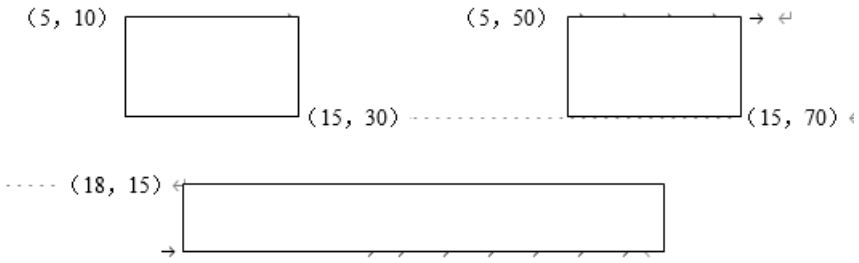
> 第十周试验任务：自行编写一个键盘输入并屏幕窗口处理程序，它可以完成键盘字符的读入并进行屏幕显示。
>
> 
>
>
> 光标首先定位在右窗口最下面一行的行首（15，50），如从键盘输入字符，则显示在右窗口，同时也显示在下窗口的最下面一行。若需要将字符显示于左窗口，则先按下←键，接着再从键盘输入字符，字符就会从左窗口的最下行开始显示，同时下窗口也显示出左窗口的内容。如若再按下→键，输入字符就会接在先前输入的字符之后显示出来。当一行字符显示满后（左右窗口一行显示20个字符，下窗口一行显示50个字符），窗口自动向上卷动一行，输入字符继续显示于最低一行，窗口最高一行向上卷动后消失。

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

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

```
; 该程序主要使用宏来实现相关的功能
scroll  macro count,r0,c0,r1,c1
    mov     al,count
    mov     bh,70h
    mov     ch,r0
    mov     cl,c0
    mov     dh,r1
    mov     dl,c1
    mov     ah,6
    int     10h
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

```
; (行号,列号)
post_c  macro y,x
    ; 显示页码
    mov     bh,0
    mov     dh,y
    mov     dl,x
    mov     ah,2
    int     10h
endm
```

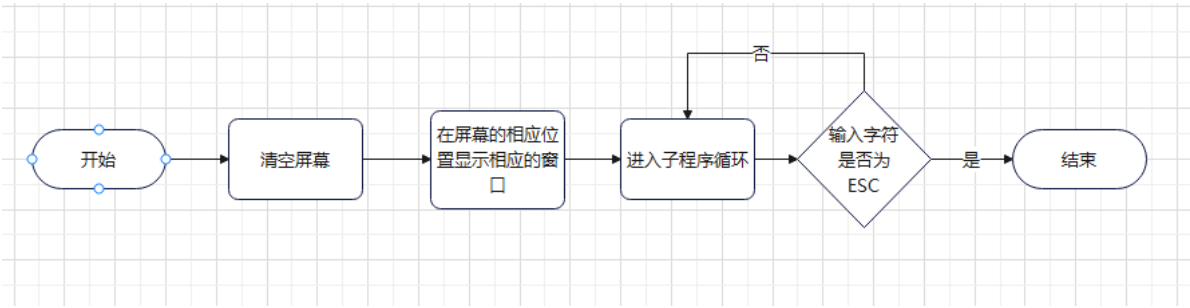
1.1.3 窗口如何实现向上翻一行

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。

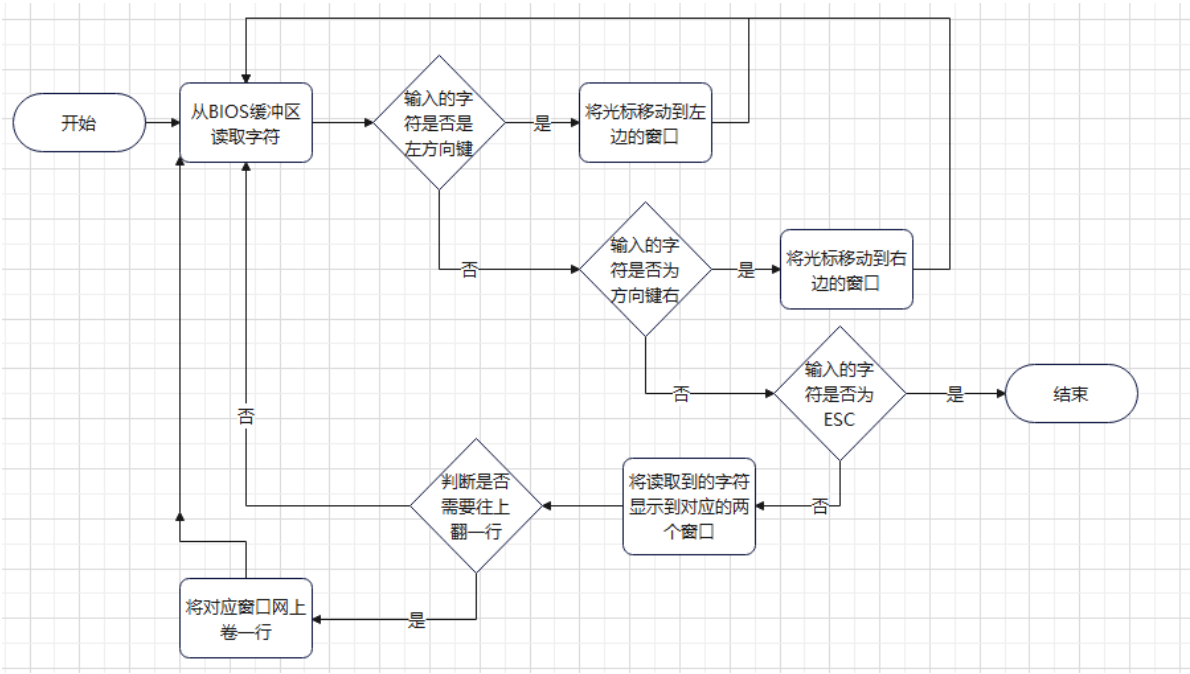
```
; 左边的窗口上翻一行
scroll  1,5,10,15,30
; 右边的窗口上翻一行
scroll  1,5,50,15,70
; 下边的窗口上翻一行
scroll  1,18,15,22,65
```

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

```
; 该程序是探索10号中断的具体实现效果
assume cs:codes,ds:data
data    segment
    ;左窗口初始光标位置
    lx  db  10
    ly  db  15

    ;右窗口初始光标位置
    rx  db  50
    ry  db  15

    ;下窗口
    dwx db  15
    dwy db  22
    windowflag db 1
    ; =1右边窗口;=2左边窗口
data    ends


; 利用宏指令来实现
; 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
; 替代相应的代码行
; a,b,c,d 是传进来的参数
clear   macro a,b,c,d
    mov     al,0
    mov     bh,7
    mov     ch,a
    mov     cl,b
    mov     dh,c
    mov     dl,d
    mov     ah,6
    int     10h
endm


scroll  macro count,r0,c0,r1,c1
    mov     al,count
    mov     bh,70h
    mov     ch,r0
    mov     cl,c0
    mov     dh,r1
    mov     dl,c1
    mov     ah,6
    int     10h
endm

; (行号,列号)
post_c  macro y,x
    ; 显示页码
    mov     bh,0
    mov     dh,y
    mov     dl,x
```

```
    mov     ah,2
    int     10h
endm

display macro
    mov     bh,0;显示页码
    mov     cx,1;重复输出字符的次数
    mov     ah,0ah
    int     10h

    ;同样在下窗口输出
    post_c  dwy,dwx
    mov     bh,0
    mov     cx,1
    mov     ah,0ah
    int     10h
endm

getchar macro
input :
    ;读取缓冲区中的字符，并进行判断，此处会有光标出现
    mov     ah,0
    int     16h
    cmp     ah,4bh
    jnz     no_left
    ; 如果相等的话，就说明在左边的窗口输入
    post_c  ly,lx
    mov     windowflag,2
    jmp     input
no_left:
    cmp     ah,4dh
    jnz     no_right
    post_c  ry,rx
    mov     windowflag,1
    jmp     input
no_right:
    ;判断字符是否为ESC
    cmp     ah,01h
    jnz     continue1
    clear   0,0,24,79

    ;mov     ah,4ch
    ;int     21h
    ret
continue1:
    display
    inc     dwx
    cmp     dwx,65
    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
    scroll  1,18,15,22,65
    mov     dwx,15
isright:
    cmp     windowflag,1
    jnz     displeft
dispright:
```

```
    ;当前光标在右窗口中
    ;判断右边的窗口是否需要上翻
    inc     rx
    cmp     rx,70
    jle     rightexit ;如果小于的话，说明并未需要翻页
    ;否则的话，需要往上翻一行
    scroll  1,5,50,15,70
    mov     rx,50
rightexit:
    post_c  ry,rx
    jmp     input

displeft:
    ;当前光标在左窗口中
    ;判断左窗口是否需要上翻
    inc     lx
    cmp     lx,30
    jle     leftexit
    scroll  1,5,10,15,30
    mov     lx,10

leftexit:
    post_c  ly,lx
    jmp     input
endm



codes segment
start:
; 显示器
main proc far
    push    ds
    sub     ax,ax
    push    ax

    mov     ax,data
    mov     ds,ax
    ; 清空屏幕
    clear   0,0,24,79
    ; 画出三个窗口
    scroll  10,5,10,15,30
    scroll  10,5,50,15,70
    scroll  4,18,15,22,65

    ; 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
    ; 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
    post_c  15,50
    getchar

    ret
main endp


codes ends
```

```
163
164 end start
```