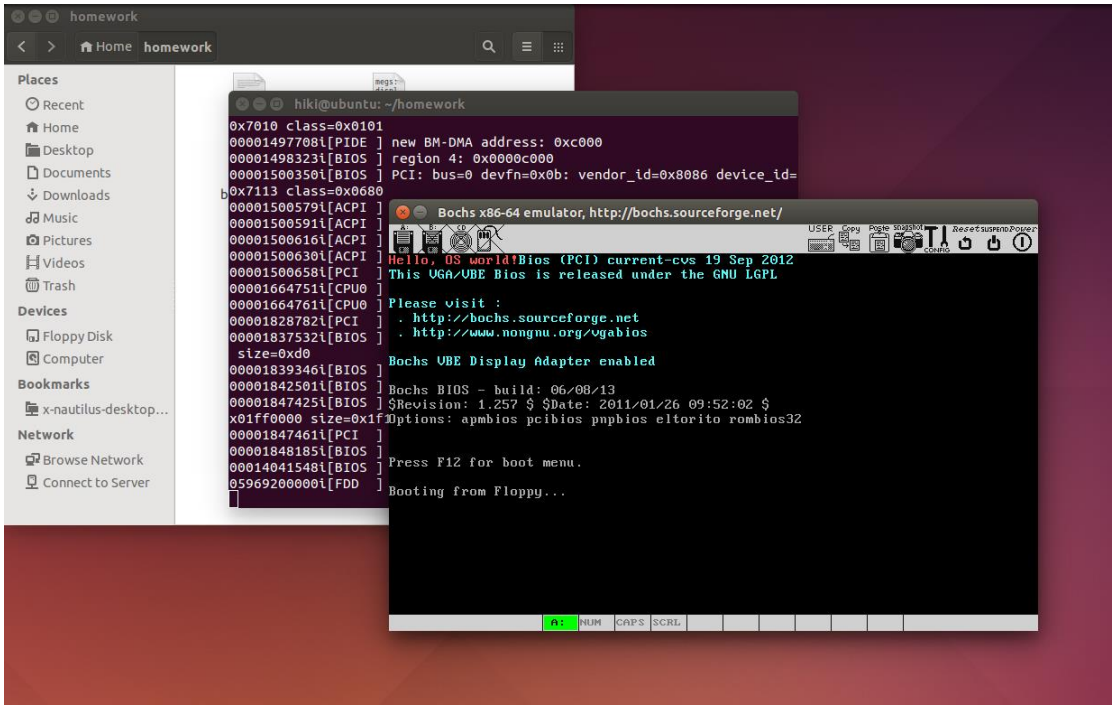
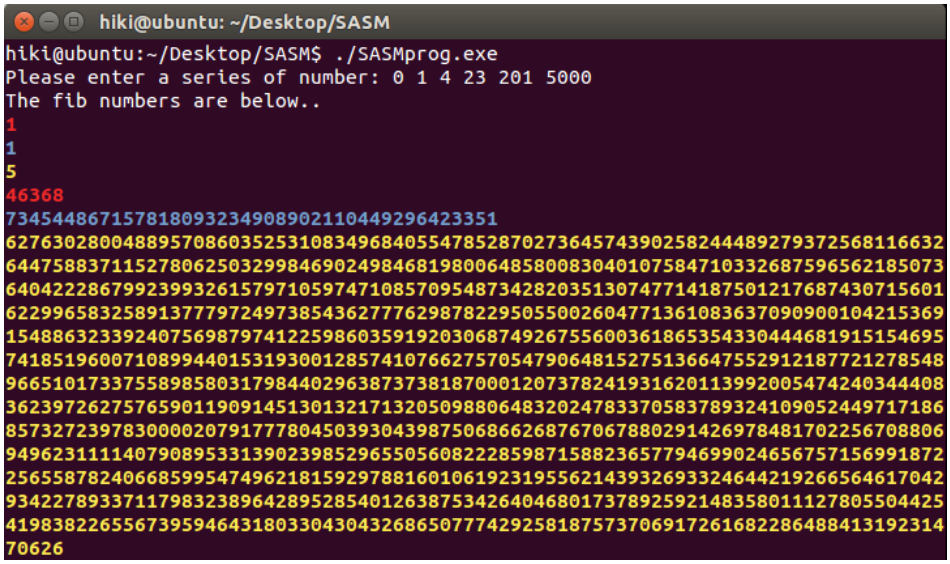
第一部分：Boot

Boot 运行截图：



第二部分：Fibonacci

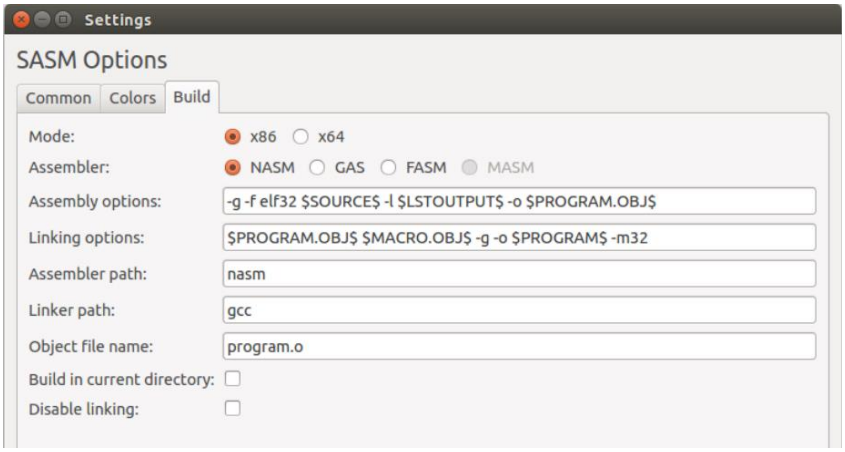
Fibonacci 运行截图



汇编环境说明

汇编系统环境：Ubuntu 14.04 32 位
IDE：SASM
IDE 说明：汇编代码是用一个 Ubuntu 下一个 IDE：SASM 写的，汇编也是通过它，所以汇编过程及其产物会与传统的方法不太一样，但都是差不多的
通过 IDE 汇编的原因：先前都是用 PPT 的方法两行代码汇编运行，但是发现在顺序读内存块的时候会出现一个 bug，就是读着读着 ecx 为 0 了，原先以为是自己逻辑的原因，搞了几个小时无果，后来用 IDE 构建了一下，发现输出是正常的，后来换了环境在命令行下还是一样，索性就用 IDE 的汇编命令来进行构建了。

IDE 汇编命令格式：



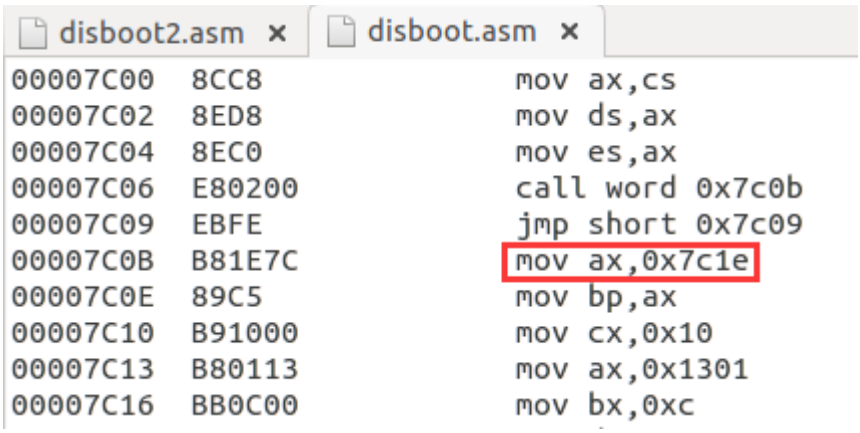
第三部分：代码阅读

Write string (EGA+, meaning PC AT minimum)	AH=13h	AL = Write mode, BH = Page Number, BL = Color, CX = String length, DH = Row, DL = Column, ES:BP = Offset of string

当 AH=13h 时，int 10h 中断表示写操作，mov bp, ax 指令将 ax，即 string 的相对地址（offset of string）传到 bp，int 10h 即可取到 BootMessage。
ax 里面的值表示 BootMessage 所在内存中的绝对地址。

相关实验：
对 boot.bin 进行反汇编，可以看到反汇编的相应结果为 mov ax, 0x7c1e，可知 ax 为绝对地址

Boot 反汇编代码图：


disboot2.asm
disboot.asm
00007C00 8CC8 mov ax,cs
00007C02 8ED8 mov ds,ax
00007C04 8EC0 mov es,ax
00007C06 E80200 call word 0x7c0b
00007C09 EBFE jmp short 0x7c09
00007C0B B81E7C mov ax,0x7c1e
00007C0E 89C5 mov bp,ax
00007C10 B91000 mov cx,0x10
00007C13 B80113 mov ax,0x1301
00007C16 BB0C00 mov bx,0xc