汇编语言程序设计课程作业（六）

姓名：袁昊男学号：2018091618008

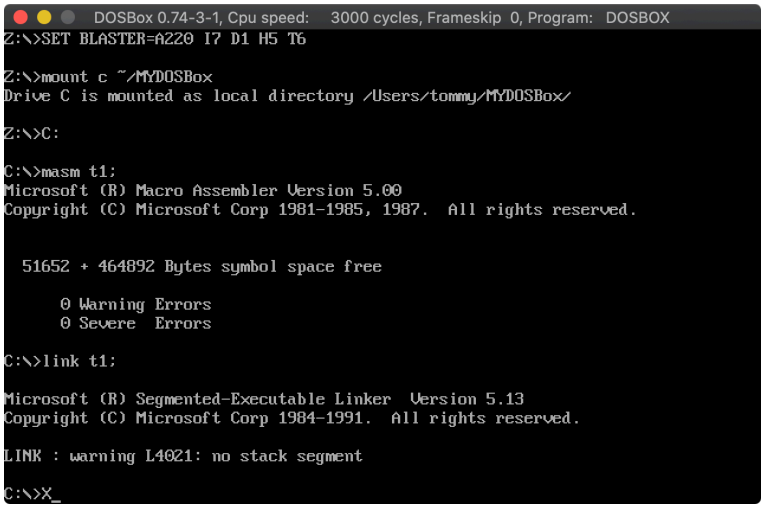
实验 5 编写、调试具有多个段的程序

(1) 将下面的程序编译、连接，用 Debug 加载、跟踪，然后回答问题。

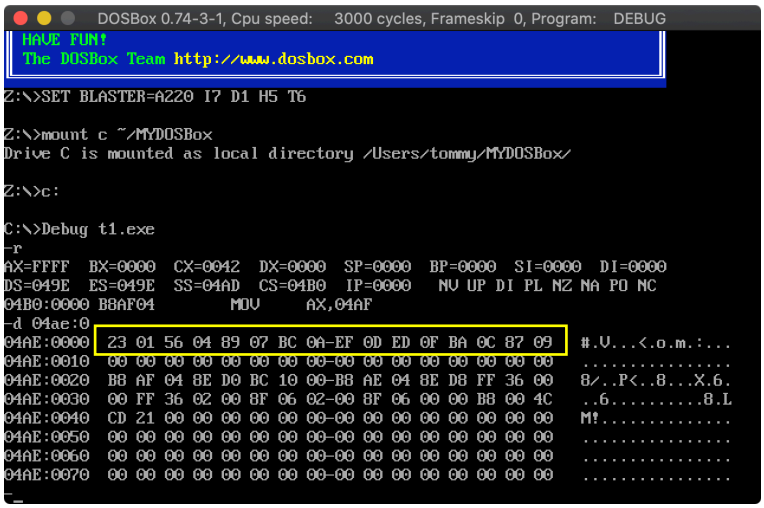
```
assume cs:code,ds:data,ss:stack

data segment
    dw 0123h,0456h,0789h,0abch,0defh,0fedh,0cbah,0987h
data ends

stack segment
    dw 0,0,0,0,0,0,0,0
stack ends

code segment

start: mov ax,stack
       mov ss,ax
       mov sp,16

       mov ax,data
       mov ds,ax

       push ds:[0]
       push ds:[2]
       pop ds:[2]
       pop  ds:[0]

       mov ax,4c00h
       int 21h

code ends

end start
```

① CPU 执行程序，程序返回前，data 段中的数据为多少？

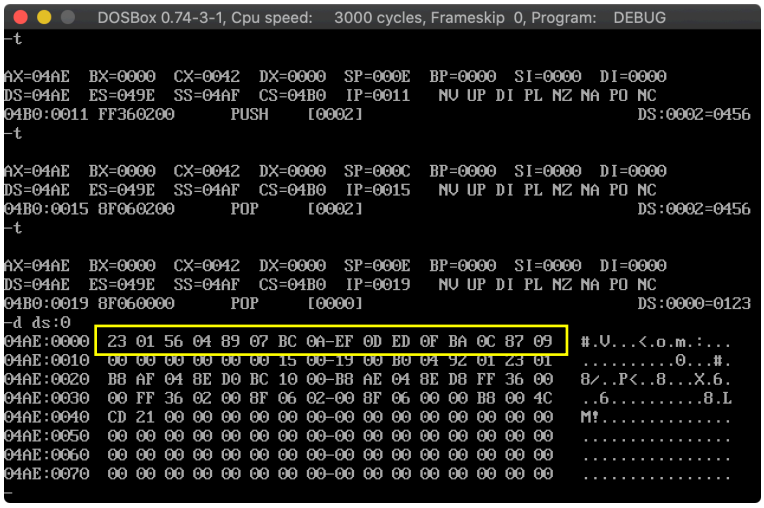
1) 编译、连接：



2) code 段运行前 data 段的数据：



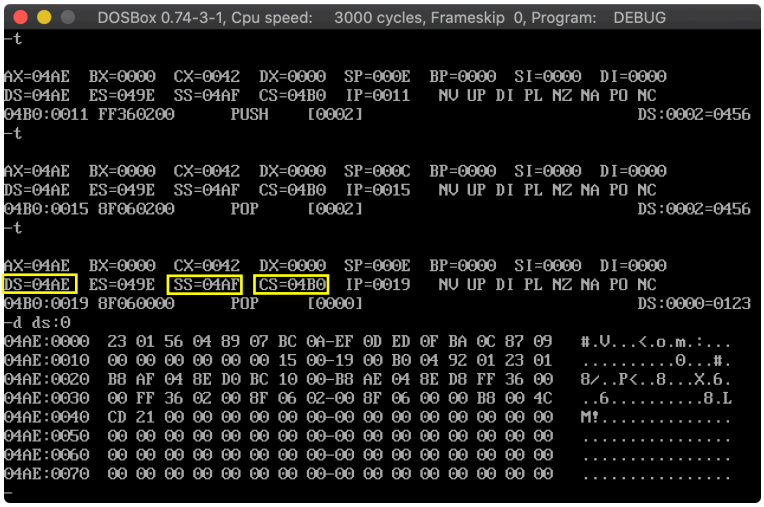
3) Debug 查看 data 段中数据：



4) 结论：data 段中的数据与程序运行前一致。

② CPU 执行程序，程序返回前，cs = _____、ss = _____、ds = _____。

1) Debug 中查看 cs、ss、ds 内容：



2) 结论：cs = 04b0h，ss = 04afh，ds = 04aeh。

③ 设程序加载后，code 段的段地址为 X，则 data 段的段地址为 X－2 ，stack 段的段地址为 X－1 。

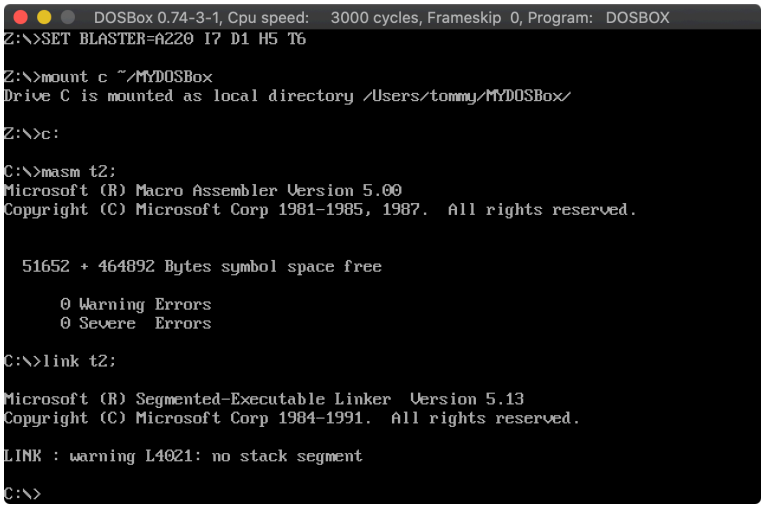
分析：data 段与 stack 段恰好各占 16 字节，因此段地址依次相差 1。

(2) 将下面的程序编译、连接，用 Debug 加载、跟踪，然后回答问题。

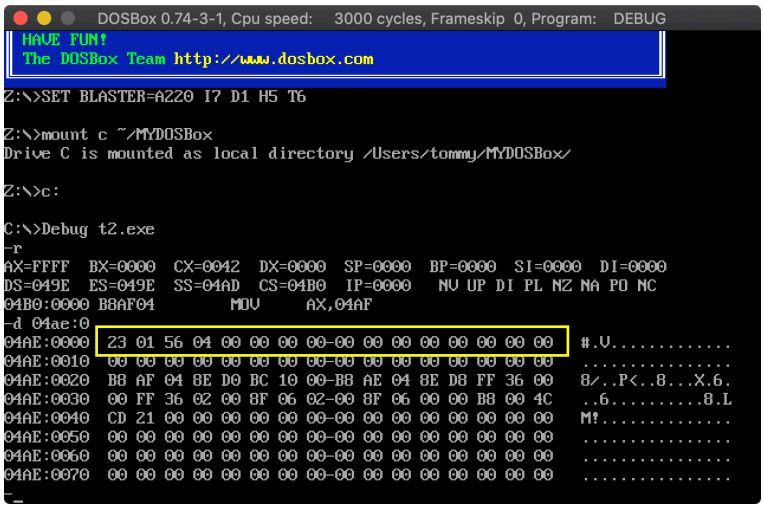
```
assume cs:code,ds:data,ss:stack

data segment
    dw 0123h,0456h
data ends

stack segment
    dw 0,0
stack ends

code segment

start: mov ax,stack
       mov ss,ax
       mov sp,16

       mov ax,data
       mov ds,ax

       push ds:[0]
       push ds:[2]
       pop ds:[2]
       pop  ds:[0]

       mov ax,4c00h
       int 21h

code ends

end start
```

① CPU 执行程序，程序返回前，data 段中的数据为多少？

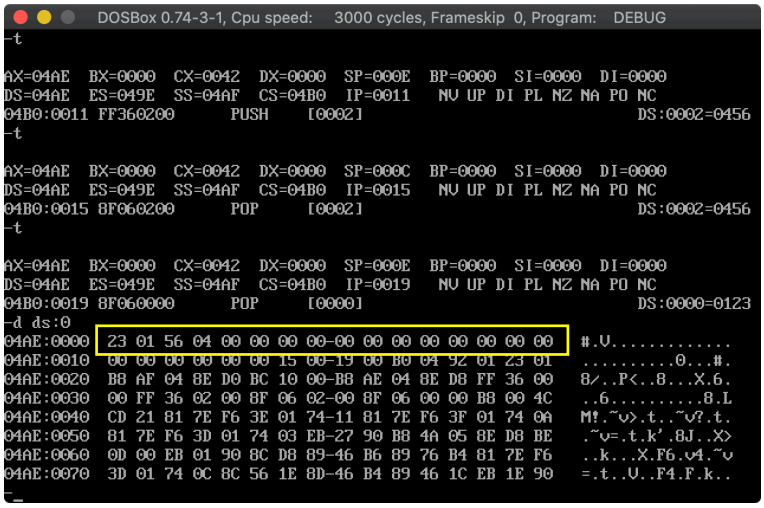
1) 编译、连接：



2) code 段运行前 data 段的数据：



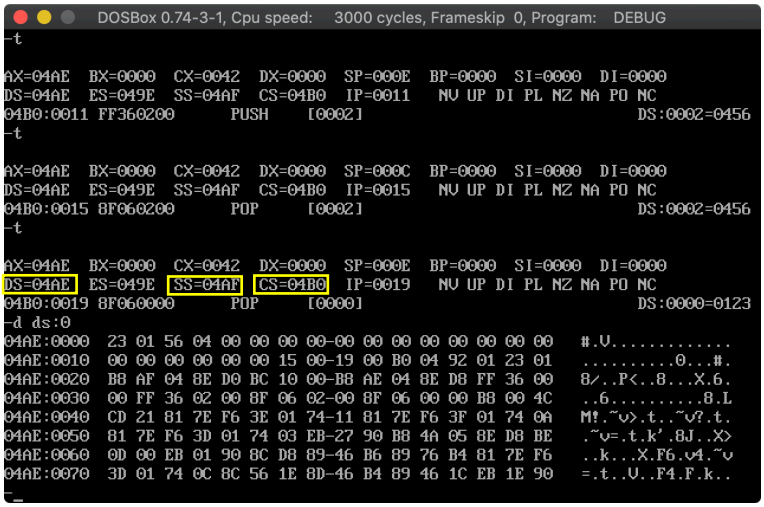
3) Debug 查看 data 段中数据：



4) 结论：data 段共 16 字节。执行程序后，定义的两个数据不变，其余用 0 补全。

② CPU 执行程序，程序返回前，cs = _____、ss = _____、ds = _____。

1) Debug 中查看 cs、ss、ds 内容：



2) 结论：cs = 04b0h，ss = 04afh，ds = 04aeh。

③ 设程序加载后，code 段的段地址为 X，则 data 段的段地址为 X－2 ，stack 段的段地址为 X－1 。

分析：data 段与 stack 段恰好各占 16 字节，因此段地址依次相差 1。

④ 对于如下定义的段：

```
name segment
    ...
name ends
```

如果段中的数据占 N 个字节，则程序加载后，该段实际占有的空间为 $\left(\left\lfloor\frac{N}{16}\right\rfloor+1\right)\times16$ 字节 。

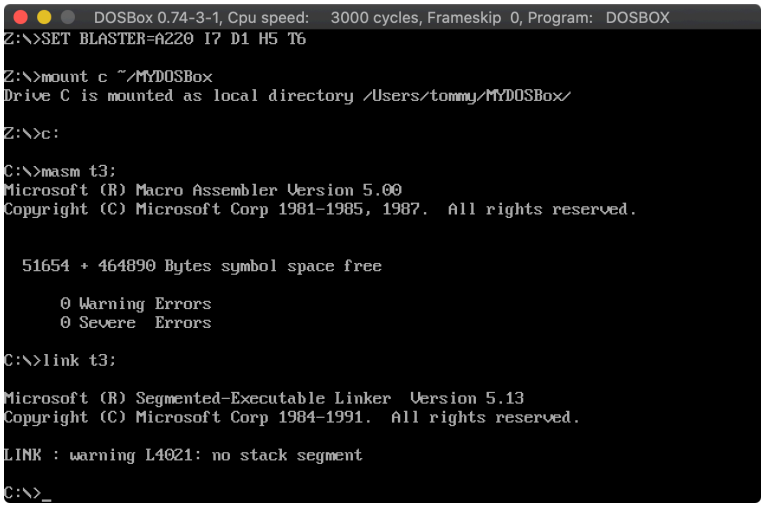
分析：地址分配以 16 字节为分配单元。不足 16 字节的按 16 字节算；超过 16 字节整数倍的按一个 16 字节算。综合以上。N 字节数据实际占有空间为 $\left(\left\lfloor\frac{N}{16}\right\rfloor+1\right)\times16$ 字节。

(3) 将下面的程序编译、连接，用 Debug 加载、跟踪，然后回答问题。

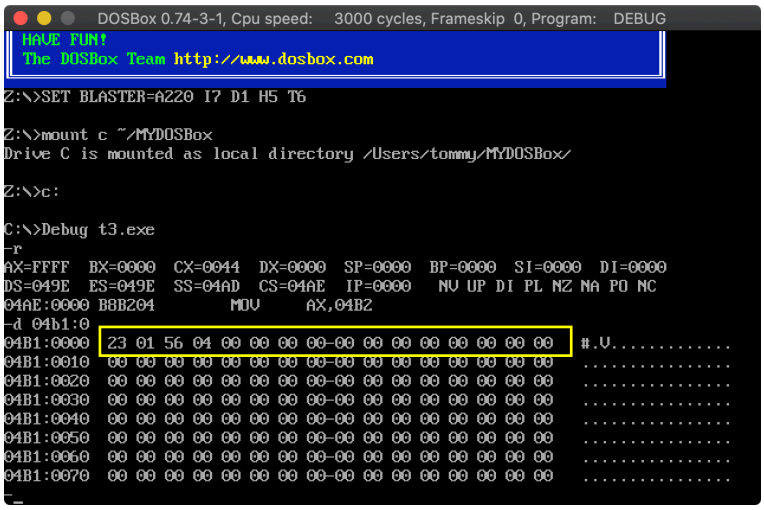
```
assume cs:code,ds:data,ss:stack

code segment

start: mov ax,stack
       mov ss,ax
       mov sp,16

       mov ax,data
       mov ds,ax

       push ds:[0]
       push ds:[2]
       pop ds:[2]
       pop  ds:[0]

       mov ax,4c00h
       int 21h

code ends

data segment
    dw 0123h,0456h
data ends

stack segment
    dw 0,0
stack ends

end start
```

① CPU 执行程序，程序返回前，data 段中的数据为多少？

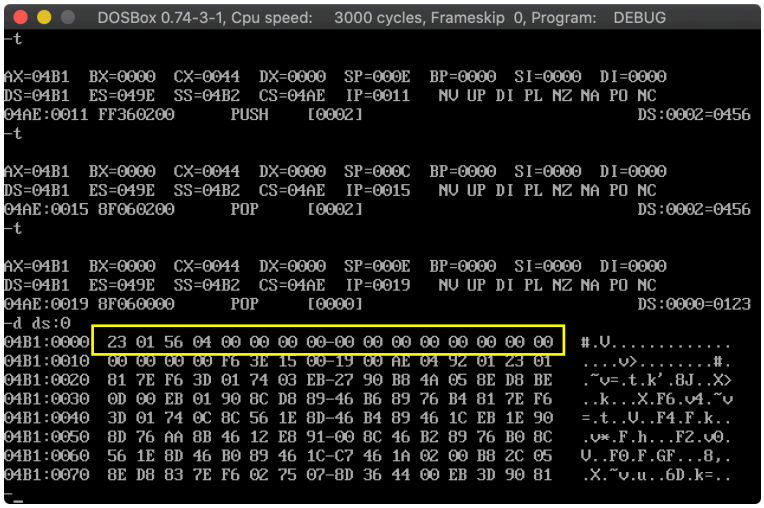
1) 编译、连接：



2) code 段运行前 data 段的数据：



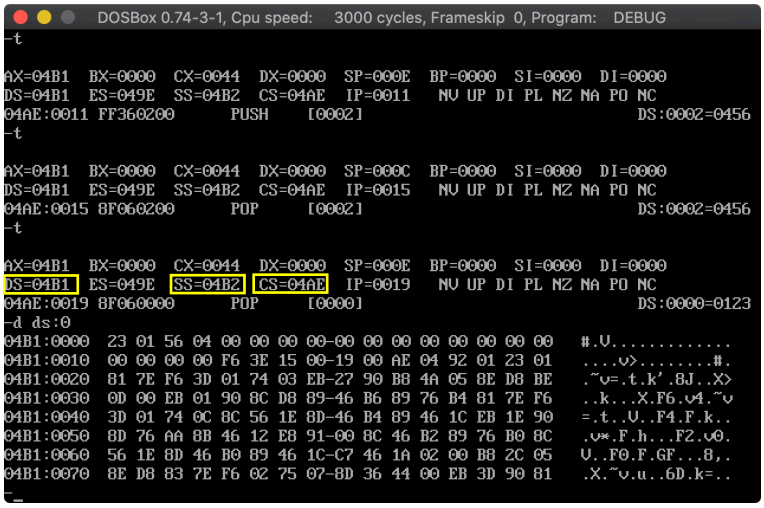
3) Debug 查看 data 段中数据：



4) 结论：data 段共 16 字节。执行程序后，定义的两个数据不变，其余用 0 补全。

② CPU 执行程序，程序返回前，cs = _____、ss = _____、ds = _____。

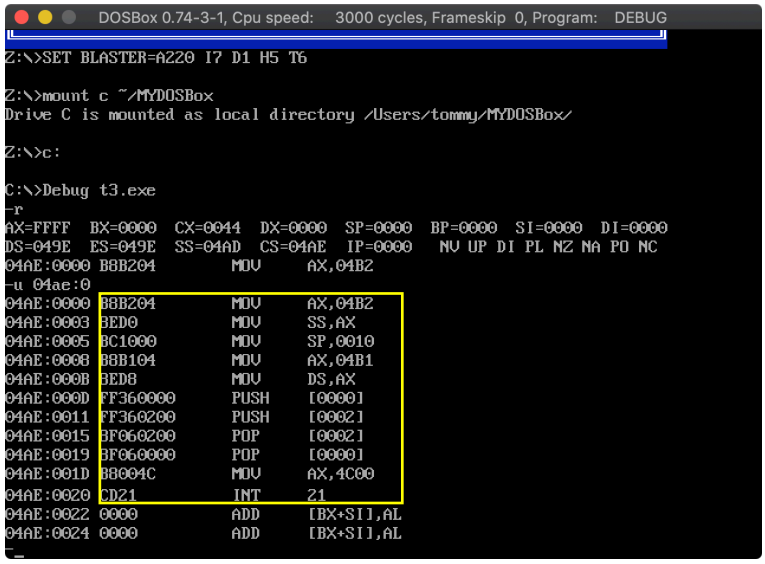
1) Debug 中查看 cs、ss、ds 内容：



2) 结论：cs = 04aeh，ss = 04b2h，ds = 04b1h。

③ 设程序加载后，code 段的段地址为 X，则 data 段的段地址为 X + 3 ，stack 段的段地址为 X + 4 。

分析：通过 Debug 查看 code 段数据长度：



简单计算可得：code 段数据长度为 34 字节，根据公式$\left(\left\lfloor\frac{N}{16}\right\rfloor+1\right)\times16$，34 字节数据实际占有 3 个 16 字节长度单元。因此，若 code 段的段地址为 X，则 data 段的段地址为 X + 3，stack 段的段地址为 X + 4。

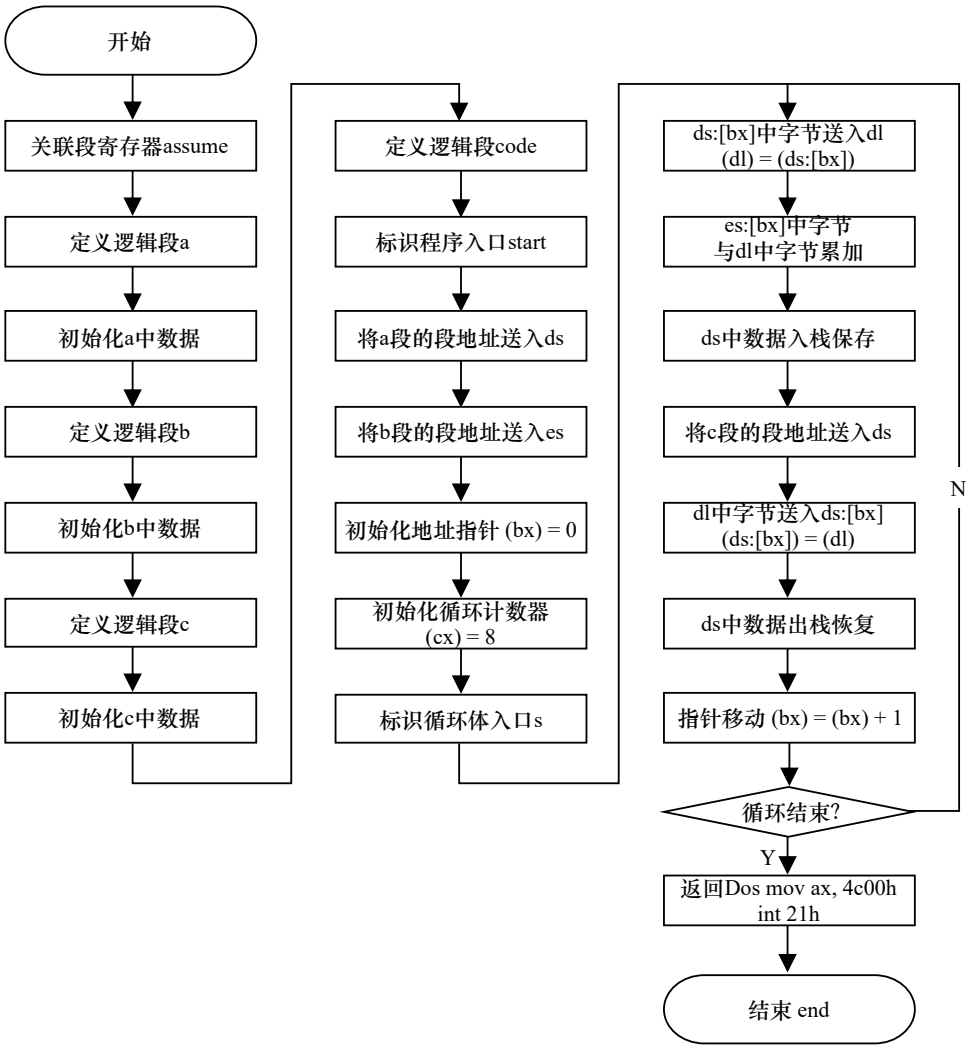
(4) 如果将(1)、(2)、(3)题中的最后一条伪指令“end start”改为“end”(也就是说，不指明程序的入口)，则哪个程序仍然可以正确执行？请说明原因。

答：都能执行，但只有(3)可以正确执行。因为如果不指明程序入口，程序将从头开始执行。那么(1)、(2)中一开始定义的数据将会被当作指令执行，而(3)由于就是以 code 段开头，因此不受影响，可以正确执行。

(5) 程序如下，编写 code 段中的代码，将 a 段和 b 段中的数据依次相加，将结果存到 c 段中。

```
assume cs:code

a segment
    db 1,2,3,4,5,6,7,8
a ends

b segment
    db 1,2,3,4,5,6,7,8
b ends

c segment
    db 0,0,0,0,0,0,0,0
c ends

code segment
start:
        ?
code ends

end start
```

1) 程序流程图：



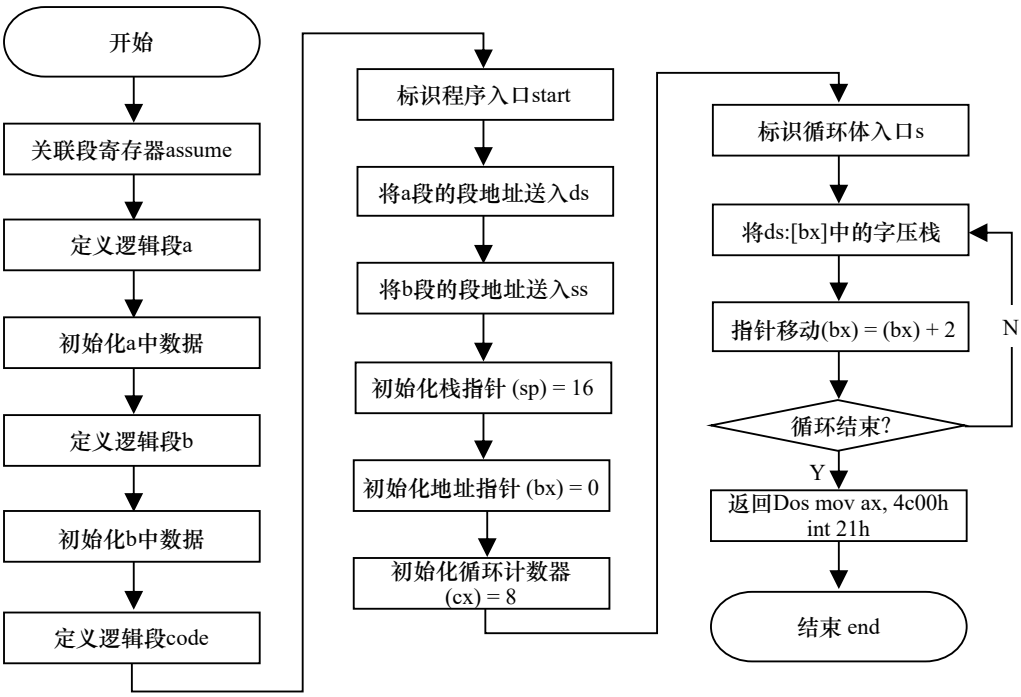
2) 补全程序：

```
start: mov ax,a                    mov ax,c
       mov ds,ax                   mov ds,ax
       mov ax,b                    mov [bx],dl
       mov es,ax                   pop ds
       mov bx,0                    inc bx
       mov cx,8                    loop s
    s: mov dl,[bx]
       add dl,es:[bx]              mov ax,4c00h
       push ds                     int 21h
```

(6) 程序如下，编写 code 段中的代码，用 push 指令将 a 段中的前 8 个字型数据逆序存储到 b 段中。

```
assume cs:code

a segment
    dw 1,2,3,4,5,6,7,8,9,0ah,0bh,0ch,0dh,0eh,0fh,0ffh
a ends

b segment
    dw 0,0,0,0,0,0,0,0
b ends

code segment
start:
       ?
code ends

end start
```

1) 程序流程图：



2) 补全程序：

```
start: mov ax,a
       mov ds,ax
       mov ax,b
       mov ss,ax
       mov sp,16
       mov bx,0
       mov cx,8
   s:  push [bx]
       add bx,2
       loop s

       mov ax,4c00h
       int 21h
```