏幕上
    push si
    push dx
    mov dx,0
    mov si,offset outbuf
    call dtoc
    mov dx,offset outbuf
    mov ah,9
    int 21h
    pop dx
    pop si
    ret
```

```
getInt:
    push bx
    push cx
    push dx
    mov ax,0
```

```

    mov bx,0
    mov cx,0
    mov dx,0
getInt_getchar:
    ; 读入一个字符
    mov ah,01h
    int 21h
    cmp al,0dh      ; 判断是否为回车符
    jz  getInt_ret  ; 判断是回车符, 跳转出

    cmp al,30h      ; <'0'
    jb  getInt_getchar
    cmp al,39h      ; >'0'<='9'
    jbe getInt_SumToAx
    jmp  getInt_getchar
getInt_SumToAx:
    mov ah,0
    push ax

    mov ax,cx      ; ax = ax*10 + al
    mov bx,10
    mul bx
    mov dx,0
    mov cx,ax
    pop ax

    sub al,30h     ; ascii 码需要-30h 才是数字
    add cx,ax
    jmp  getInt_getchar
getInt_ret:
    mov ax,cx
    pop dx
    pop cx
    pop bx
    ret

dtoc:
    ; 逐次取余数, 并入栈, 最后再倒序拿出来, 在 buffer 最后放 0
    push ax
    push di
    push cx
    push dx
    push si

```

```

    mov di,0          ; 记录入栈次数

dtoc_s1:
    mov cx,10
    mov dx,0
    div cx

    mov cx,ax          ; 如果商为 0，则求值完成
    jcxz dtoc_s2

    add dx,30h
    push dx            ; 求得的 ASCII 码入栈
    inc di
    jmp dtoc_s1

dtoc_s2:
    add dx,30h
    push dx
    inc di
    mov cx,di          ; cx 为转化后的字符串长度
dtoc_s3:
    pop ax
    mov [si],al
    inc si
    loop dtoc_s3       ; 将 ASCII 码出栈
    mov al,"$"
    mov [si],al        ; 最后一位放$（虽然本来初始化的时候就已经是 0 了）
    pop si
    pop dx
    pop cx
    pop di
    pop ax
    ret

code ends
end start

```

实验思考与分析：

本次实验我使用的是 windows10 下的 dosbox 运行程序。汇编程序的扩展名应当是 .asm(assembly)，但实际编译时编译器不对扩展名做出要求，这应当是因为实际的文件都是二进制字符流文件（比如 .txt 和各种语言的代码文件），它是没有结构的（区别于 word、excel 等），所以可以在多个平台下被读出。

第一个编程题要求出 1-150 的加和并将结果显示输出。这给我的第一个挑战是：**如何将计算的结果输出到屏幕上**？在这里，我参考了王爽的《汇编语言》，利用了其中提到的“**显示缓冲区**”，即内存地址空间中 B80000H-BFFFFH 共 32KB 的空间是 80×25 彩色字符模式的显示缓冲区，向这个空间写入数据，写入的内容将立即出现在显示器上。缓冲区分为 8 页，每页 4KB，显示器默认显示第 0 页的内容，也就是 B8000H-B8F9FH 的内容。每个字符在显示缓冲区中要占两个字节，其中偶字节为字符的 ASCII 码，与之相邻的后一个奇字节是它的颜色信息。如此，我可以将内存中的一个字符串输出到屏幕的任意位置，我给出的程序中实现了输出到第 9 行第 3 列。第二个挑战是：**如何将计算结果的整数值转化为 ASCII 码的字符串**？一般的想法是利用**除法取余数，每次将余数放入栈中，最后再倒序取出**，程序中的 dtoc 函数实现了这个过程。这里用到了除法指令 div，我只考虑了 16 位被除数的情况，此时被除数默认在 ax 中，除数是传入的 8 位寄存器或内存单元的值，商在 al 中，余数在 ah 中。

对于第二个编程题，我提高了一下要求：从键盘输入一个整数 N，接下来 N 行每行输入一个整数，要求对这 N 个整数进行从大到小排序，并将结果输出。首先要处理的问题是处理键盘的读入，经查阅资料，得知 int 21h 中断以来寄存器 ah 中不同的值，有如下的基本功能：

功能号（ah 中的值）	功能描述
1	键盘输入，从 stdin 输入一个字符给 al
2	打印输出，将 dl 中的字符输出到 stdout
9	输出 ds:dx 指向的以“\$”结尾的字符串
0ah	将字符串读入到 ds:dx 指向的缓冲区

在程序中，我在 getInt 函数中调用 1 号中断，逐个获取输入的字符，以回车字符分隔整数；在 sort 函数中，我对 10 个数字进行冒泡排序；在 putInt 函数中，我调用 dtoc 将整数转化为字符，再调用 5 号中断完成输出。

如上，我们拥有了 3 个基本功能的汇编实现：读入字符与数字，整数转化为字符串，输出字符与字符串，这为我们接下来的设计奠定了基础。