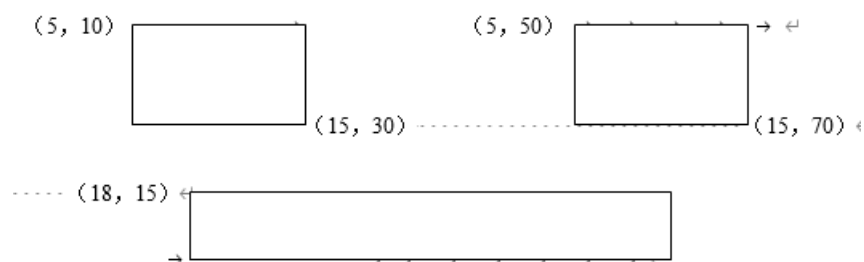


第十周试验任务：自行编写一个键盘输入并屏幕窗口处理程序，它可以完成键盘字符的读入并进行屏幕显示。



光标首先定位在右窗口最下面一行的行首 (15, 50)，如从键盘输入字符，则显示在右窗口，同时也显示在下窗口的最下面一行。若需要将字符显示于左窗口，则先按下←键，接着再从键盘输入字符，字符就会从左窗口的最下行开始显示，同时下窗口也显示出左窗口的内容。如若再按下→键，输入字符就会接在先前输入的字符之后显示出来。当一行字符显示满后（左右窗口一行显示20个字符，下窗口一行显示50个字符），窗口自动向上卷动一行，输入字符继续显示于最低一行，窗口最高一行向上卷动后消失。

第十周实验报告：键盘输入并屏幕窗口处理程序

1. 具体功能实现细节：

1.1 具体实现难点：

1.1.1 窗口的初始化

1.1.2 光标的移动

1.1.3 窗口如何实现向上翻一行

1.2 流程图

附录1：输出结果

附录2：具体代码实现

第十周实验报告：键盘输入并屏幕窗口处理程序

1. 具体功能实现细节：

1.1 具体实现难点：

1.1.1 窗口的初始化

该程序中主要是利用了BIOS中的10号中断的6号功能实现相应窗口的实现：

10号中断的6号功能和7号功能：

- 功能描述：初始化屏幕或滚屏
- 入口 参数：AH = 06H——向上滚屏，07H——向下滚屏
- AL = 滚动行数(0——清窗口)
- BH = 空白区域的缺省属性
- (CH、CL) = 窗口的左上角位置(Y坐标，X坐标)

- (DH、DL) = 窗口的右下角位置(Y坐标, X坐标)
- 出口参数: 无

```

1 ; 该程序主要使用宏来实现相关的功能
2 scroll macro count,r0,c0,r1,c1
3     mov     al,count
4     mov     bh,70h
5     mov     ch,r0
6     mov     cl,c0
7     mov     dh,r1
8     mov     dl,c1
9     mov     ah,6
10    int     10h
11 endm

```

1.1.2 光标的移动

我们想要实现光标的定点移动, 我们需要用到10号中断的2号功能:

10号中断的2号功能:

- 功能描述: 用文本坐标下设置光标位置
- 入口参数: AH = 02H
- BH = 显示页码
- DH = 行(Y坐标)
- DL = 列(X坐标)
- 出口参数: 无

```

1 ; (行号,列号)
2 post_c macro y,x
3     ; 显示页码
4     mov     bh,0
5     mov     dh,y
6     mov     dl,x
7     mov     ah,2
8     int     10h
9 endm

```

1.1.3 窗口如何实现向上翻一行

这里我们还是主要利用10号中断的6号功能, 其中AL寄存器存储的是往上滚动的行数, 如果我们设置成0的话就相当于清屏; 如果我们设置为其他的数字比如1的话, 就会实现将指定区域的“屏幕”上翻一行的操作。

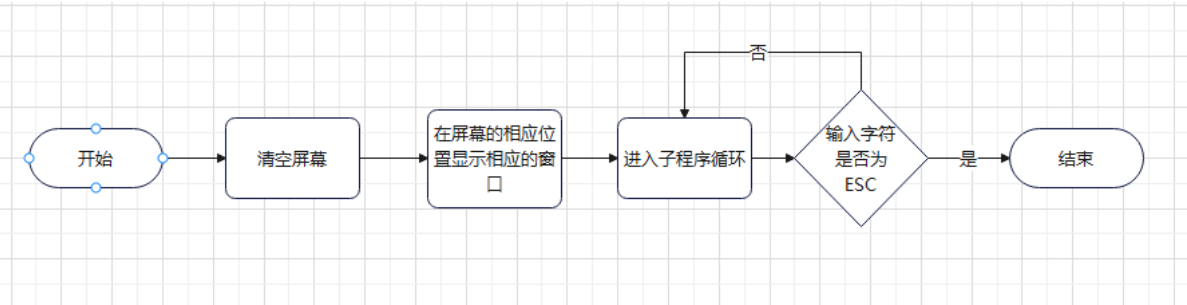
```

1 ; 左边的窗口上翻一行
2 scroll 1,5,10,15,30
3 ; 右边的窗口上翻一行
4 scroll 1,5,50,15,70
5 ; 下边的窗口上翻一行
6 scroll 1,18,15,22,65

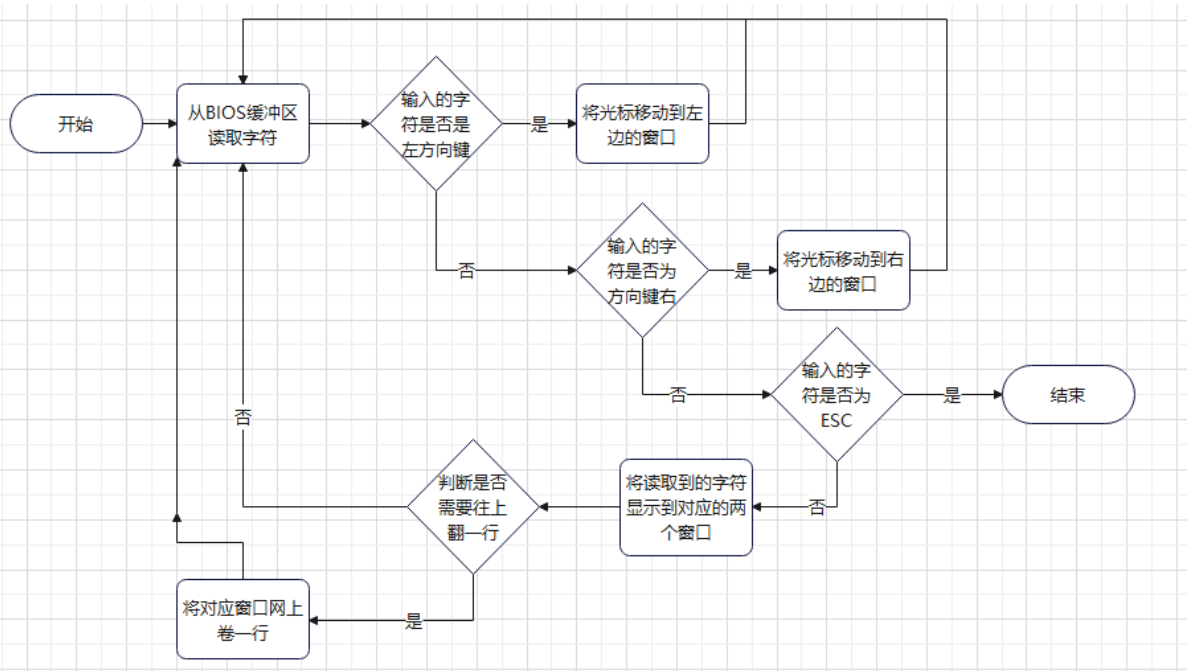
```

1.2 流程图

主程序:



子程序:



附录1：输出结果



附录2：具体代码实现

```
1 ; 该程序是探索10号中断的具体实现效果
2 assume cs:codes,ds:data
3 data segment
4 ;左窗口初始光标位置
5 lx db 10
6 ly db 15
7
8 ;右窗口初始光标位置
9 rx db 50
10 ry db 15
11
12 ;下窗口
13 dwx db 15
14 dwy db 22
15 windowflag db 1
16 ; =1右边窗口;=2左边窗口
17 data ends
18
19
20 ; 利用宏指令来实现
21 ; 在此处macro 是伪指令，告诉大家这里是伪指令，是在编译的时候执行的，应该就是
22 ; 替代相应的代码行
23 ; a,b,c,d 是传进来的参数
24 clear macro a,b,c,d
25     mov     al,0
26     mov     bh,7
27     mov     ch,a
28     mov     cl,b
29     mov     dh,c
30     mov     dl,d
31     mov     ah,6
32     int     10h
33 endm
34
35
36 scroll macro count,r0,c0,r1,c1
37     mov     al,count
38     mov     bh,70h
39     mov     ch,r0
40     mov     cl,c0
41     mov     dh,r1
42     mov     dl,c1
43     mov     ah,6
44     int     10h
45 endm
46
47 ; (行号,列号)
48 post_c macro y,x
49     ; 显示页码
50     mov     bh,0
51     mov     dh,y
52     mov     dl,x
```

```

53     mov     ah,2
54     int     10h
55 endm
56
57 display macro
58     mov     bh,0;显示页码
59     mov     cx,1;重复输出字符的次数
60     mov     ah,0ah
61     int     10h
62
63     ;同样在下窗口输出
64     post_c   dwy,dwx
65     mov     bh,0
66     mov     cx,1
67     mov     ah,0ah
68     int     10h
69 endm
70
71 getchar macro
72 input :
73     ;读取缓冲区中的字符，并进行判断，此处会有光标出现
74     mov     ah,0
75     int     16h
76     cmp     ah,4bh
77     jnz     no_left
78     ; 如果相等的话，就说明在左边的窗口输入
79     post_c   ly,lx
80     mov     windowflag,2
81     jmp     input
82 no_left:
83     cmp     ah,4dh
84     jnz     no_right
85     post_c   ry,rx
86     mov     windowflag,1
87     jmp     input
88 no_right:
89     ;判断字符是否为ESC
90     cmp     ah,01h
91     jnz     continue1
92     clear    0,0,24,79
93
94     ;mov     ah,4ch
95     ;int     21h
96     ret
97 continue1:
98     display
99     inc     dwx
100    cmp     dwx,65
101    jle     isright ;如果不相等说明下窗口中光标没有移动到末尾，不需要往上翻
102    scroll  1,18,15,22,65
103    mov     dwx,15
104 isright:
105    cmp     windowflag,1
106    jnz     displeft
107 displeft:

```

```

108      ;当前光标在右窗口中
109      ;判断右边的窗口是否需要上翻
110      inc     rx
111      cmp     rx,70
112      jle     rightexit ;如果小于的话,说明并未需要翻页
113      ;否则的话,需要往上翻一行
114      scroll   1,5,50,15,70
115      mov     rx,50
116  rightexit:
117      post_c  ry,rx
118      jmp     input
119
120  displeft:
121      ;当前光标在左窗口中
122      ;判断左窗口是否需要上翻
123      inc     lx
124      cmp     lx,30
125      jle     leftexit
126      scroll   1,5,10,15,30
127      mov     lx,10
128
129  leftexit:
130      post_c  ly,lx
131      jmp     input
132  endm
133
134
135
136  codes segment
137  start:
138      ; 显示器
139  main proc far
140      push    ds
141      sub     ax,ax
142      push    ax
143
144      mov     ax,data
145      mov     ds,ax
146      ; 清空屏幕
147      clear   0,0,24,79
148      ; 画出三个窗口
149      scroll   10,5,10,15,30
150      scroll   10,5,50,15,70
151      scroll   4,18,15,22,65
152
153      ; 光标的初始位置,注意这里设置光标的初始位置并不是说运行程序之后就可以看到
154      ; 后面我们还需要调用DOS的21号中断,输入字符的时候才会有光标显示
155      post_c  15,50
156      getchar
157
158      ret
159  main endp
160
161
162  codes ends

```

163

164

end start