**中山大学数据科学与计算机学院本科生实验报告**

**（2020学年春季学期）**

课程名称：**操作系统实验**  任课教师：凌应标 助教：

|  |  |  |  |
| --- | --- | --- | --- |
| 年级&班级 | **2018级1班** | 专业(方向) | **计算机科学与技术(大数据方向)** |
| 学号 | **18308045** | 姓名 | **谷正阳** |
| 电话 | **13355426001** | Email | [**Guzy0324@163.com**](mailto:Guzy0324@163.com) |
| 开始日期 | **2020.5.1** | 完成日期 | **2020.5.6** |

**一、实验题目**

操作系统 实验2

1. **实验目的**

1、了解监控程序执行用户程序的主要工作

2、了解一种用户程序的格式与运行要求

3、加深对监控程序概念的理解

4、掌握加载用户程序方法

5、掌握几个BIOS调用和简单的磁盘空间管理

1. **实验要求：**

1、知道引导扇区程序实现用户程序加载的意义

2、掌握COM/BIN等一种可执行的用户程序格式与运行要求

3、将自己实验一的引导扇区程序修改为3-4个不同版本的COM格式程序，每个程序缩小显示区域，在屏幕特定区域显示，用以测试监控程序，在1.44MB软驱映像中存储这些程序。

4、重写1.44MB软驱引导程序，利用BIOS调用，实现一个能执行COM格式用户程序的监控程序。

5、设计一种简单命令，实现用命令交互执行在1.44MB软驱映像中存储几个用户程序。

6、编写实验报告，描述实验工作的过程和必要的细节，如截屏或录屏，以证实实验工作的真实性

1. **实验内容：**

**1. 实验步骤**

(1) 将自己实验一的引导扇区程序修改为一个的COM格式程序，程序缩小显示区域，在屏幕第一个1/4区域显示，显示一些信息后，程序会结束退出，可以在DOS中运行。在1.44MB软驱映像中制定一个或多个扇区，存储这个用户程序a。

相似地、将自己实验一的引导扇区程序修改为第二、第三、第四个的COM格式程序，程序缩小显示区域，在屏幕第二、第三、第四个1/4区域显示，在1.44MB软驱映像中制定一个或多个扇区，存储用户程序b、用户程序c、用户程序d。

(2) 重写1.44MB软驱引导程序，利用BIOS调用，实现一个能执行COM格式用户程序的监控程序。程序可以按操作选择，执行一个或几个用户程序。解决加载用户程序和返回监控程序的问题，执行完一个用户程序后，可以执行下一个。

(3)设计一种命令，可以在一个命令中指定某种顺序执行若干个用户程序。可以反复接受命令。

(4)在映像盘上，设计一个表格，记录盘上有几个用户程序，放在那个位置等等信息，如果可以，让监控程序显示出表格信息。

(5)拓展自己的软件项目管理目录，管理实验项目相关文档

**2.** 实验原理

1. 监控程序执行用户程序的主要工作：

运行监控程序→加载用户程序→运行用户程序→运行监控程序

2. COM/BIN/EXE可执行的用户程序格式和运行要求：

<https://www.cnblogs.com/arkhe/articles/2615928.html>

3. 掌握加载用户程序方法：

使用BIOS调用int 13将映像上指定扇区的用户程序加载到指定位置的内存中

4. BIOS调用和简单的磁盘空间管理：

<https://blog.csdn.net/weixin_37656939/article/details/79684611>

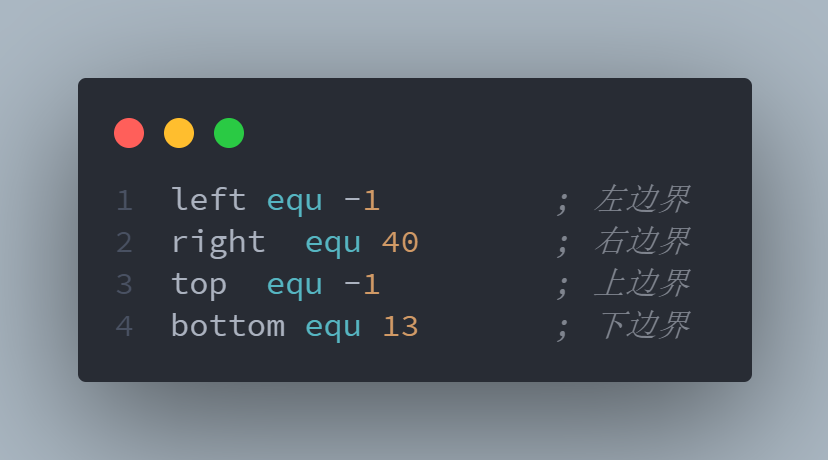
5. bochs使用：

<https://www.cnblogs.com/jikebiancheng/p/6160337.html>

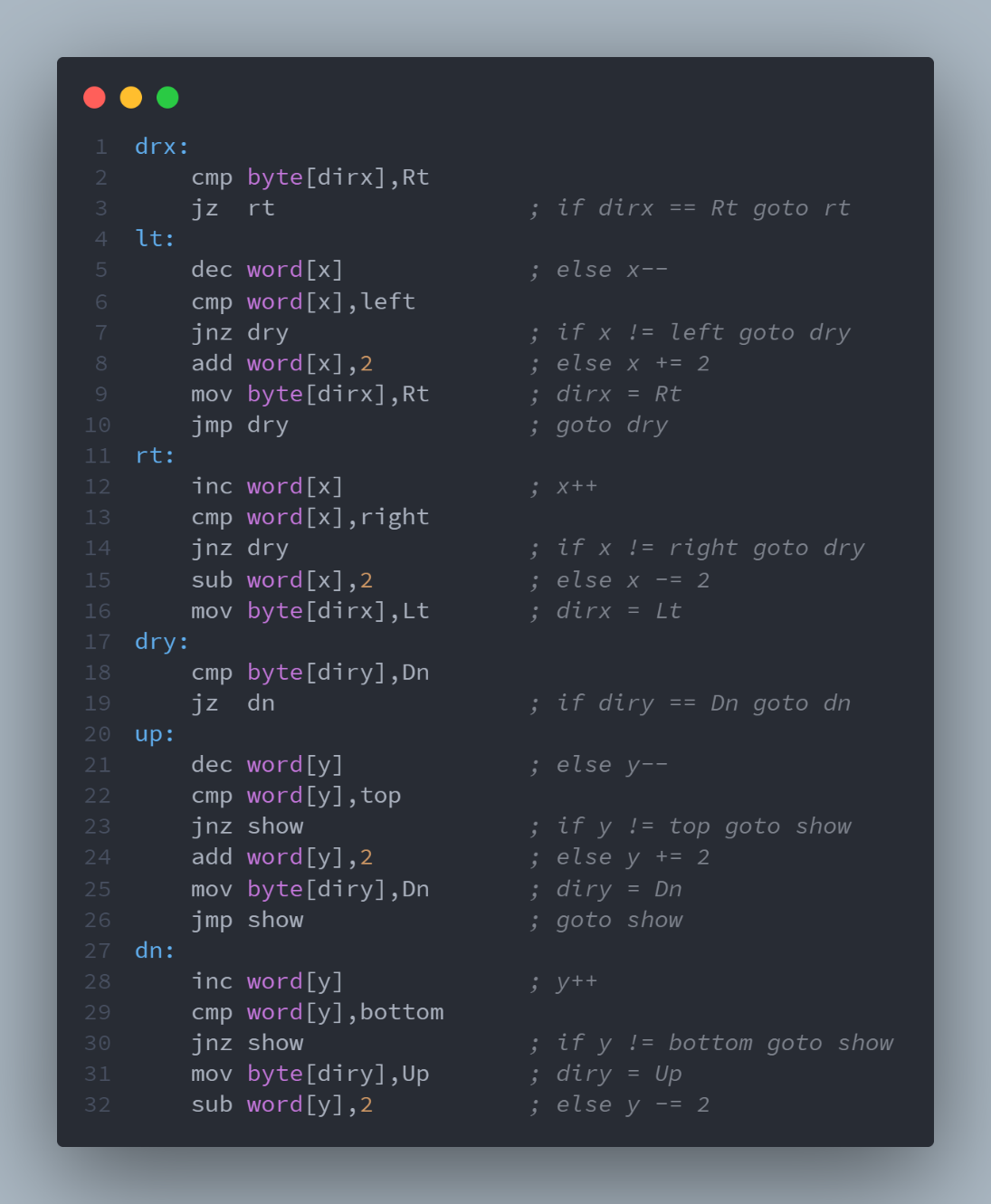
1. **实验结果**

1.

a. stoneN.asm，边界未定义，扩展性不够强，因而增加边界定义（修改后：stoneN00\_not\_used.asm

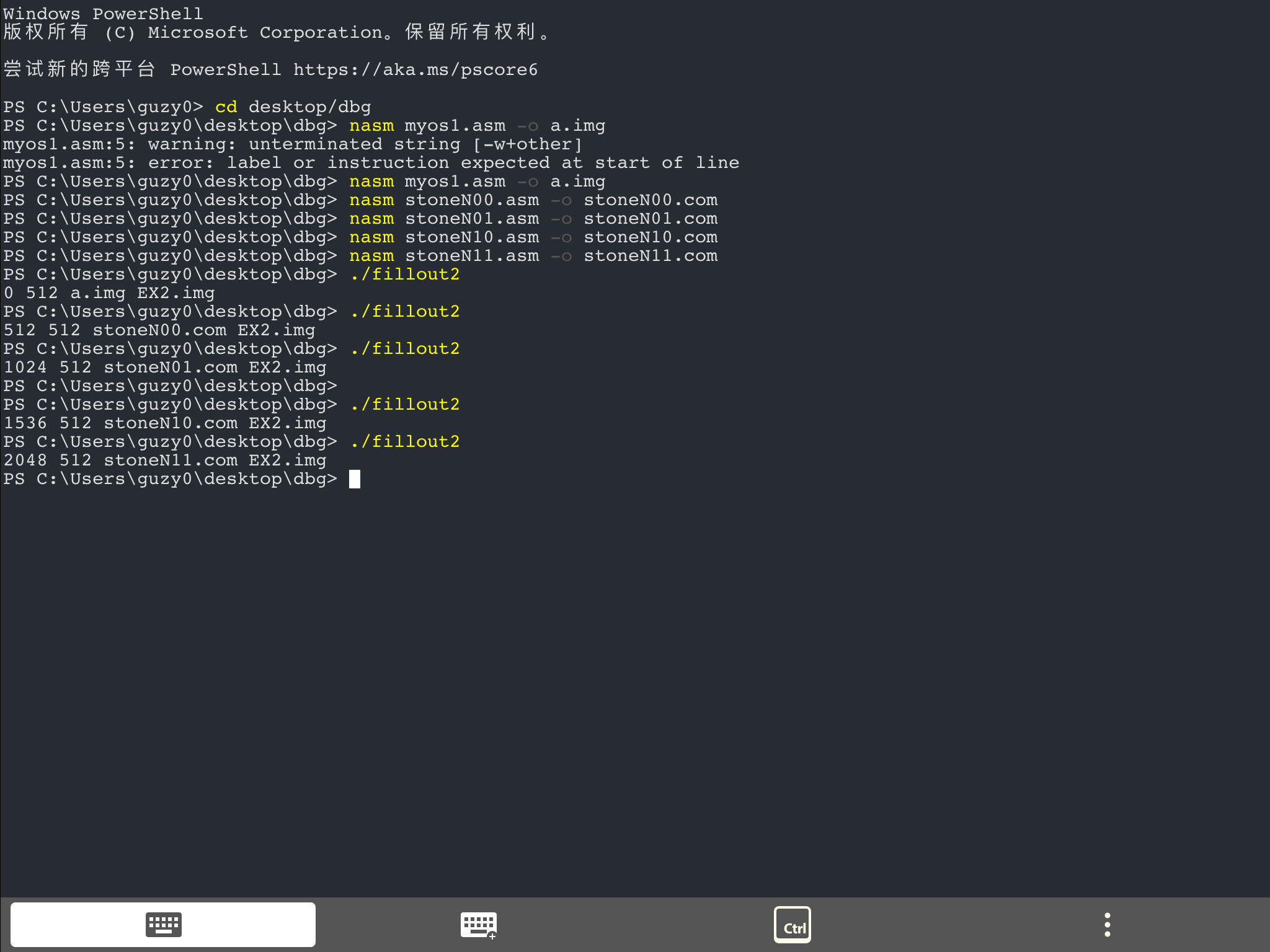


b. stoneN00\_not\_used.asm如果打到角上，无法原路反弹，原因是老师给的stoneN.asm只考虑碰边界的情况，未考虑碰角的情况。因而重写运动轨迹模块为横向判断一次左右，纵向判断一次上下

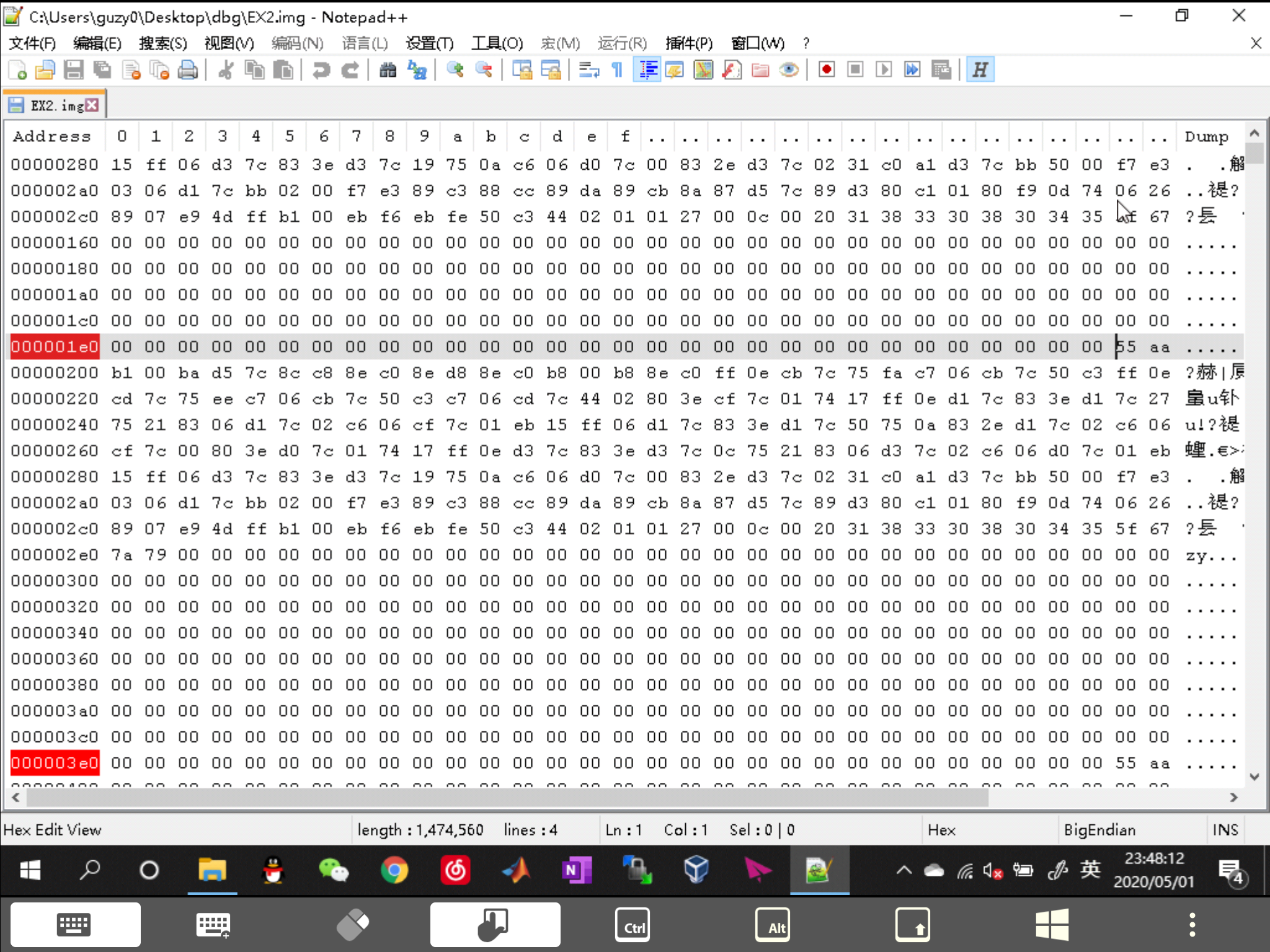


c. 之前写的fillout.cpp只能填充首扇区，扩展性不够强，因而增加两个输入分别是首字节位置和占用字节数

操作如下：



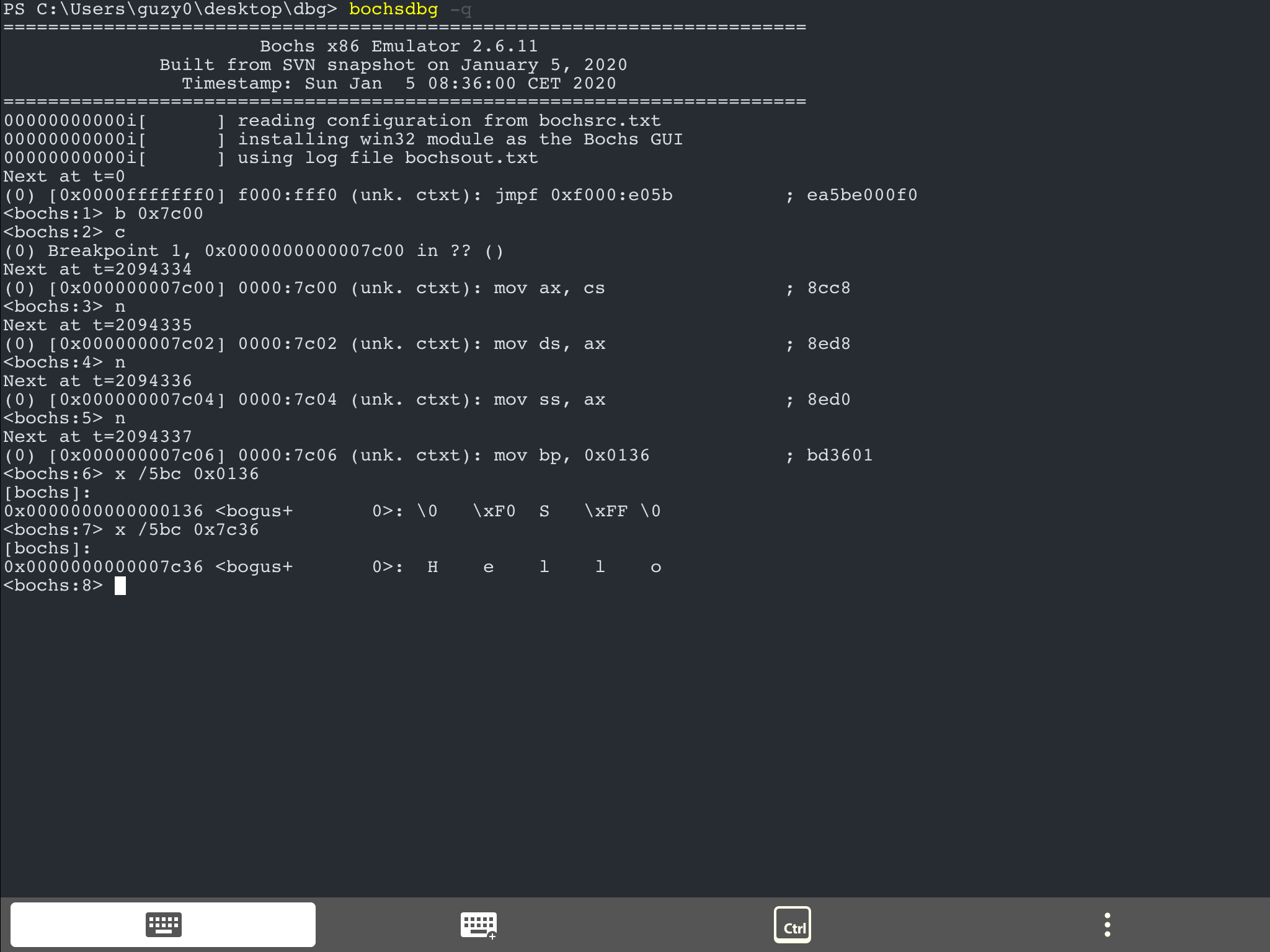
效果如下：



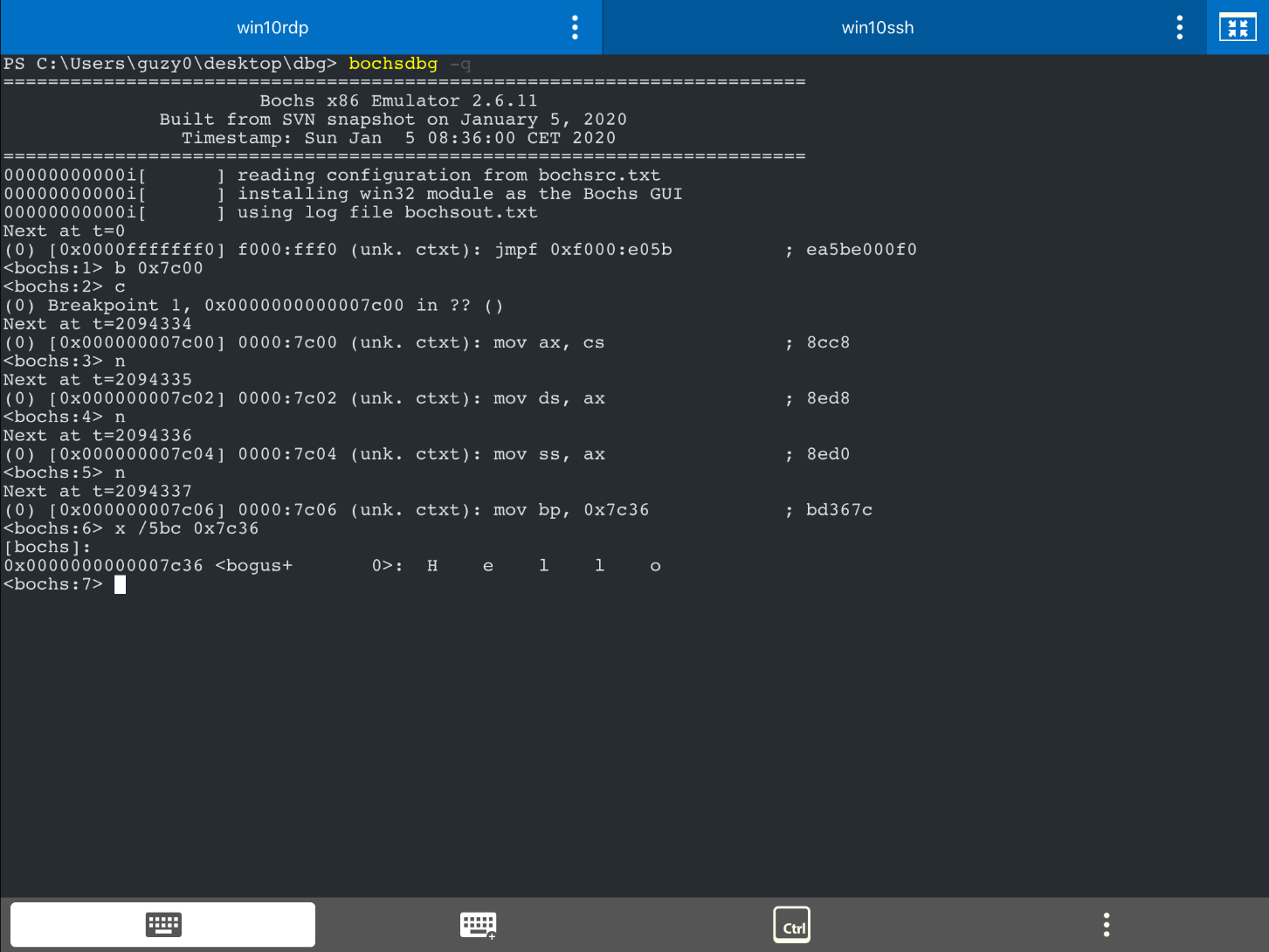
2.

a. myos1.asm中org的问题：org 100h填到首扇区无法正常显示Message。使用bochsdbg断点检查内存，发现org是用来调整虚拟地址的偏移

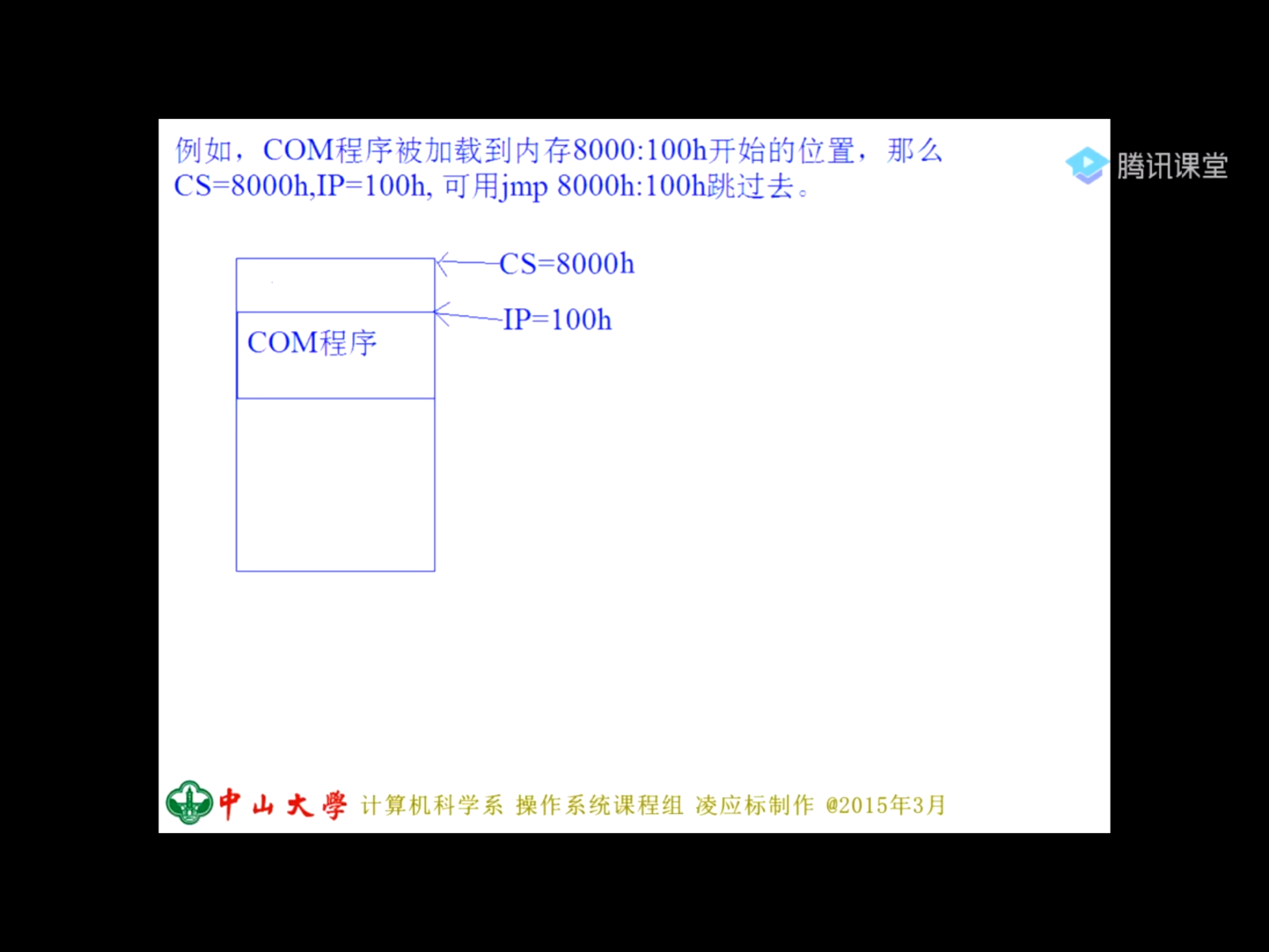
如下是偏移100h，发现虚拟地址被替换成了100h+36h，而物理地址为0x7c36，int 10输出的是物理地址0x0136的字符串



如下是偏移7c00h，输出正常



b. stoneN00.asm中org的问题：我最初是将其加上org 0x8100，虽然能正常运行，但是是不符合COM用户程序的要求的，COM用户程序要求偏移100h。这个问题从老师的ppt中，我得到了解答：



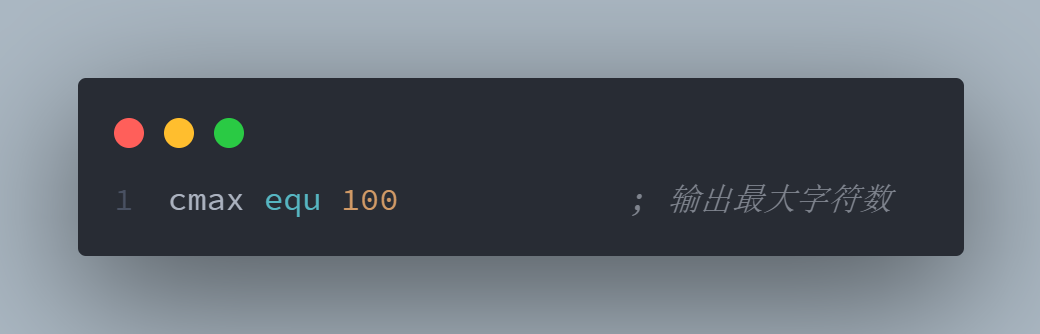
是段寄存器出了错（后面另一个bug对此有验证

而jmp 8000h:100h是可以自动修改段寄存器为8000h的

然而另一方面，段地址16位偏移量4位，一共20位。如此8100h对应段地址应为810h，而又有org 100h，所以段地址应为800h。综上，应该为



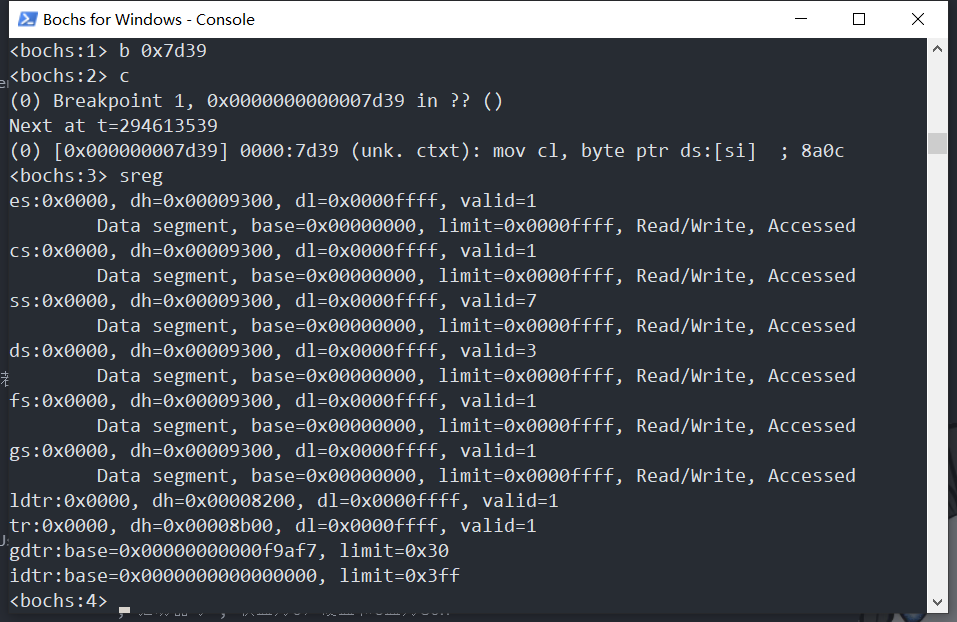
c. 用户程序无法结束，定义一个输出字符数最大值，如下：



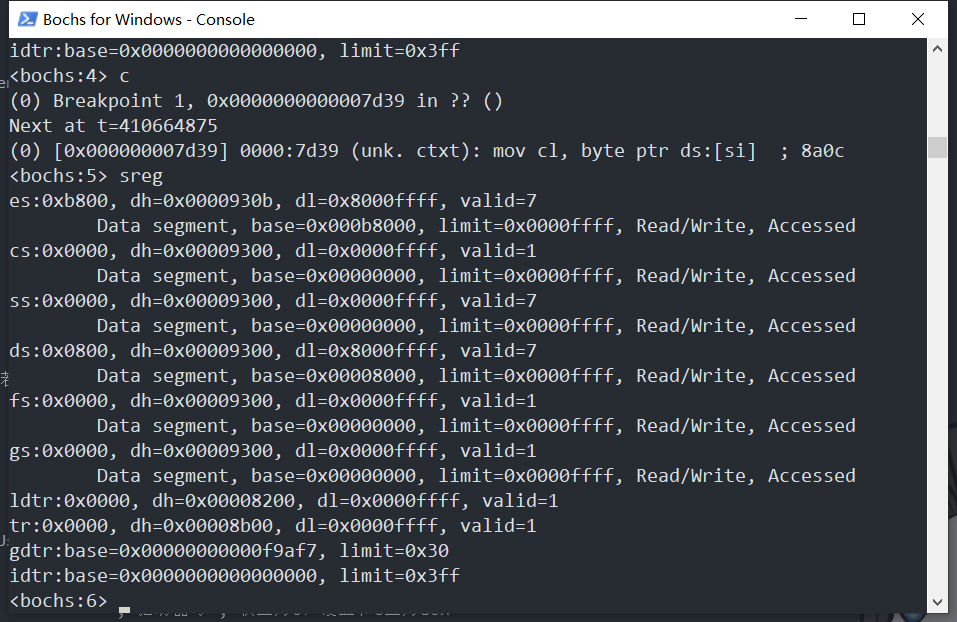
为了切换到下一个，将jmp改为call，加上retf

注意：retf不会改变DS和ES的值

call前：



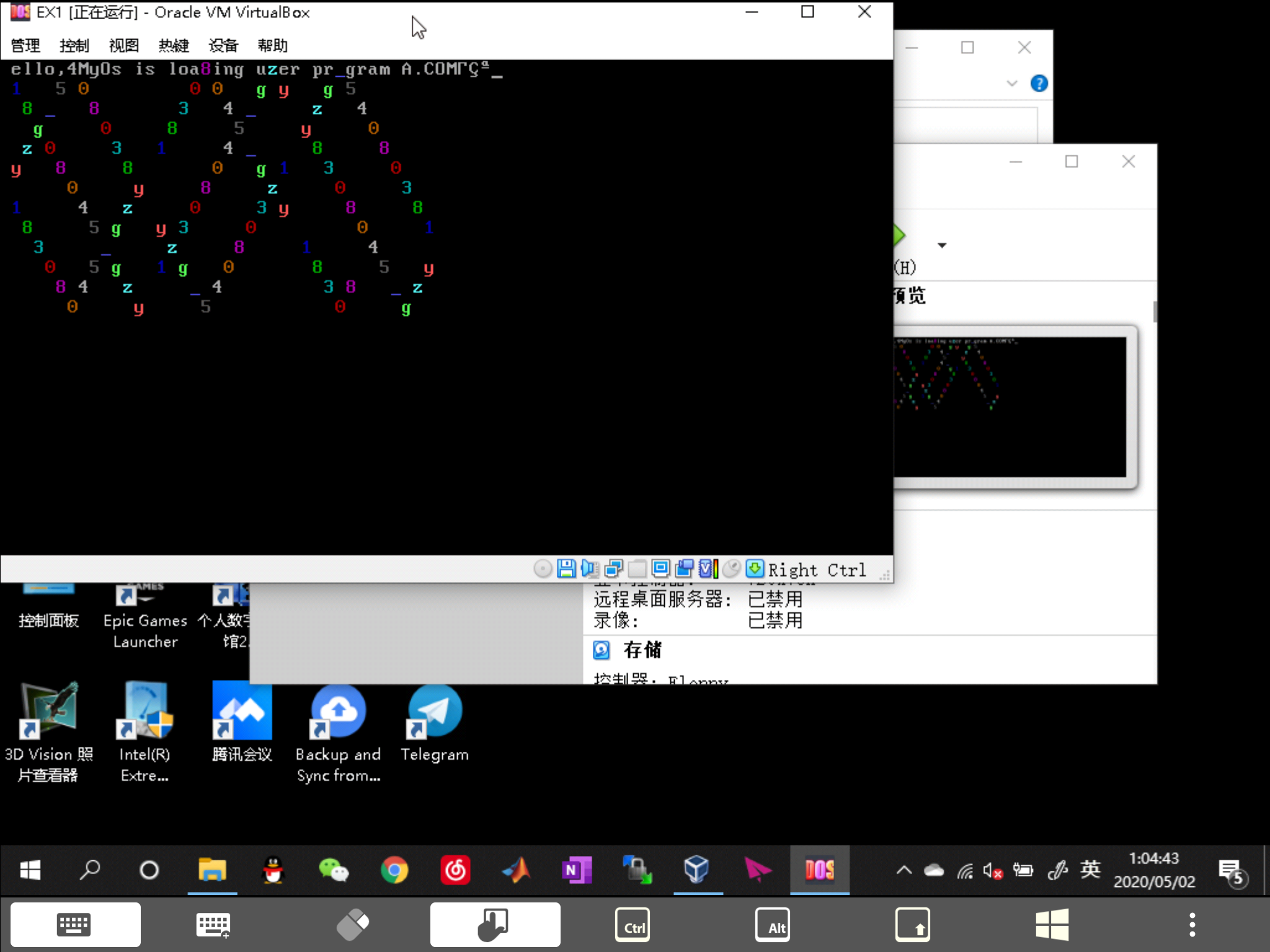
call后：

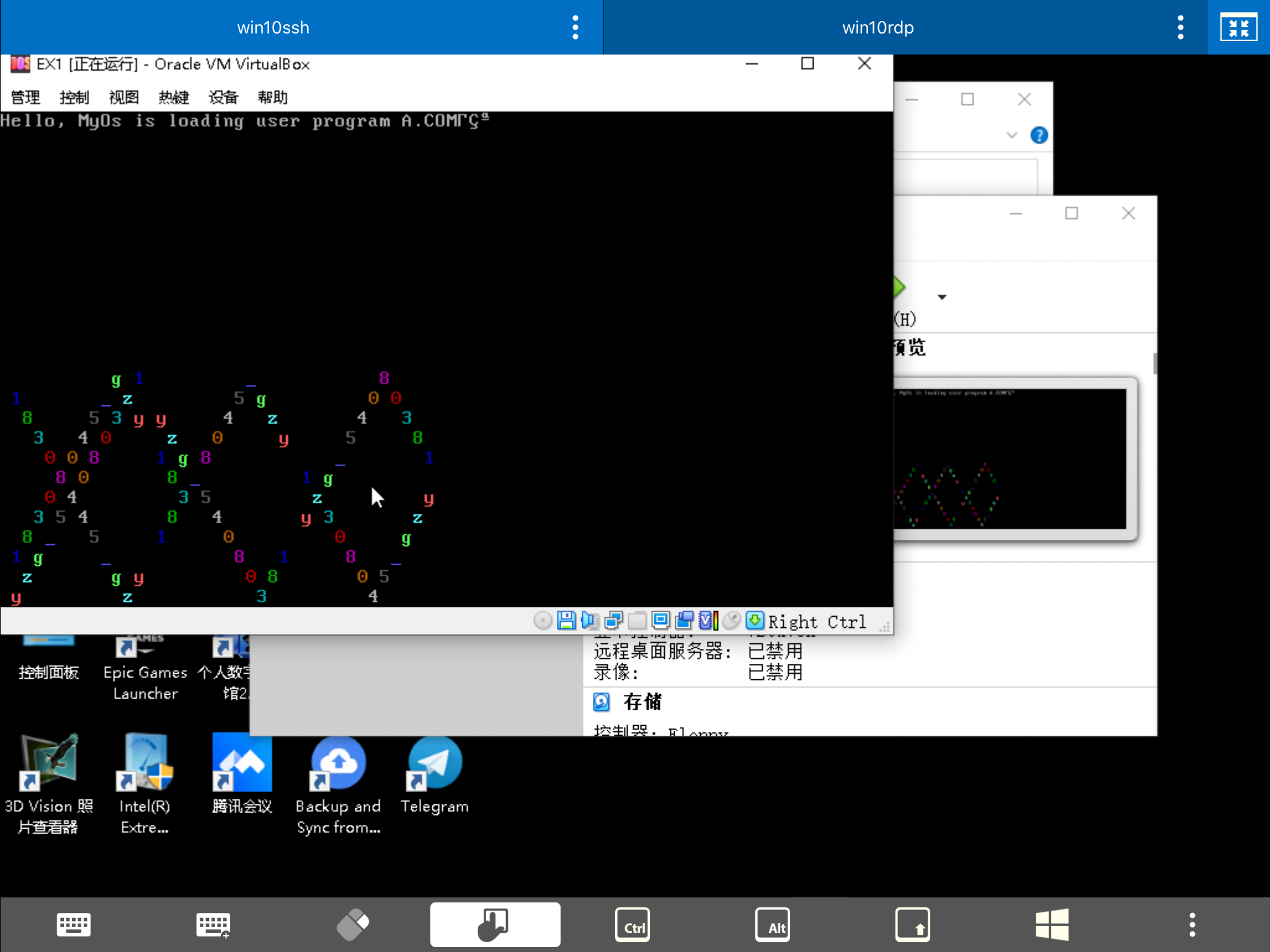


所以需要retf后手动修改：



e. 结果：





3. 使用int 10调整光标，打印字符，int 13接受字符的服务模拟输入的感觉：

说明：

↑：回到句首

↓：到句尾

←：左移一位

→：右移一位

退格：删除光标后的一个字符

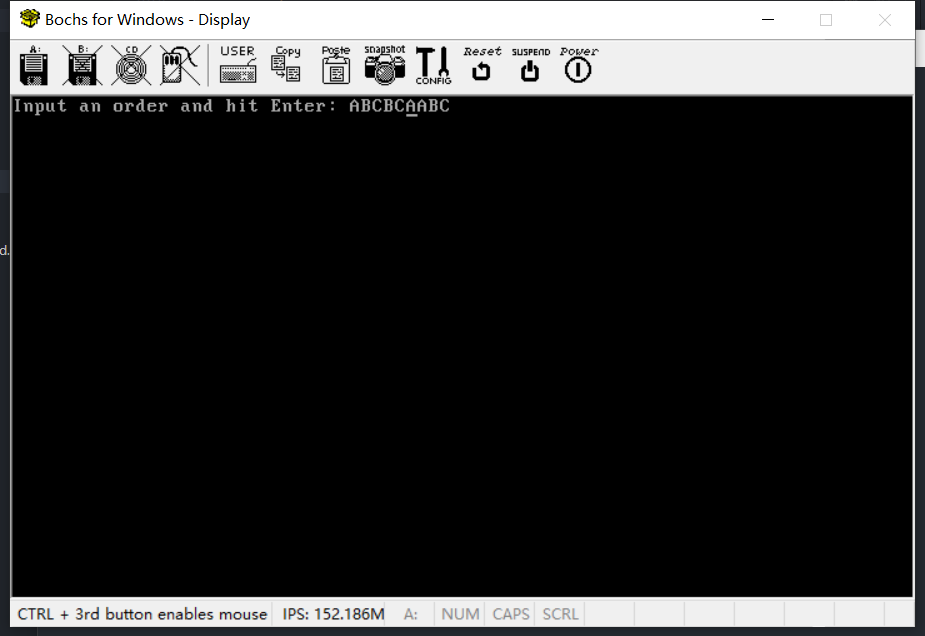
回车：执行字符串对应的用户程序序列

指定字母（ABCD）：显示，并作为序列

有效其他字符无效

执行完毕后，会记忆上次执行的序列

效果：



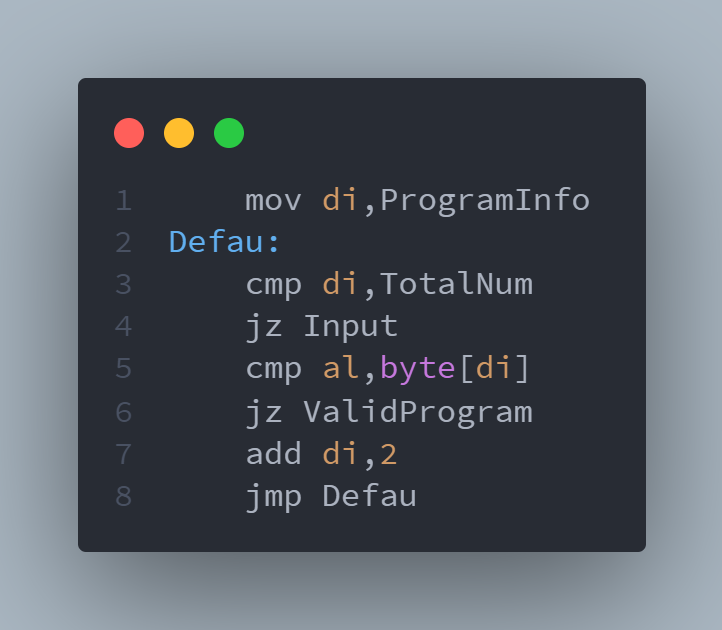
反思：

其中字符串用连续存储，因为要频繁插入删除，链接存储可能会更高效

4. a. 构建用户程序和位置的映射表



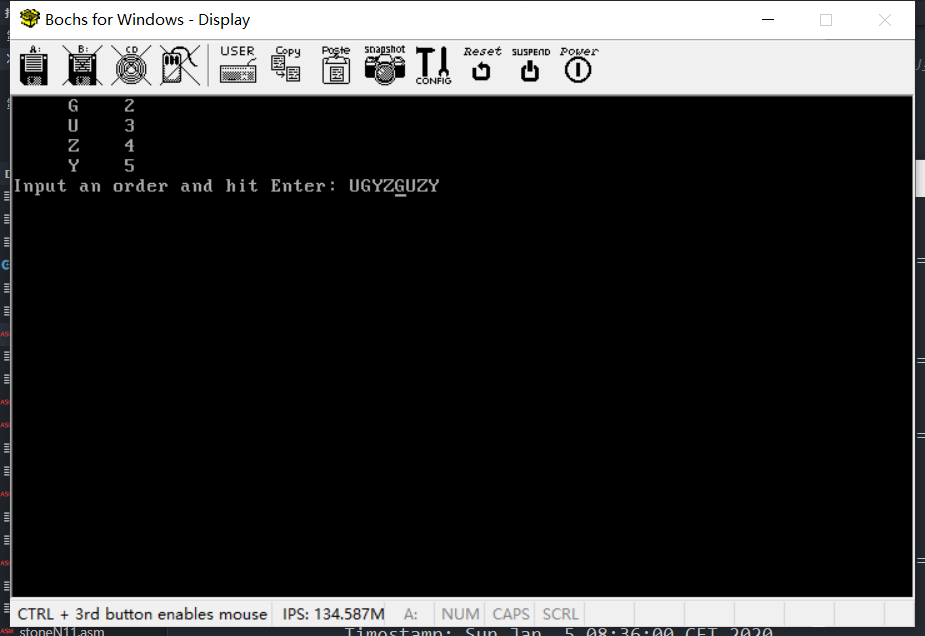
b. 更改判断输入是否有效（如是否是ABCD左右上下退格）



c. 更改映射算法为遍历整个表



c. 并在开头显示表，



**五、实验感想**

深刻理解了引导扇区程序实现用户程序加载COM，且更加深入地了解了nasm的部分语法（如BIOS中断，jmp，call）和bochs的使用，且通过nasm一定程度上实现了简单的IO操作。

**附录（流程图，注释过的代码）：**

代码见

<https://github.com/guzhegnyang/operating_system_experiment.git>