检测点14.1

（1）

代码如下：

*assume* cs:code

code segment

*mov* al,2

*out* 70h,al

*in* al,71h

*mov* ax,4c00h

*int* 21h

code ends

end

结果如下：

文本

描述已自动生成

（2）

代码如下：

*assume* cs:code

code segment

*mov* al,2

*out* 70h,al

*mov* al,0

*out* 71h,al

*mov* ax,4c00h

*int* 21h

code ends

end

检测点14.2

代码如下：

结果如下：

文本

描述已自动生成

文本

描述已自动生成

可以看出，ax从原来的000ah（10）变为了0064h（100），即实现了ax\*10

实验14

代码如下：

*assume* cs:code ds:data

data segment

  db 9,8,7,4,2,0

data ends

code segment

start:

*mov* bx,data

*mov* ds,bx

*mov* si,0

*mov* bx,0b800h*;设置基本参数*

*mov* es,bx

*mov* di,160\*12+30\*2

*mov* cx,6

s0:

*push* cx

*mov* al,ds:[si]

*push* ax

*inc* si

*out* 70h,al

*in* al,71h

*mov* ah,al

*mov* cl,4

*shr* ah,cl

*and* al,00001111b

*add* ah,30h

*add* al,30h

*mov* byte ptr es:[di],al

*add* di,2

*pop* ax

*cmp* al,0*;读到0为结束*

*je* over

*cmp* al,7h*;判断分隔符号是什么*

*je* kongge

*ja* fanxiegang

*jb* maohao

fanxiegang:

*mov* ax,'/'

*jmp* next

maohao:

*mov* ax,':'

*jmp* next

kongge:

*mov* ax,' '

*jmp* next

next:

*mov* byte ptr es:[di],al

*add* di,2

over:

*pop* cx

*loop* s0

*jmp* start   *;产生死循环，再从头开始执行程序*

*mov* ax,4c00h

*int* 21h

code ends

end start

结果如下：

图片包含 文本

描述已自动生成

实验15：

代码如下：

*assume* cs:code

stack segment

 db 128 dup (0)

stack ends

code segment

start:

*mov* ax,stack

*mov* ss,ax

*mov* sp,128

*push* cs

*pop* ds

*mov* ax,0

*mov* es,ax

*mov* si,offset int9

*mov* di,204h

*mov* cx,offset int9end - offset int9

*cld*

    rep *movsb*

*cli*

*push* es:[9\*4]

*pop* es:[200h]

*push* es:[9\*4+2]

*pop* es:[202h]

*mov* word ptr es:[9\*4],204h

*mov* word ptr es:[9\*4+2],0

*sti*

*mov* ax,4c00h

*int* 21h

int9:

*push* ax

*push* bx

*push* cx

*push* dx

*in* al,60h

*pushf*

*call* dword ptr cs:[200h]

*cmp* al,9eh    *;设置a的阿斯克码*

*jne* int9ret   *;若不是A，则结束*

*mov* ax,0b800h

*mov* es,ax

*mov* bx,0

*mov* cx,2000

    s: *mov* byte ptr es:[bx],'A'*;将A输入在屏幕上*

*add* bx,2

*loop* s

int9ret:

*pop* es

*pop* cx

*pop* bx

*pop* ax

*iret*

int9end:*nop*

code ends

end start

实验结果如下：

背景图案

描述已自动生成

检测点16.1：

代码如下：

*assume* cs:code

code segment

        a dw 1,2,3,4,5,6,7,8

        b dd 0

start:  *mov* si,0

*mov* cx,8

s:

*mov* ax,a[si]

*add* word ptr b[0],ax

*adc* word ptr b[2],0

*add* si,2

*loop* s

*mov* ax,4c00h

*int* 21h

code ends

end start

结果如下：

日历

描述已自动生成

检测点17.1

答：正确。如果IF始终为零，则int 16h的0号功能在缓存区为零的情况下会一直循环，不会处理其他指令。如果没有使用int 9h指令读入一个数据后，使它终止，则计算机就不会读入其他数据，会陷入死循环。

实验17

使用除法计算查询各个分区，而后对其进行读取。

代码如下：

*assume* cs:code

code segment

start:

*mov* ax,cs

*mov* ds,ax

*mov* si,offset int7ch

*mov* ax,0

*mov* es,ax

*mov* di,200h

*mov* cx,offset int7ch\_end - offset int7ch

*cld*

    rep *movsb*

*cli*

*mov* word ptr es:[7ch\*4],200h

*mov* word ptr es:[7ch\*4+2],0

*sti*

*mov* ah,0

*mov* bx,1

*int* 7ch

*mov* ax,4c00h

*int* 21h

*;bx为入口参数，16位除法，dx高位，ax低位，ax商，dx余数*

int7ch:

*cmp* ah,1

*ja* over

*push* cx

*push* dx

*push* bx

*push* ax

*mov* ax,bx

*mov* dx,0

*mov* bx,1440

*div* bx

*mov* cl,al       *;盘面号，先存在cl中*

*mov* ax,dx

*mov* dh,cl       *;将暂存在cl中的盘面号存在dh中*

*mov* dl,0        *;驱动器号，软驱 0*

*mov* bl,18

*div* bl

*mov* ch,al       *;磁道号*

*inc* ah

*mov* cl,ah       *;扇区号*

*mov* ax,0b800h

*mov* es,ax

*mov* bx,160\*5    *;es:bx  指向读写区*

*pop* ax

*push* ax

*mov* al,8        *;读写扇区数*

*cmp* ah,0

*je* read

*cmp* ah,1

*je* write

read:

*mov* ah,2

*jmp* short ok

write:

*mov* ah,3

*jmp* short ok

ok:

*int* 13h

*pop* ax

*pop* bx

*pop* dx

*pop* cx

over:

*iret*

int7ch\_end:

*nop*

code ends

end start

因为dosbox无法模拟磁盘，故无法验证其是否正确。

而若能进行测试，则结果情况如下：

如果实验成功，al会存储操作的扇区数，ah=0；不成功会在ah提供出错代码。