一、基本功能描述

1. 实现汉字与英文的混合显示。
2. 对文本的特殊字符进行如下处理：遇到回车换行符号则换行，遇到'\t'则留出一个空格。
3. 实现根据键盘输入执行相应的功能，具体如下表：

键盘输入	功能描述
q	(quit)退出并重新返回选择文件界面
s	(slant)设置斜体
A	(Auto)设置自动翻屏
a	(auto)设置自动跳转到下一行
I	(Increase)增加行间距
D	(Decrease)减少行间距
i	(increase)增加列间距
d	(decrease)减少列间距
c	(color)改变字体颜色
n	(next)跳转到下一行
p	(previous)跳转到前一行（取决于上一次的'n'操作）
P	(Previous)跳转到前一屏（取决于上一次的'N'操作）
其他按键	退出程序

二、主要变量及函数介绍

2.1 主要变量

变量	描述
line	当前字符所在的起始行
column	当前字符所在的起始列
line_interval	字符的行间距
column_interval	字符的列间距
line_off	一个字符内的行偏移量，用于实际显示时计算点的坐标
column_off	一个字符内的列偏移量，用于实际显示时计算点的坐标
max_char_num	一个常数，表示一屏所能显示的最多字节数
line_auto_mov	决定自动跳转下一行功能的开启与关闭
scr_auto_mov	决定自动跳转下一屏功能的开启与关闭
italic_char	选择斜体字显示还是正常显示
change_init_col	决定在显示斜体字时是否要将初始列值减 1
char_color	字符颜色
eng_file	英文点阵文件 asc16 的路径

hzk_file	中文点阵文件 Hzk16 的路径
data_file_name	存放 int 21h（0ah 从键盘读入字符串）得到的数据
data_file	要显示的文件的实际文件名
data_file_header_high	data_file 文件指针
data_file_header_low	
info	提示输入信息
eng_file_error	文件打开错误信息
hzk_file_error	
data_file_error	
head_exceed_error	移动文件指针出错信息
eng_handle	文件句柄
hzk_handle	
data_handle	
disp_data	一屏要显示的字符串的缓存区
char_num_limit	一屏所能显示的字节数上限（从 data_file 文件中实际读到的字节数）
char_num	当前屏幕实际显示的字节数
last_char_num	被保存的上一屏幕实际显示的字节数
first_line_charnum	当前屏幕第一行实际显示的字节数
last_line_charnum	被保存的上一行实际显示的字节数
eng_buffer	英文字符点阵数据的缓存区
hzk_buffer	汉字字符点阵数据的缓存区

2.2 主要函数

函数	功能
eng_char_handle	英文字符处理函数，并进行当前空间是否足够显示的判断和显示完一个英文字符后列值的更新
get_eng_dots	eng_char_handle 的子函数，得到英文字符点阵数据
eng_disp	eng_char_handle 的子函数，显示英文字符，并实现对斜体字的显示
hzk_char_handle	中文字符处理函数，并进行当前空间是否足够显示的判断、显示完一个中文字符后列值的更新和 lodsb 操作对实际显示的字节数 char_num 及第一行实际显示的字节数 first_line_charnum 的更新
get_hzk_dots	hzk_char_handle 的子函数，得到中文字符点阵数据
hzk_disp	hzk_char_handle 的子函数，显示中文字符，并实现对斜体字的显示
func	处理按键功能
control_italic	设置斜体
auto_mov_screen	自动翻屏
auto_mov_line	自动跳转下一行
delay	延时函数，auto_mov_screen 和 auto_mov_line 需要调用此函数来实现延时一段时间后自动跳转（下一屏\下一行）

inc_line_interval	增加行间距的函数
dec_line_interval	减少行间距
inc_col_interval	增加列间距
dec_col_interval	减少列间距
change_char_color	改变字体颜色
next_line	跳转到下一行
pre_line	跳转到前一行
next_screen	跳转到下一屏
pre_screen	跳转到上一屏
change_line	换行

三、算法设计及分析

3.1 汉字英文混合显示功能

采用汉字和英文的点阵显示，算法步骤如下：

1. 判断当前是中文字符还是英文字符：由于 ASCII 表的的低 128 位以一个字节存放英文的内码，高于 128 的以两个字节来存放汉字的内码，所以只需判断 lodsb 得到的 byte 是否大于 127：如果不大于 127 则当前读到的字符为英文；如果大于 127 则当前读到的字符为中文，此时继续 lodsb，得到的 word 即为当前读到的完整的中文字符。
2. 根据读到的字符找到它在字库文件的地址
   a) 英文：由于一个英文字模占用了 16 字节，所以
   地址=字符*16
   与汇编对应的 c 语言的伪代码如下：

```
char *s;                    //当前读到的英文字符
int location=(*s)*16;
```

   b) 中文：汉字字模的前一个字节为该汉字的区码，后一个字节为该汉字的区内编码，即该字在该区中的位置。每一个区共记录 94 个汉字。因为汉字的内码从 A0 区开始，而数组是以 0 为开始而区码和区内编码是以 1 为开始的,所以需要将区码和区内编码都减去 a1h。又因为一个中文字模占用了 32 字节，所以
   地址=((区码-a1h)*94 + (区内编码-a1h))*32
   与汇编对应的 c 语言的伪代码如下：

```
char *s;                    //当前读到的中文字符前八位
char qm= *s;                //区码
char qnm= *(s+1);           //区内编码
int location=(94*(qm-0xa1)+(qnm-0xa1))*32;
```

3. 根据偏移地址得到字符的点阵数据，并存放到相应的缓存区
   a) 英文
   与汇编对应的 c 语言的伪代码如下：

```
fseek(eng_handle,location,0);   //eng_handle是英文点阵文件asc16的文件指针
fread(eng_buffer,1,16,eng_handle); //eng_buffer是英文点阵数据的缓存区
```

   b) 中文

与汇编对应的 c 语言的伪代码如下：

```
fseek(hzk_handle,location,0);   //hzk_handle是中文点阵文件Hzk16的文件指针
fread(hzk_buffer,1,32,hzk_handle); // hzk_buffer是中文点阵数据的缓存区
```

4. 根据点阵数据在屏幕上显示
   a) 英文

与汇编对应的 c 语言的伪代码如下：

```
int i;
int j;
for(i=0;i<16;i++)         //16行
{
    for(j=0;j<8;j++)      //8列
    {
        if(((eng_buffer<<j)&0x80)!=NULL)
            putpixel(column+j,line+i,char_colour);
    }
}
```

   b) 中文

与汇编对应的 c 语言的伪代码如下：

```
int i;
int j;
for(i=0;i<16;i++)         //16行
{
    for(j=0;j<16;j++)     //16列
    {
        if(((hzk_buffer<<j)&0x80)!=NULL)
            putpixel(column+j,line+i,char_colour);
    }
}
```

3.2 斜体字功能

选择斜体字显示还是正常显示由变量 italic_char 控制的（1 为斜体，0 为正常显示）。
由于我的斜体字采用的是 60°的倾斜，而且每一个字都占了 16 行，所以斜体字的列数会增加 8。实现斜体字的算法是在画一个字时先把初始的列值加 8，每扫描完 2 行后将列值减 1，直到减为原来初始的列值，整个字就已经显示完毕了。判断是否扫描完 2 行、应该将初始列值减 1 是由变量 change_init_col 控制的：每次扫描完一行、判断时斜体字后判断 change_init_col 是否为 1，如果为 1 则将 change_init_col 重置为 0，列值不变；如果为 0 则列值减 1 ，并将 change_init_col 置 1。

3.3 换行处理及判断字符是否显示

以下情况都需要换行：

1. 列数达到上限 640
2. 遇到换行符号
3. 当前列数已容纳不下最后一个字符的空间：
   a) 对于英文至少需要 8 列，所以列数大于 632 的必须换行
   b) 对于中文至少需要 16 列，所以列数大于 624 的必须换行

c) 对于斜体字，还需要再预留 8 列的空间，否则下一个字的显示可能会不完整

换行由函数 change_line 实现。

由于程序有增加行间距的功能，当行值大于 464 时，当前的空间已无法完整显示最后一行字符串，此时就不应该显示最后一行字符串，需要跳出主程序中的_LOOP 循环。

3.4 跳转下/上一行功能

在显示当前一屏字符串时对第一行实际显示的字符串进行计数（只需要在显示的_LOOP 循环中判断行数是不是 1 并对 lodsb 操作进行计数），将计数值放到变量 first_line_charnum 中。

执行跳转下一行的功能时，只需把当前的文件指针再往下移动 first_line_charnum 即可（文件指针的偏移量定义在变量 data_file_header_high 和 data_file_header_low 中）。

每一次执行跳转下一行的功能时，都把旧的 first_line_charnum 的值拷贝到变量 last_line_charnum 中，这样就可以在这样的操作后回到上一行（把当前的文件指针再往上移动 last_line_charnum）。

跳转下/上一行功能分别由函数 next_screen 和 pre_screen 实现。

3.5 自动跳转下一行功能

自动跳转下一行功能的开启与关闭由变量 line_auto_mov 决定（1 为开启，0 为关闭）。每次显示完一屏字符串后进行 line_auto_mov 的判断：当 line_auto_mov 为 1 时，先调用 delay 进行延时，再调用 next_line 跳转到下一行。

自动跳转下一行功能由函数 auto_mov_line 实现。

3.6 跳转下/上一屏功能

640*480 的一屏幕最多可以显示 2400 个 byte，但实际从文件读到的并不一定有这么多。所以需要更新实际读到的 byte 的数量作为_LOOP 循环的计数值。但由于有回车换行，加上 3.3 提到的空间不够显示的问题，所以需要对实际显示当前一屏的字符串进行计数（即_LOOP 循环实际执行 lodsb 的次数），将计数值放到变量 char_num 中。执行跳转下一屏的功能时，只需把当前的文件指针再往下移动 char_num 即可（文件指针的偏移量定义在变量 data_file_header_high 和 data_file_header_low 中）。

每一次执行跳转下一行的功能时，都把旧的 char_num 的值拷贝到变量 last_char_num 中，这样就可以在这样的操作后回到上一行（把当前的文件指针再往上移动 last_char_num）。

跳转下/上一行功能分别由函数 next_screen 和 pre_line 实现。

3.7 自动跳转下一屏功能

自动跳转下一屏功能的开启与关闭由变量 scr_auto_mov 决定（1 为开启，0 为关闭）。每次显示完一屏字符串后进行 scr_auto_mov 的判断：当 scr_auto_mov 为 1 时，先调用 delay 进行延时，再调用 next_screen 跳转到下一行。

自动跳转下一行功能由函数 auto_mov_line 实现。

3.8 改变列间距

变量 column_interval 定义了字符间的列间距，每画完一个英文字符后列值需要增加(8+column_interval)，每画完一个中文字符后列值需要增加(16+column_interval)。需要注意的是需要对 column_interval 进行非负判断以保证该变量不小于 0，否则相邻的字会有重叠。

3.9 改变行间距

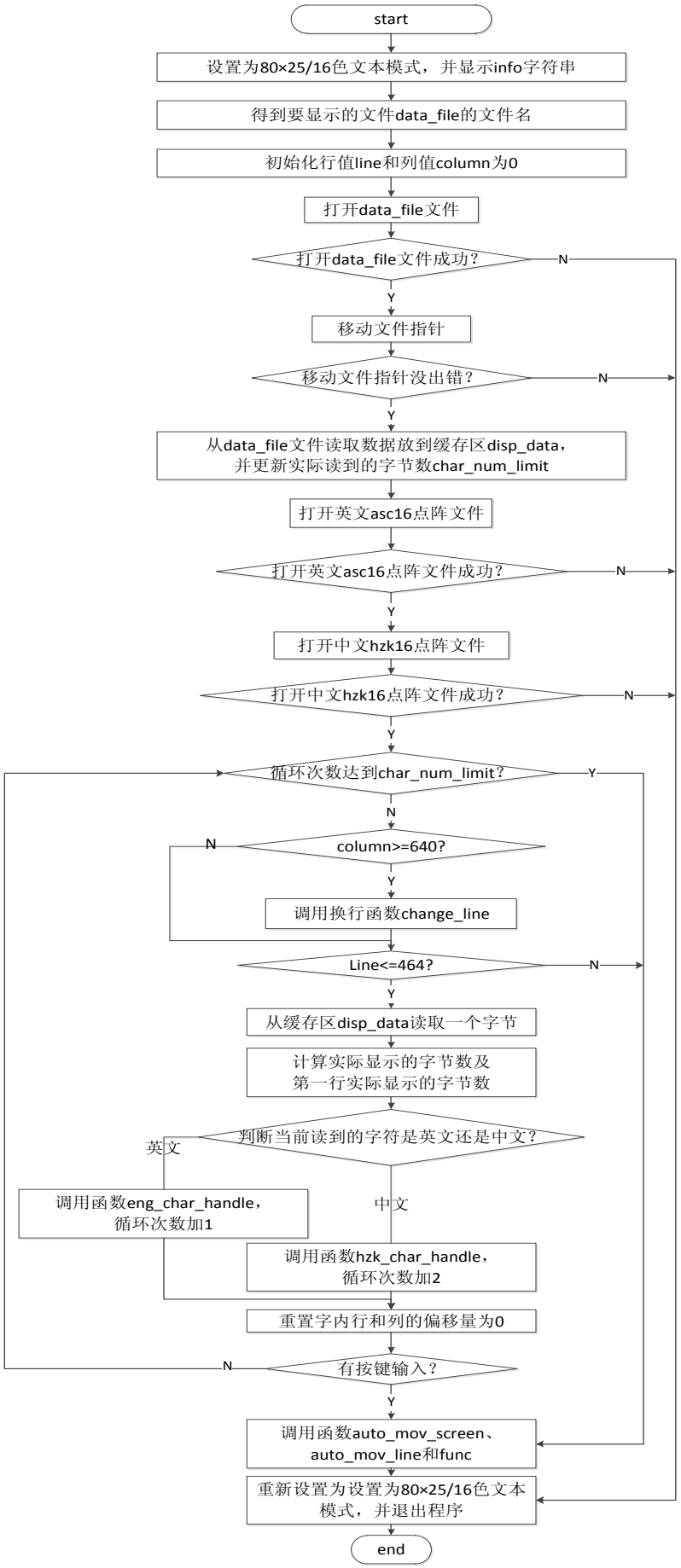
变量 line_interval 定义了字符间的行间距，每次换行时行值需增加(16+line_interval)。需要注意的是需要对 column_interval 进行非负判断以保证该变量不小于 0，否则上下两行的字会有重叠。

3.10 改变字体颜色

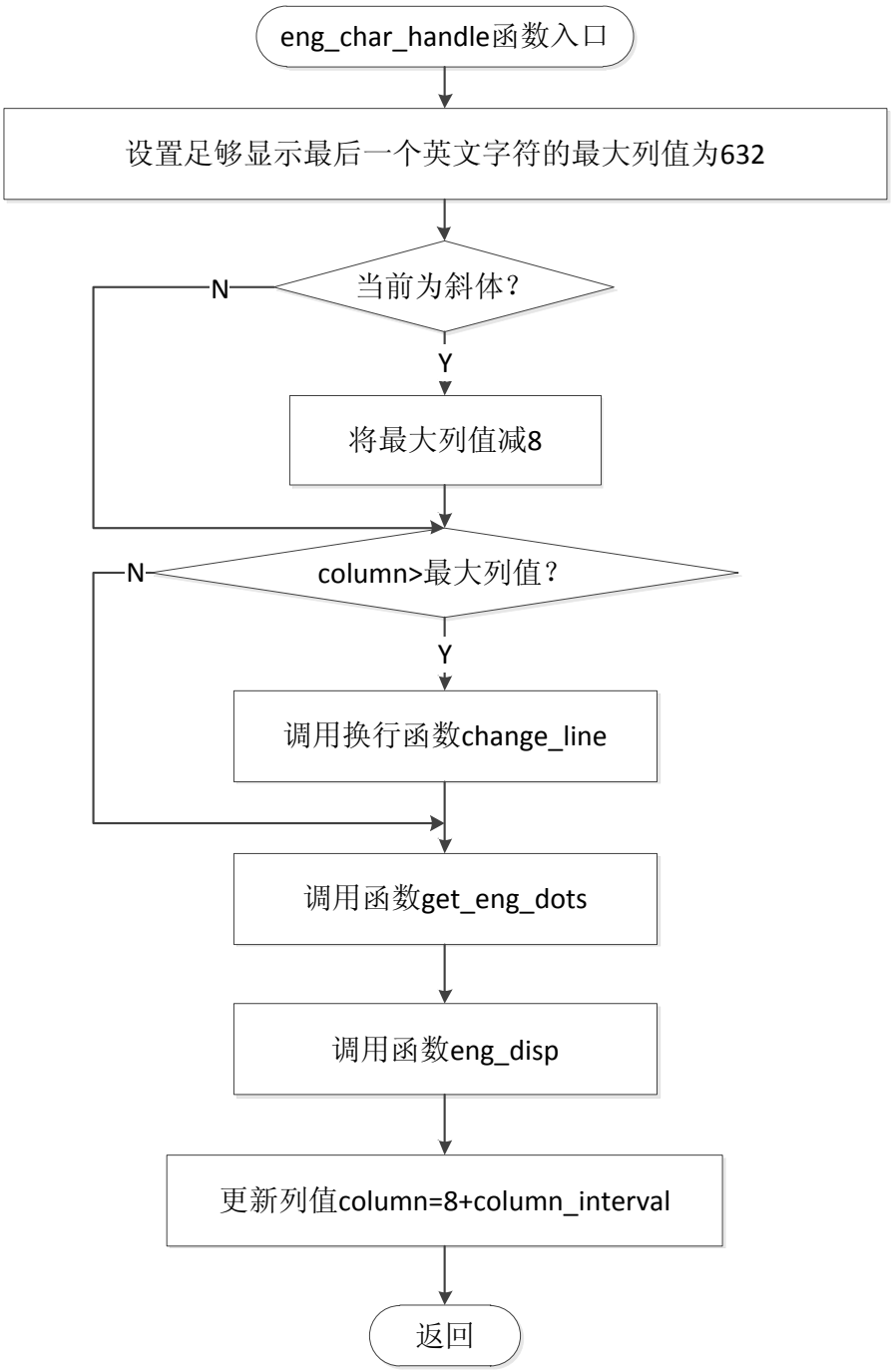
字体颜色定义在变量 char_color 中，只需要改变 char_color 的值再重新扫描即可。需要注意的是当 char_color 为 16 的倍数时，字体颜色为黑色，与背景色同色，此时无法显示，所以需要改变时 16 倍数的 char_color 的值。

四、主要过程程序框图

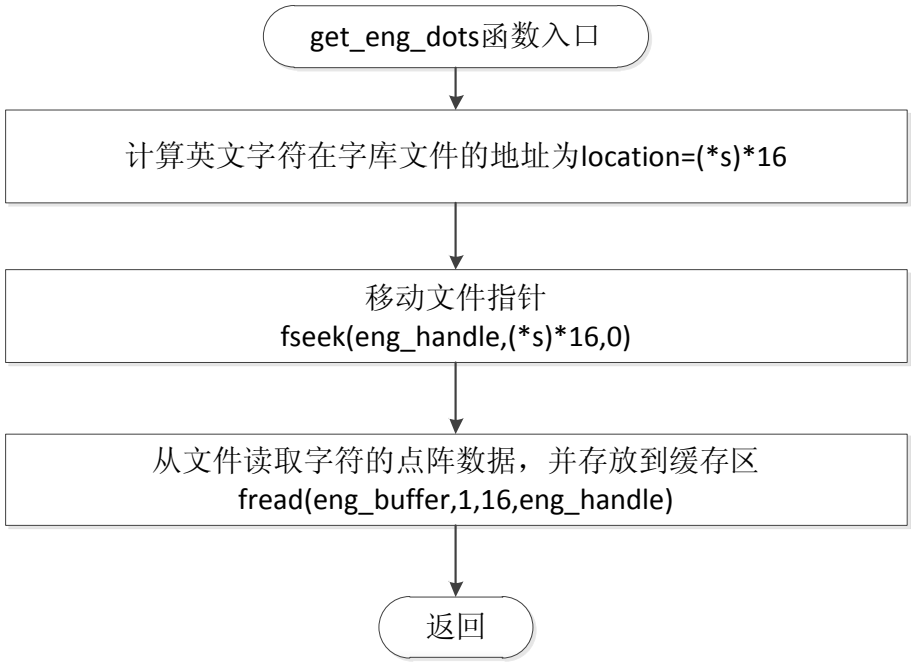
4.1 主程序的程序框图


start
设置为80×25/16色文本模式，并显示info字符串
得到要显示的文件data_file的文件名
初始化行值line和列值column为0
打开data_file文件
打开data_file文件成功?
N
Y
移动文件指针
移动文件指针没出错?
N
Y
从data_file文件读取数据放到缓存区disp_data，
并更新实际读到的字节数char_num_limit
打开英文asc16点阵文件
打开英文asc16点阵文件成功?
N
Y
打开中文hzk16点阵文件
打开中文hzk16点阵文件成功?
N
Y
循环次数达到char_num_limit?
Y
N
column>=640?
N
Y
调用换行函数change_line
Line<=464?
N
Y
从缓存区disp_data读取一个字节
计算实际显示的字节数及
第一行实际显示的字节数
判断当前读到的字符是英文还是中文?
英文
中文
调用函数eng_char_handle，
循环次数加1
调用函数hzk_char_handle，
循环次数加2
重置字内行和列的偏移量为0
有按键输入?
N
Y
调用函数auto_mov_screen、
auto_mov_line和func
重新设置为设置为80×25/16色文本
模式，并退出程序
end

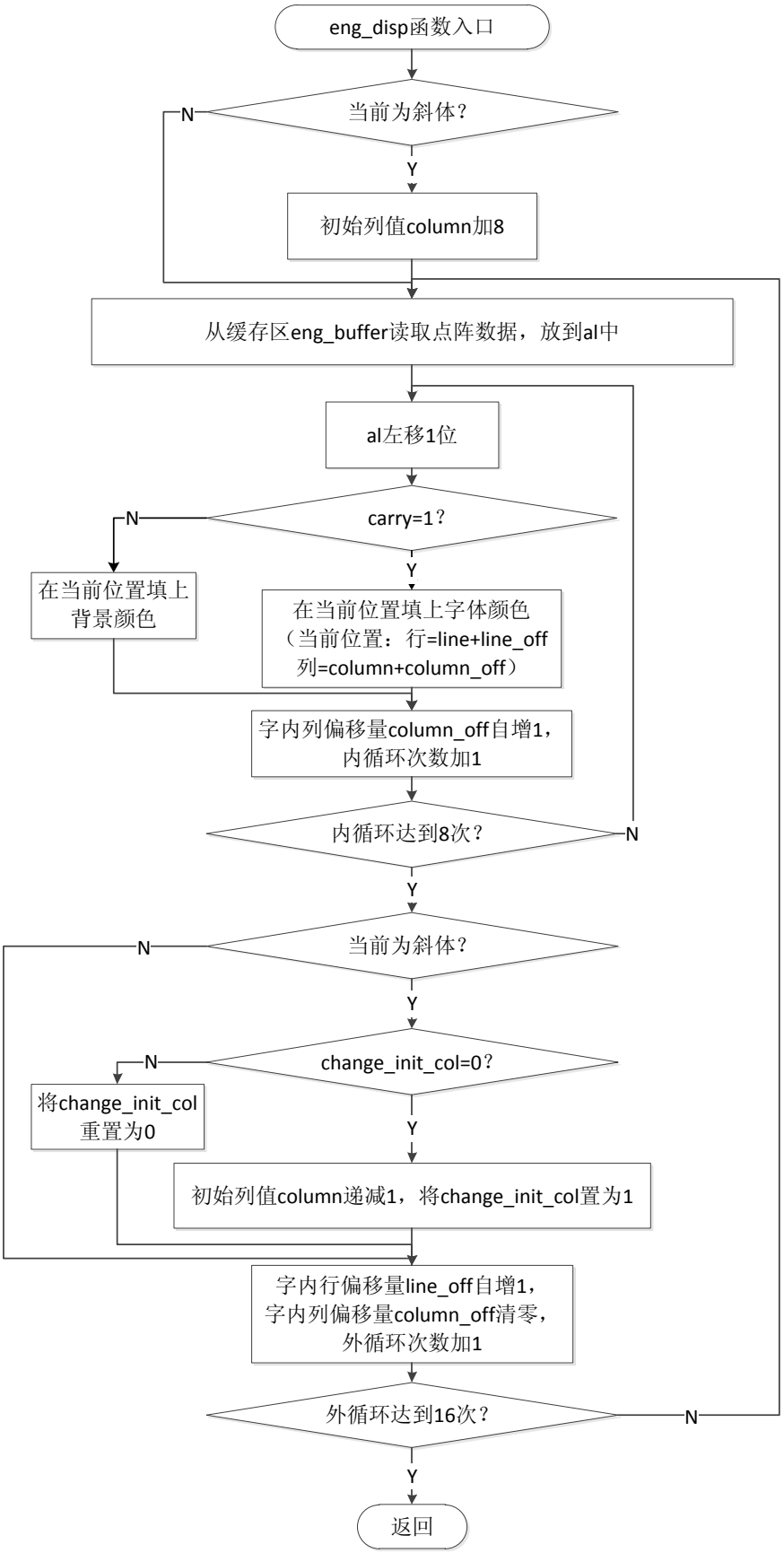
4.2 英文字符处理函数 eng_char_handle 的程序框图



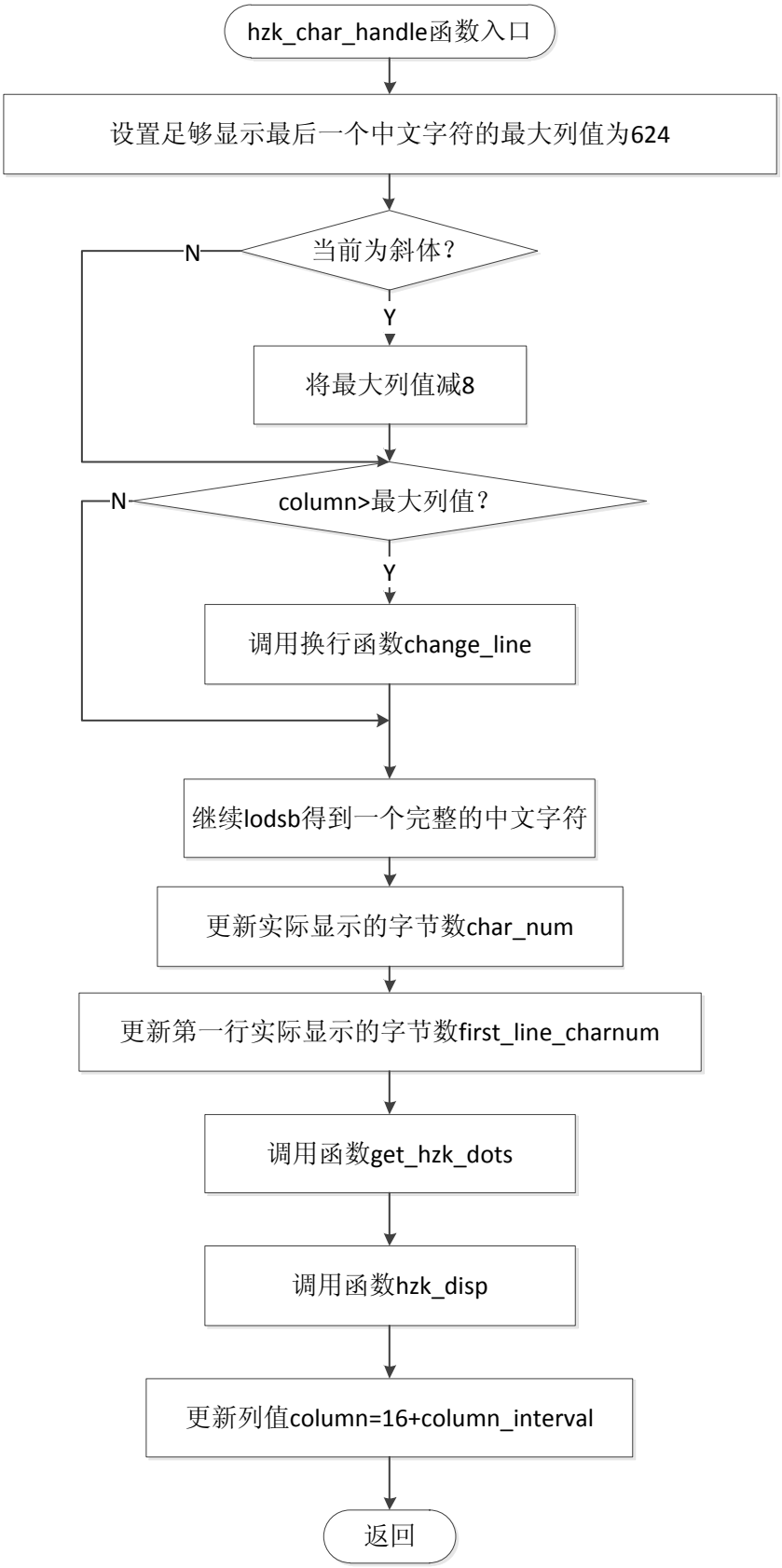
4.3 得到英文字符点阵数据的函数 get_eng_dots 的程序框图



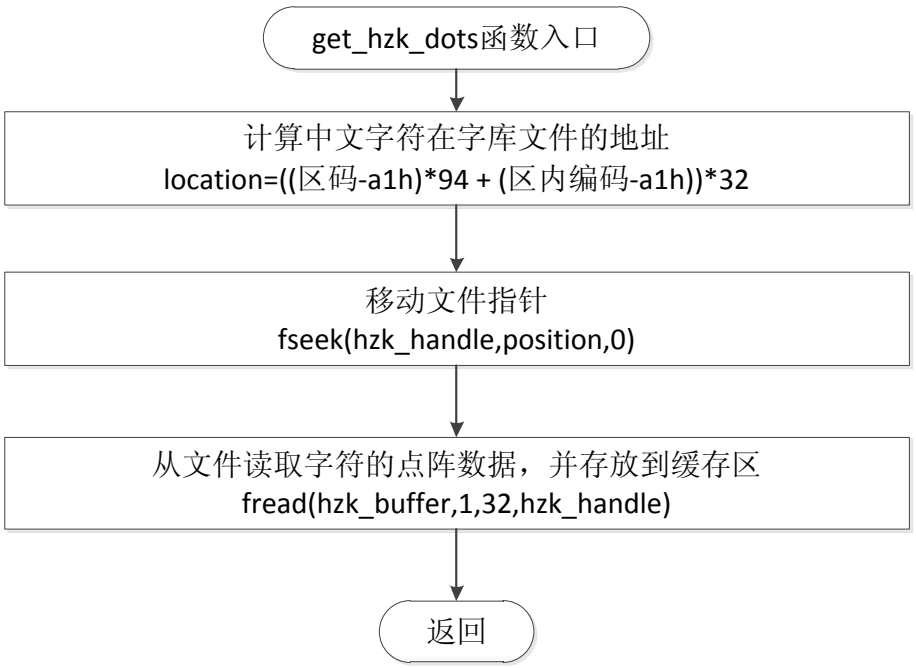
4.4 显示英文字符函数 eng_disp 的程序框图



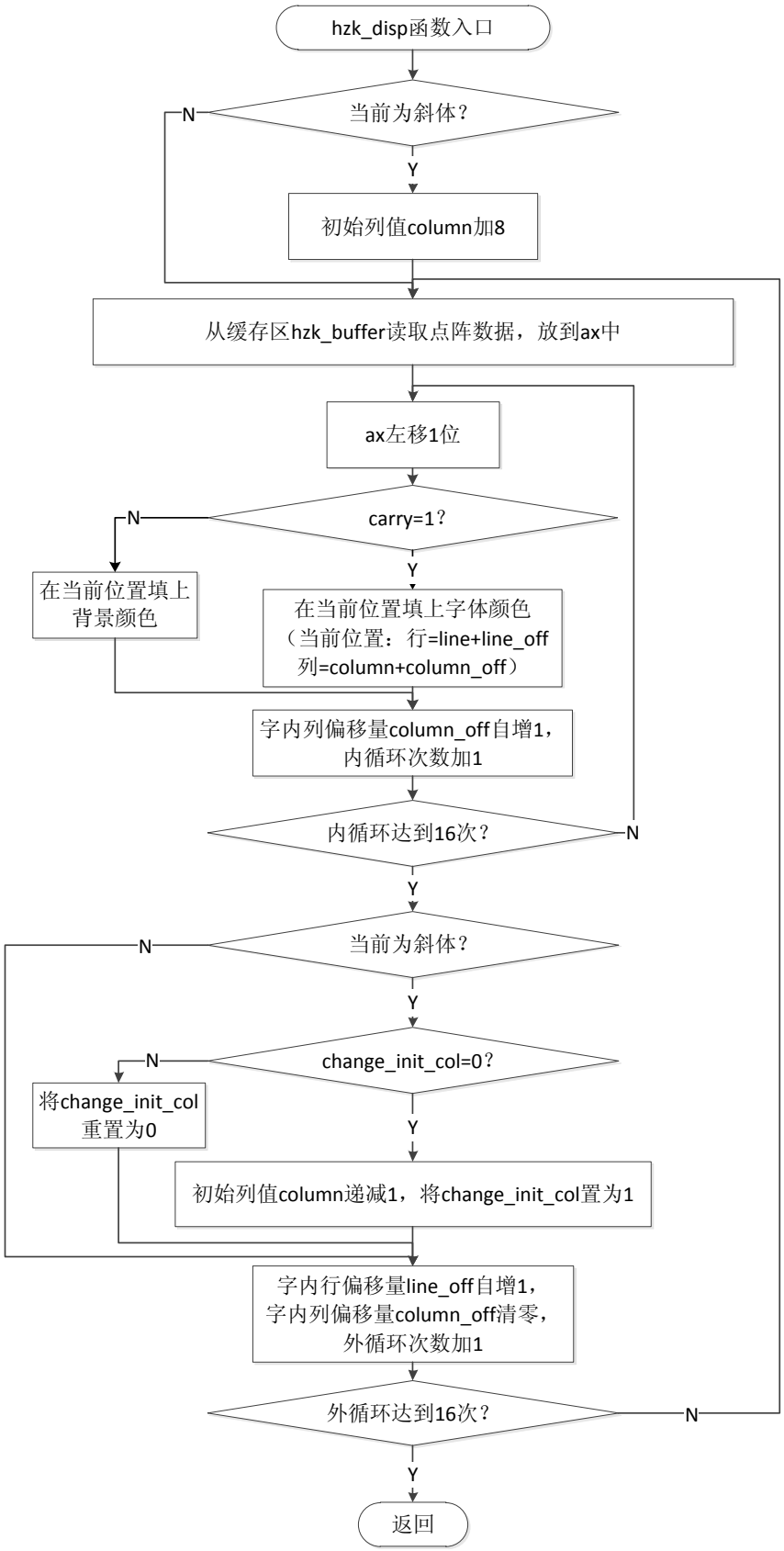
4.5 中文字符处理函数 hzk_char_handle 的程序框图



4.6 得到中文字符点阵数据的函数 get_hzk_dots 的程序框图

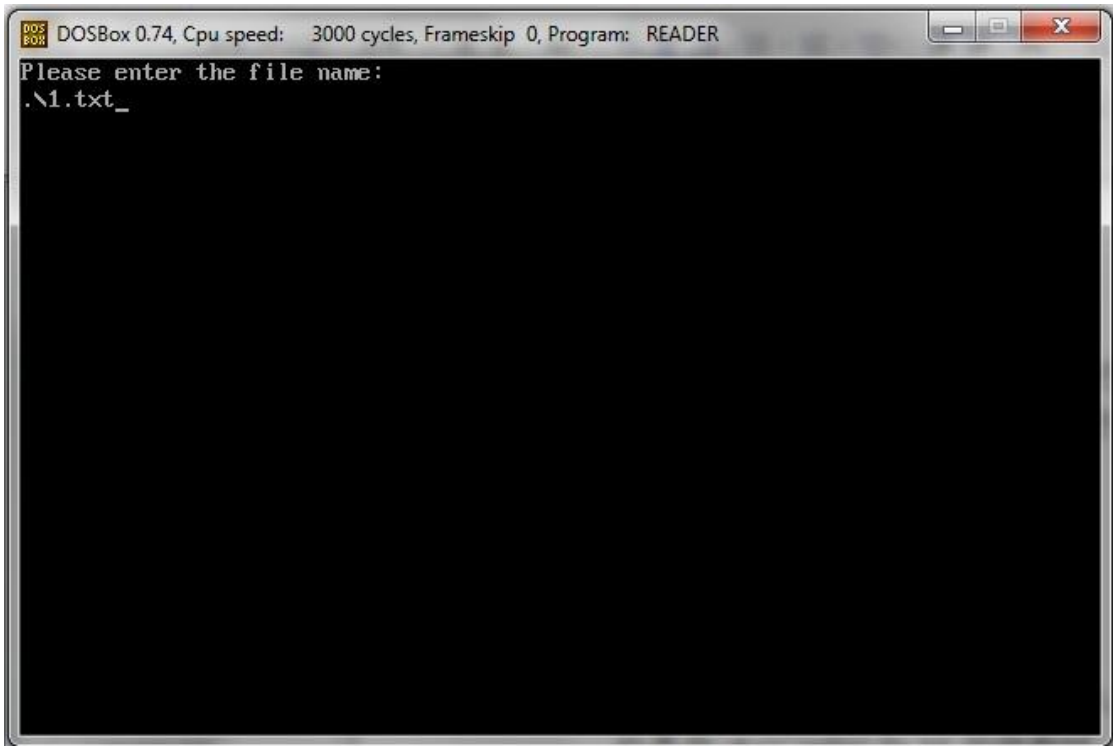


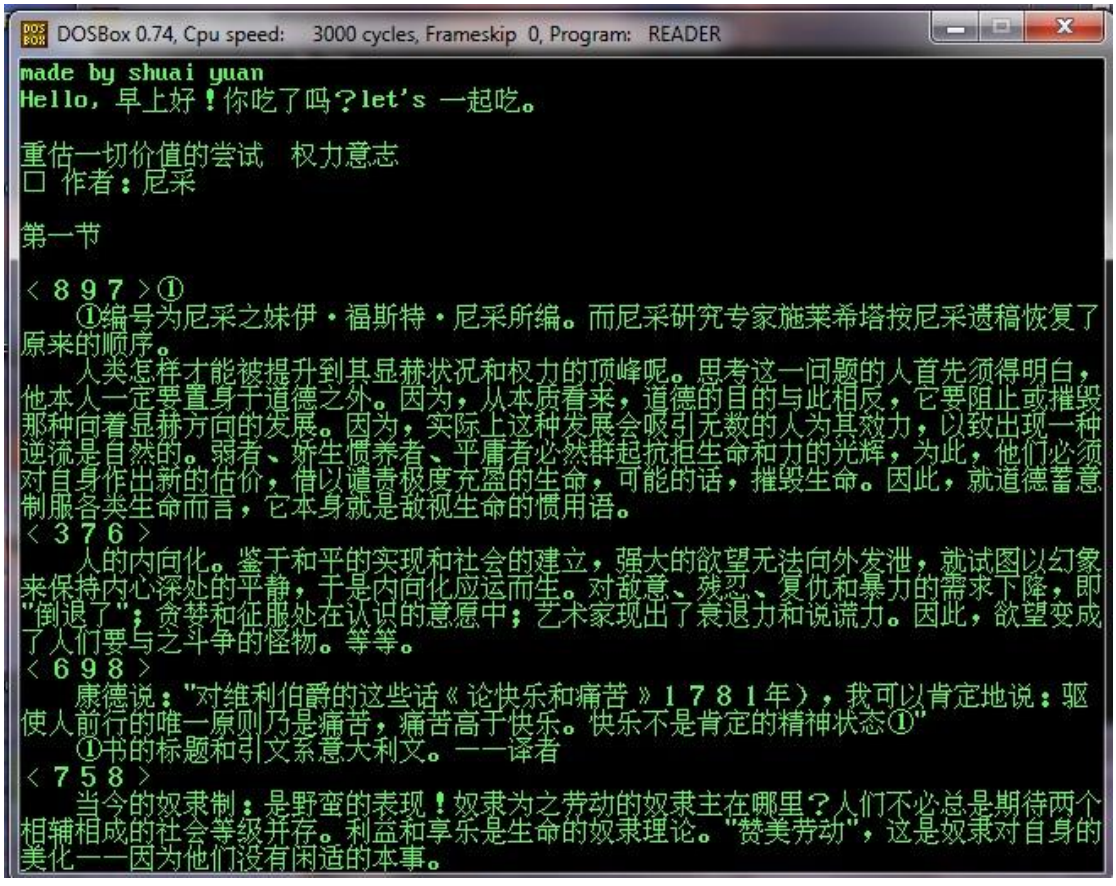
4.7 显示中文字符函数 hzk_disp 的程序框图



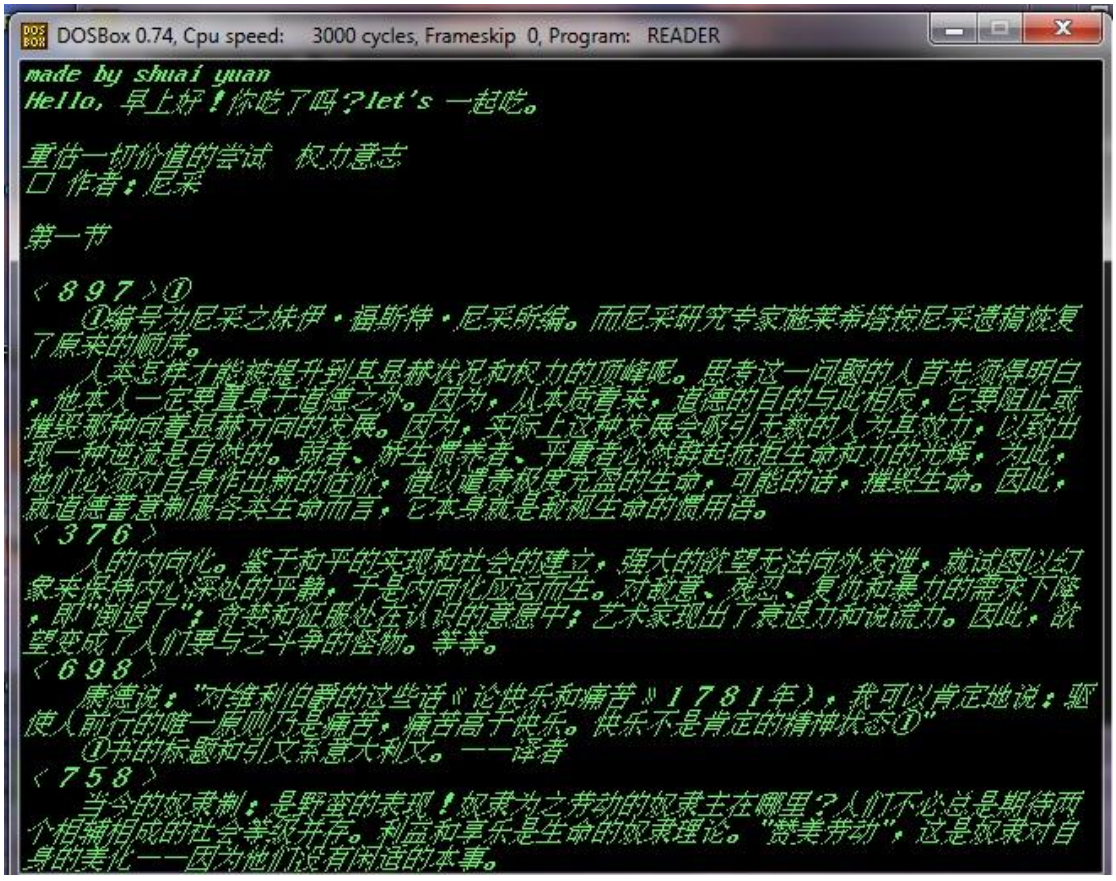
五、运行结果

下面是用 DOSBox 运行该程序部分功能的截图：
选择文件并进行显示：





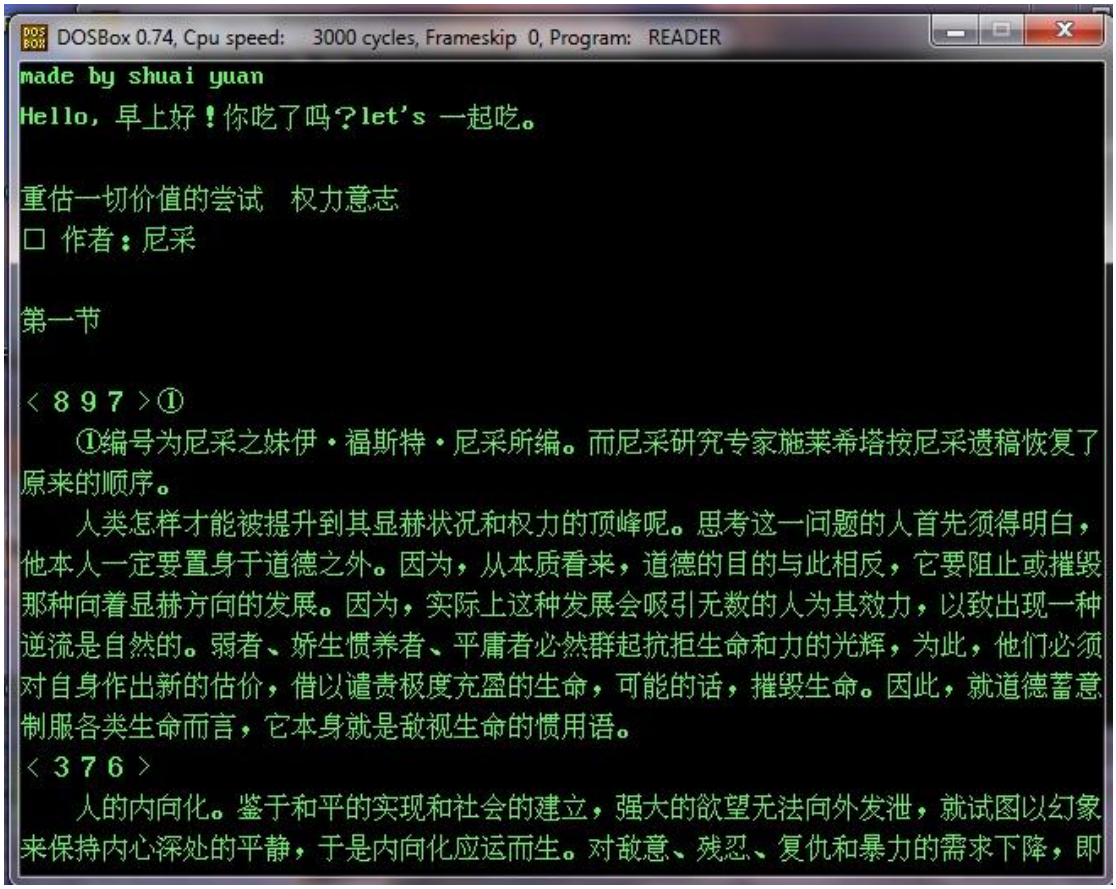
斜体字：


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: READER
made by shuai yuan
Hello, 早上好！你吃了吗？let's 一起吃。
重估一切价值的尝试 权力意志
□ 作者：尼采
第一节
〈897〉①
①编号为尼采之妹伊·福斯特·尼采所编。而尼采研究专家施莱希塔按尼采遗稿恢复了原来的顺序。
〈376〉
〈698〉
康德说："对维利伯爵的这些话《论快乐和痛苦》1781年），我可以肯定地说：驱使人前行的唯一原则乃是痛苦，痛苦高于快乐。快乐不是肯定的精神状态①"
①书的标题和引文系意大利文。——译者
〈758〉

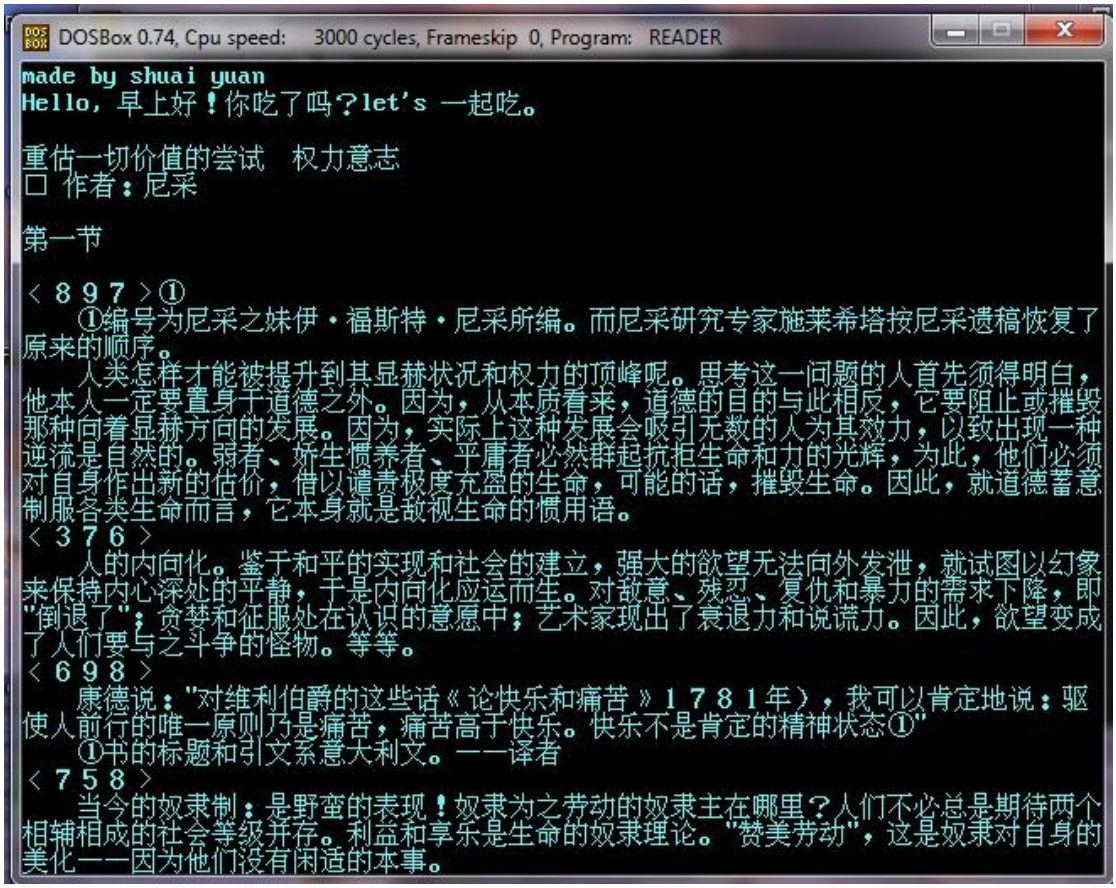
增加列间距：


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: READER
made by shuai yuan
Hello, 早上好！你吃了吗？let's 一起吃。
重估一切价值的尝试 权力意志
□ 作者：尼采
第一节
〈897〉①
①编号为尼采之妹伊·福斯特·尼采所编。而尼采研究专家施莱希塔按尼采遗稿恢复了原来的顺序。
〈376〉
〈698〉
康德说："对维利伯爵的这些话《论快乐和痛苦》1781年），我可以肯定地说：驱使人前行的唯一原则乃是痛苦，痛苦高于快乐。快乐不是肯定的精神状态①"
①书的标题和引文系意大利文。——译者
〈758〉

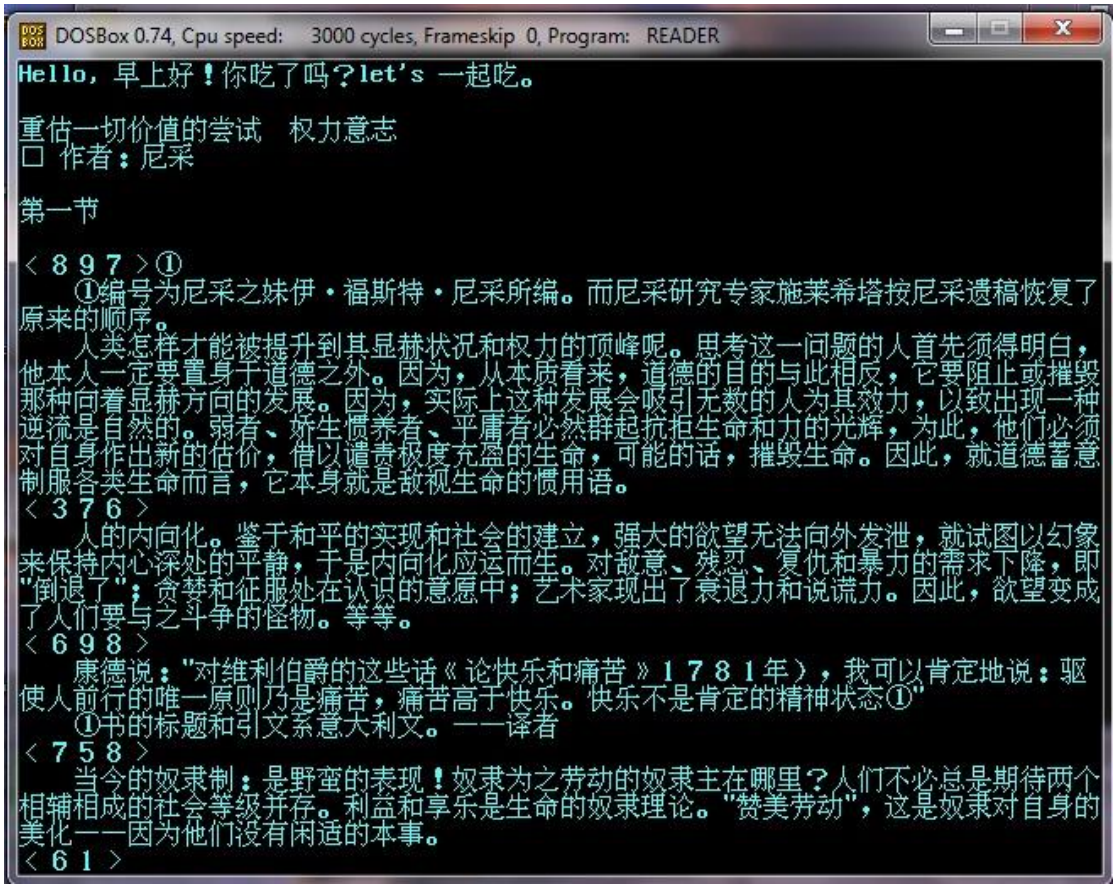
增加行间距：



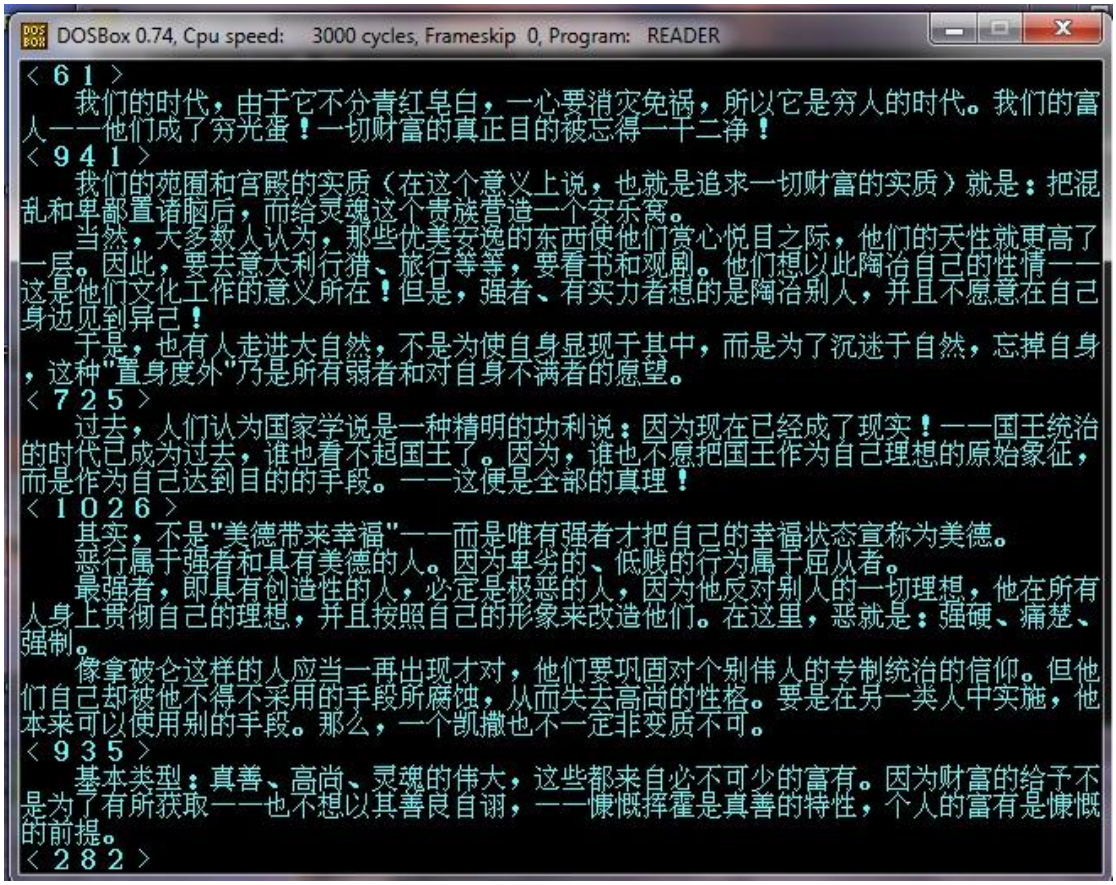
改变字体颜色：


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: READER
made by shuai yuan
Hello, 早上好！你吃了吗？let's 一起吃。
重估一切价值的尝试 权力意志
□ 作者：尼采
第一节
〈 8 9 7 〉①
①编号为尼采之妹伊·福斯特·尼采所编。而尼采研究专家施莱希塔按尼采遗稿恢复了原来的顺序。
人类怎样才能被提升到其显赫状况和权力的顶峰呢。思考这一问题的人首先须得明白，他本人一定要置身于道德之外。因为，从本质看来，道德的目的与此相反，它要阻止或摧毁那种向着显赫方向的发展。因为，实际上这种发展会吸引无数的人为其效力，以致出现一种逆流是自然的。弱者、娇生惯养者、平庸者必然群起抗拒生命和力的光辉，为此，他们必须对自身作出新的估价，借以谴责极度充盈的生命，可能的话，摧毁生命。因此，就道德蓄意制服各类生命而言，它本身就是敌视生命的惯用语。
〈 3 7 6 〉
人的内向化。鉴于和平的实现和社会的建立，强大的欲望无法向外发泄，就试图以幻象来保持内心深处的平静，于是内向化应运而生。对敌意、残忍、复仇和暴力的需求下降，即"倒退了"；贪婪和征服处在认识的意愿中；艺术家现出了衰退力和说谎力。因此，欲望变成了人们要与之斗争的怪物。等等。
〈 6 9 8 〉
康德说："对维利伯爵的这些话《论快乐和痛苦》1 7 8 1年），我可以肯定地说：驱使人前行的唯一原则乃是痛苦，痛苦高于快乐。快乐不是肯定的精神状态①"
①书的标题和引文系意大利文。——译者
〈 7 5 8 〉
当今的奴隶制：是野蛮的表现！奴隶为之劳动的奴隶主在哪里？人们不必总是期待两个相辅相成的社会等级并存。利益和享乐是生命的奴隶理论。"赞美劳动"，这是奴隶对自身的美化——因为他们没有闲适的本事。

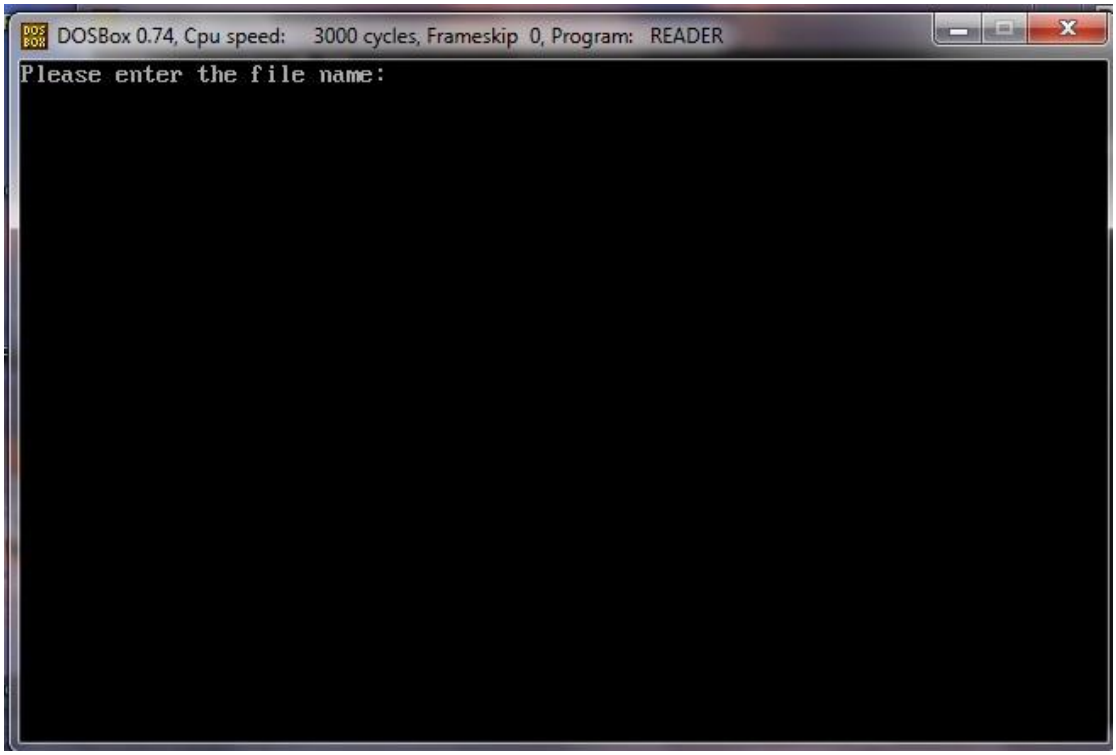
跳转到下一行：


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: READER
Hello, 早上好！你吃了吗？let's 一起吃。
重估一切价值的尝试 权力意志
□ 作者：尼采
第一节
〈 8 9 7 〉①
①编号为尼采之妹伊·福斯特·尼采所编。而尼采研究专家施莱希塔按尼采遗稿恢复了原来的顺序。
人类怎样才能被提升到其显赫状况和权力的顶峰呢。思考这一问题的人首先须得明白，他本人一定要置身于道德之外。因为，从本质看来，道德的目的与此相反，它要阻止或摧毁那种向着显赫方向的发展。因为，实际上这种发展会吸引无数的人为其效力，以致出现一种逆流是自然的。弱者、娇生惯养者、平庸者必然群起抗拒生命和力的光辉，为此，他们必须对自身作出新的估价，借以谴责极度充盈的生命，可能的话，摧毁生命。因此，就道德蓄意制服各类生命而言，它本身就是敌视生命的惯用语。
〈 3 7 6 〉
人的内向化。鉴于和平的实现和社会的建立，强大的欲望无法向外发泄，就试图以幻象来保持内心深处的平静，于是内向化应运而生。对敌意、残忍、复仇和暴力的需求下降，即"倒退了"；贪婪和征服处在认识的意愿中；艺术家现出了衰退力和说谎力。因此，欲望变成了人们要与之斗争的怪物。等等。
〈 6 9 8 〉
康德说："对维利伯爵的这些话《论快乐和痛苦》1 7 8 1年），我可以肯定地说：驱使人前行的唯一原则乃是痛苦，痛苦高于快乐。快乐不是肯定的精神状态①"
①书的标题和引文系意大利文。——译者
〈 7 5 8 〉
当今的奴隶制：是野蛮的表现！奴隶为之劳动的奴隶主在哪里？人们不必总是期待两个相辅相成的社会等级并存。利益和享乐是生命的奴隶理论。"赞美劳动"，这是奴隶对自身的美化——因为他们没有闲适的本事。
〈 6 1 〉

跳转到下一屏：



按 q 跳出，回到输入文件名界面：



六、设计总结

在做大程时遇到了不少细节上的问题。例如之前出现过在增加列间距后换行后的字符显示重复的问题，后来发现时换行函数 change_line 没有保存寄存器 ax，导致显示的字符错了。一开始在写显示中文字符函数 hzk_disp 和显示英文字符函数 eng_disp 时我用了比较多的寄存器，需要很多的 push 和 pop 指令，不仅容易出问题而且较难排查错误，所以后来定义了 line_off 和 column_off 等变量来避免这样的问题。此外，判断当前是否应该换行及字符应不应该显示也是个需要足够小心的细节问题，尤其当行间距和列间距可以调整后，字符显示是否完整的问题就很突出了。

在最后我也对代码进行了重构，将一些代码块整合成函数，并用 macro 去除代码的冗余，使得代码的模块更加清晰，方便修改和移植。

在这个程序中，我认为（自动）换行和换屏功能是一大亮点，但实现的思路其实是很简单的，只需要找到文件指针的新的偏移量再重新显示。但目前向上翻行\屏的功能还不完全，因为这实现起来要比向下翻复杂得多。我的思路是：跳转到向上行首先需要读出前 80 个 byte（80 是一行的字节数，显然也是一行所能显示的字节数上界），如果没有回车换行则直接显示，有回车换行的话还需要再往前移动文件指针，知道找到上一个回车换行或找不到回车换行而来到文件头，然后计算这部分的字节数，并由此求出这部分最后一行（即当前的上一行）实际显示的字符数；向上翻屏也是先要读出前 2400 个 byte，再根据这部分字符串中存在的回车换行进行类似的计算和处理。由于时间比较仓促，这部分功能还有待进一步实现。