**操作系统实验一**

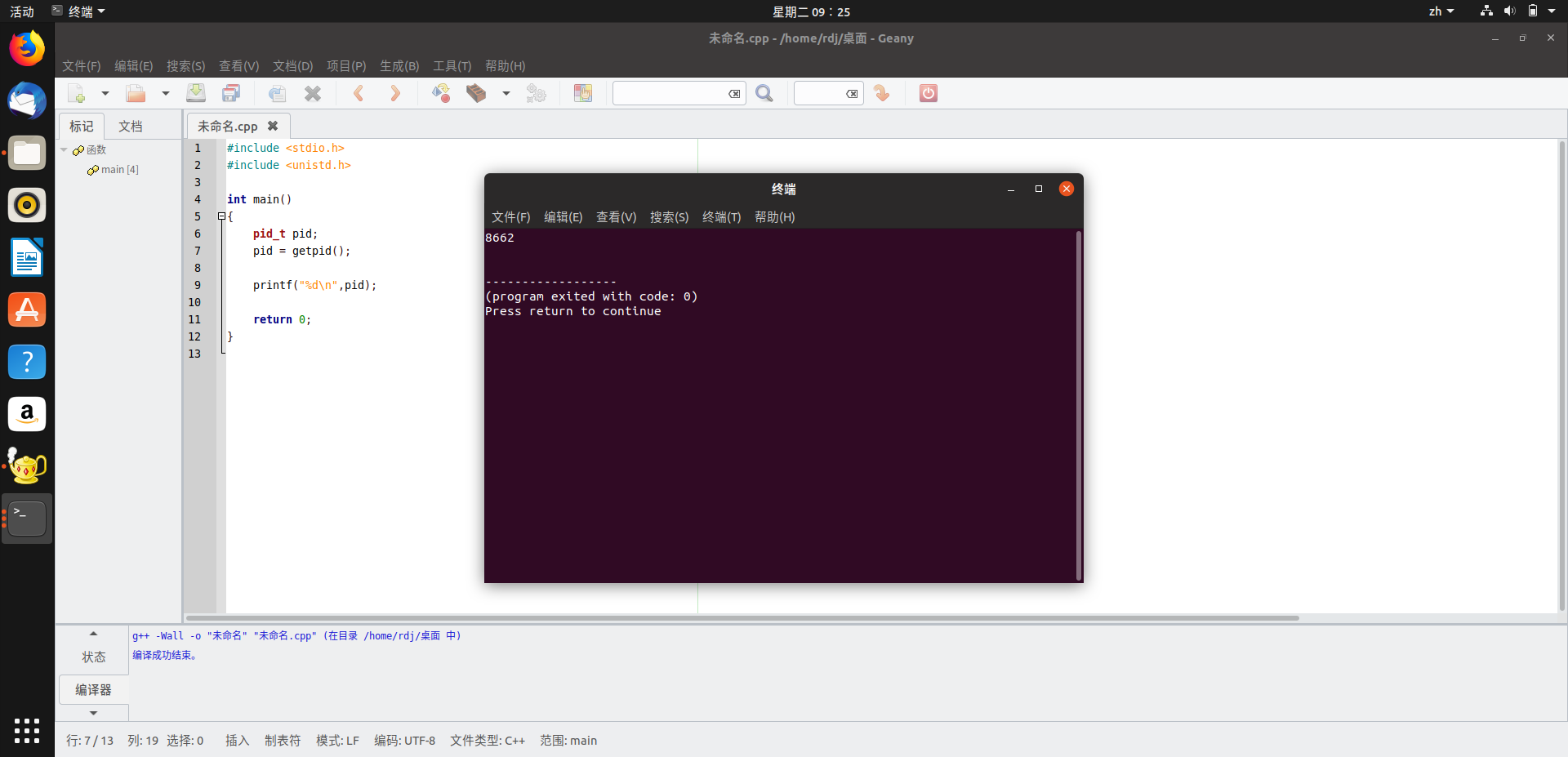
**16281053 杨瑷彤 计科1601**

**一、系统调用实验**

（1）

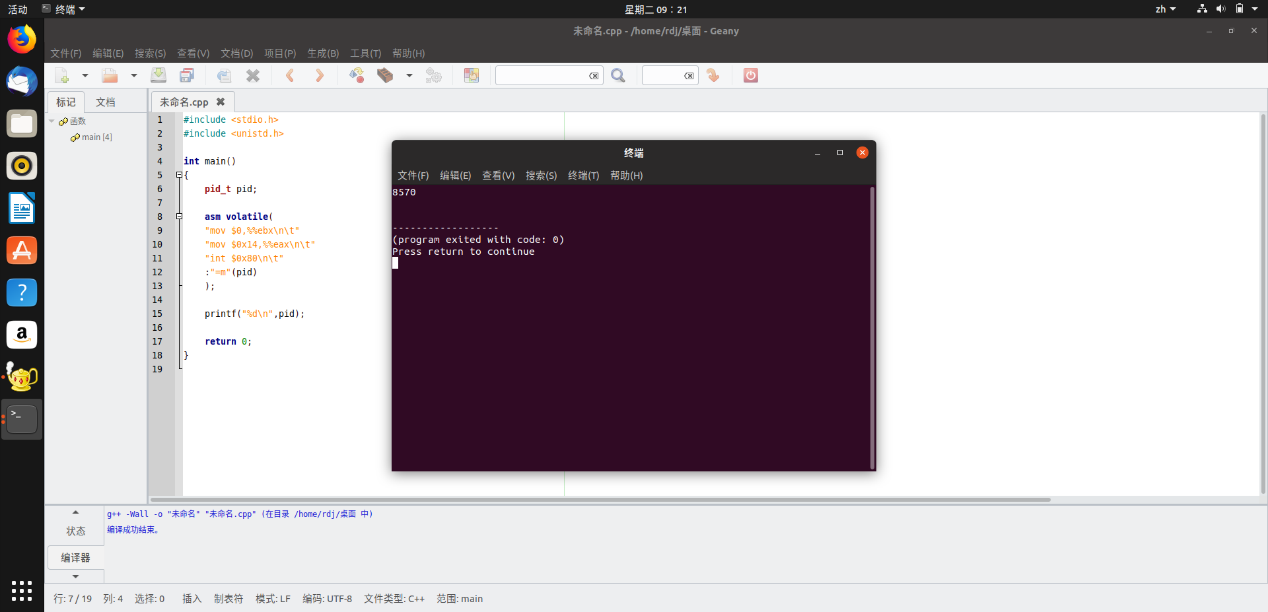
直接调用getpid（）函数：

使用直接调用getpid（）函数，取得进程识别码，结果为8662。代码见如下截图。



汇编中断方式调用getpid（）函数：

使用汇编中断方式调用getpid（）函数，首先存放一个标志到ebx中，然后eax中存放系统调用函数的调用号，根据传递进来的系统中断号，然后找到getpid()函数调用。结果为8570，代码见如下截图。

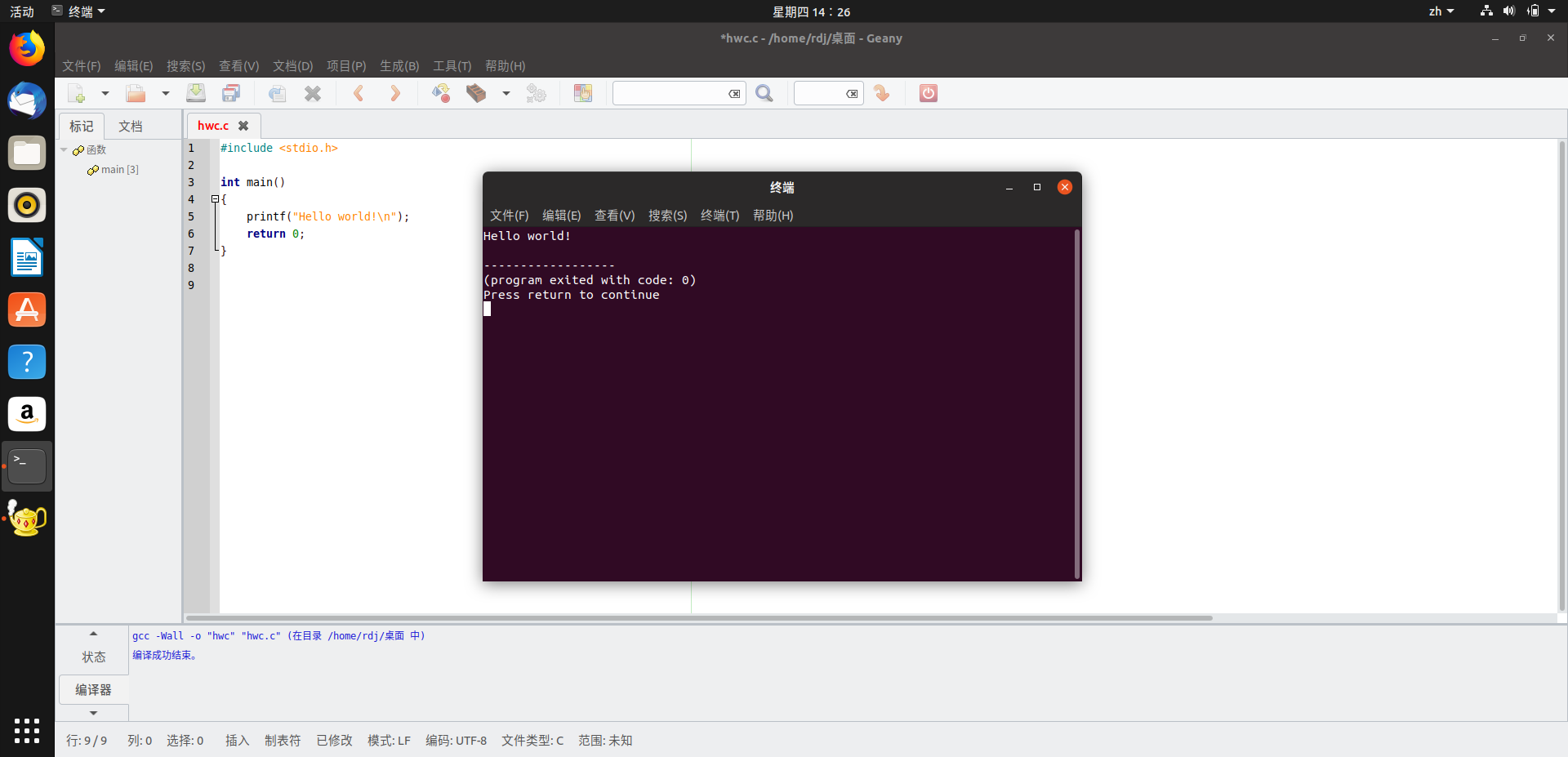


Getpid的系统调用号为39，linux系统调用的中断向量号是80.

（2）习题1.13

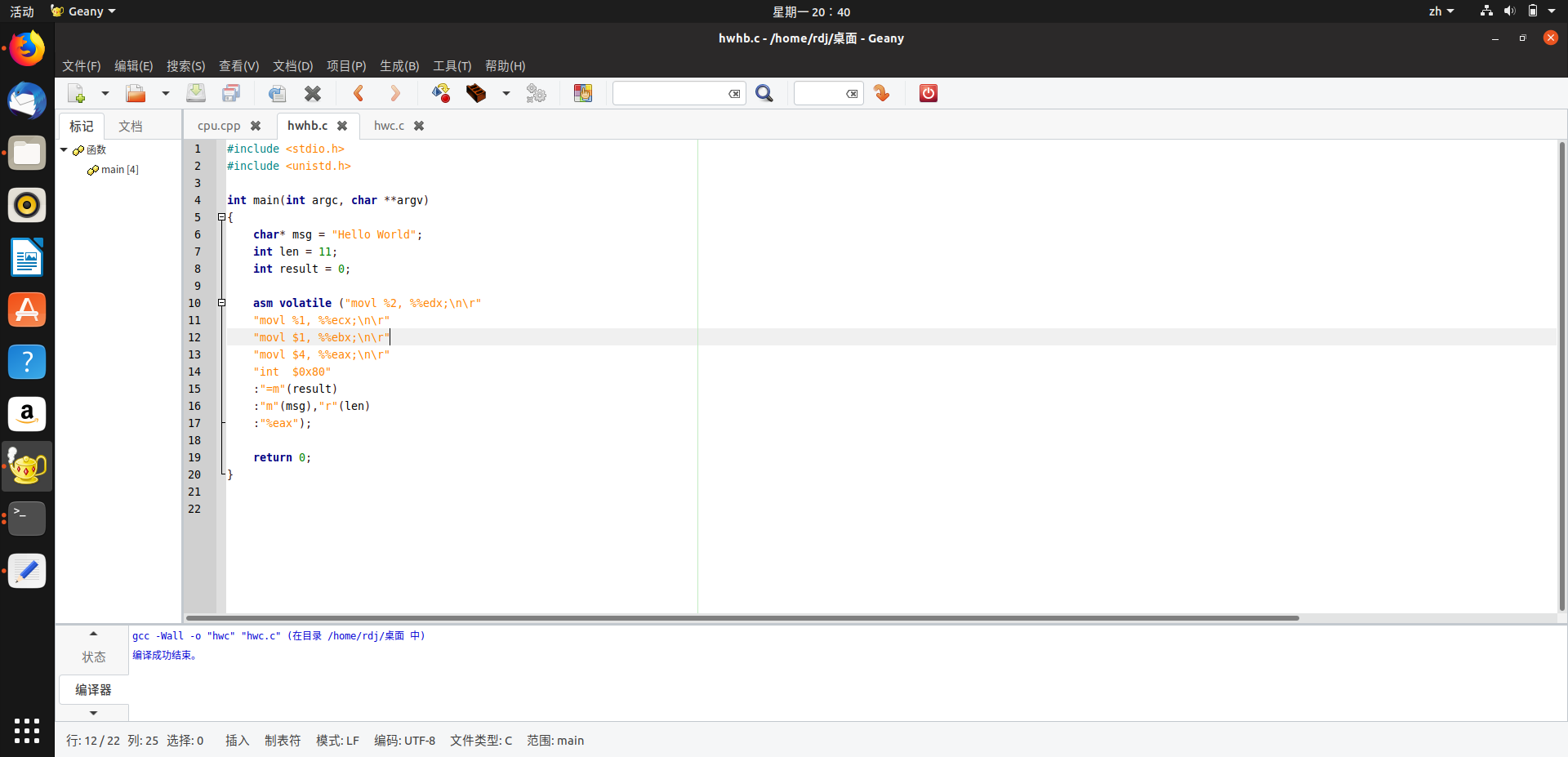
使用C函数形式输出hello world：

直接使用printf输出即可，代码见如下截图。

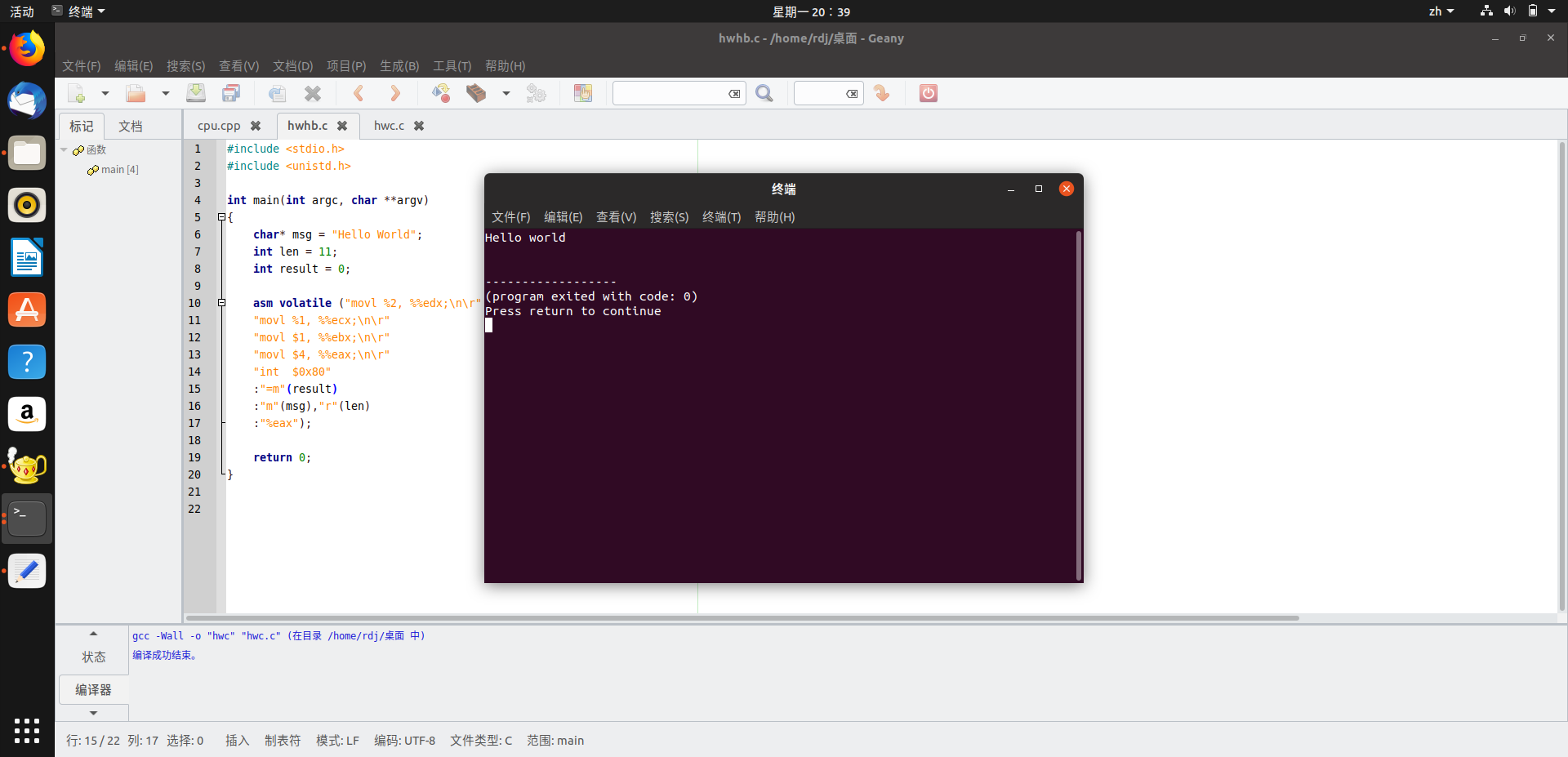


使用汇编形式编写代码输出hello world：

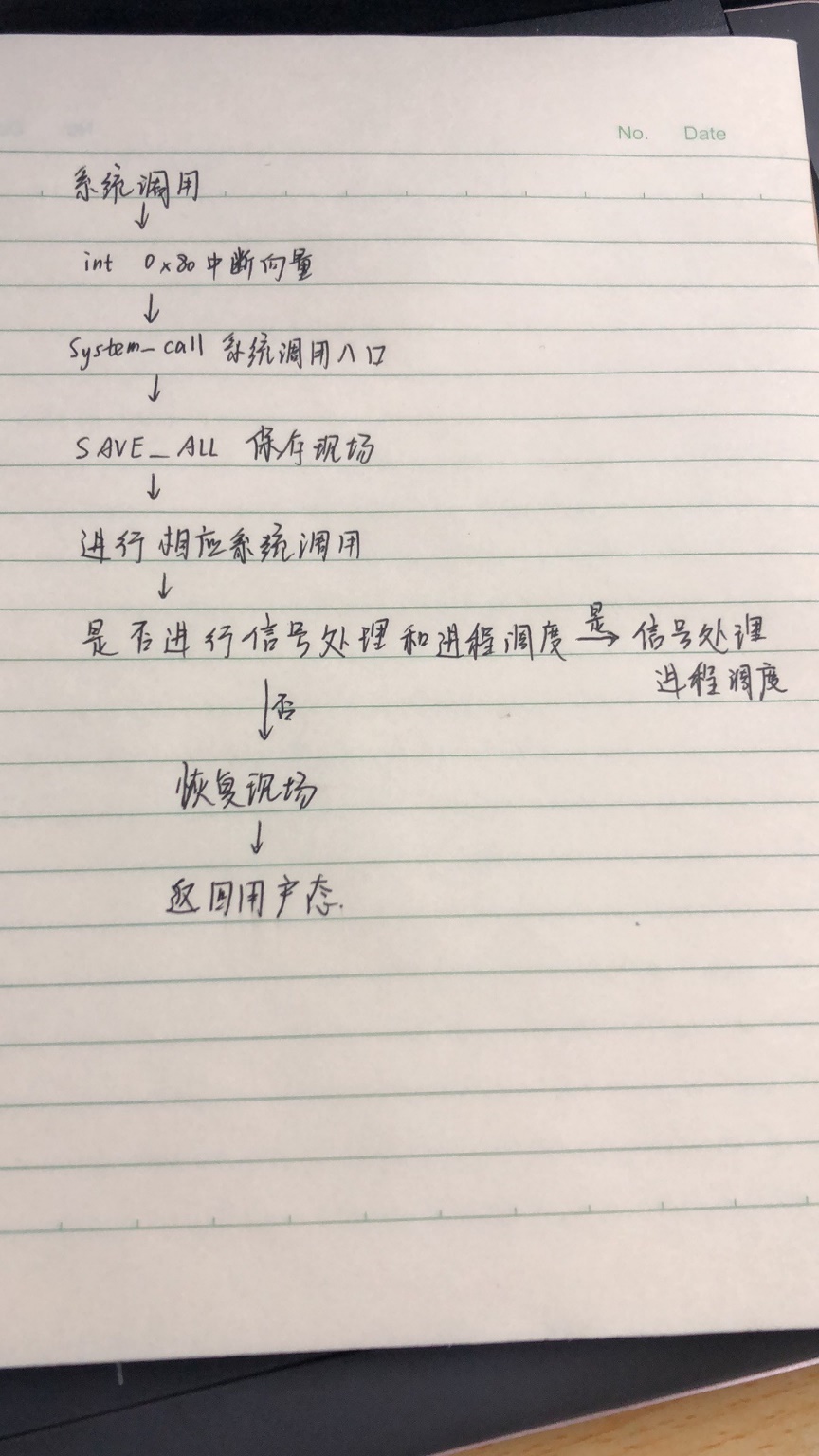
代码如下图：



直接编译运行：

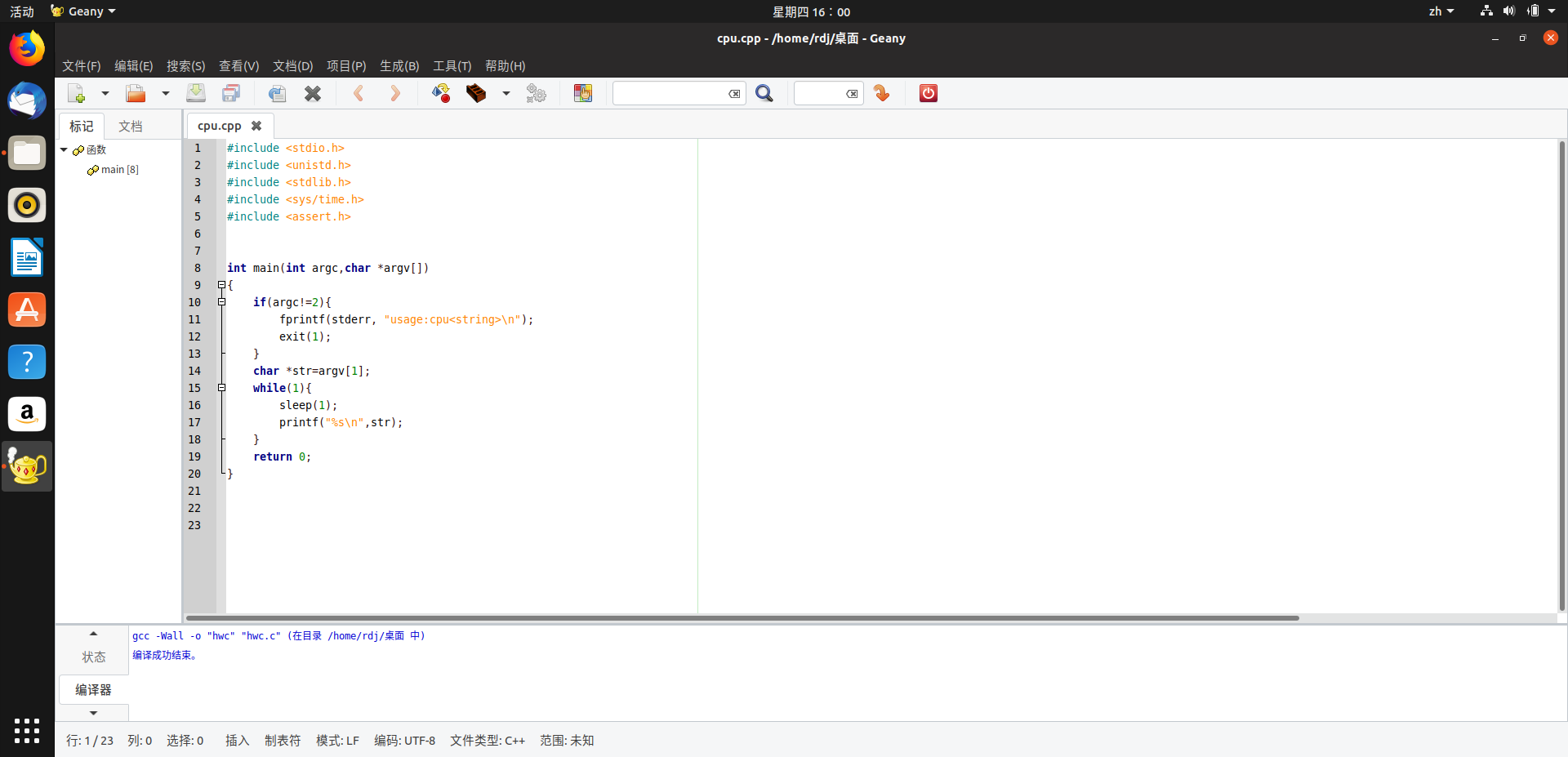


（3）系统调用实现的流程图

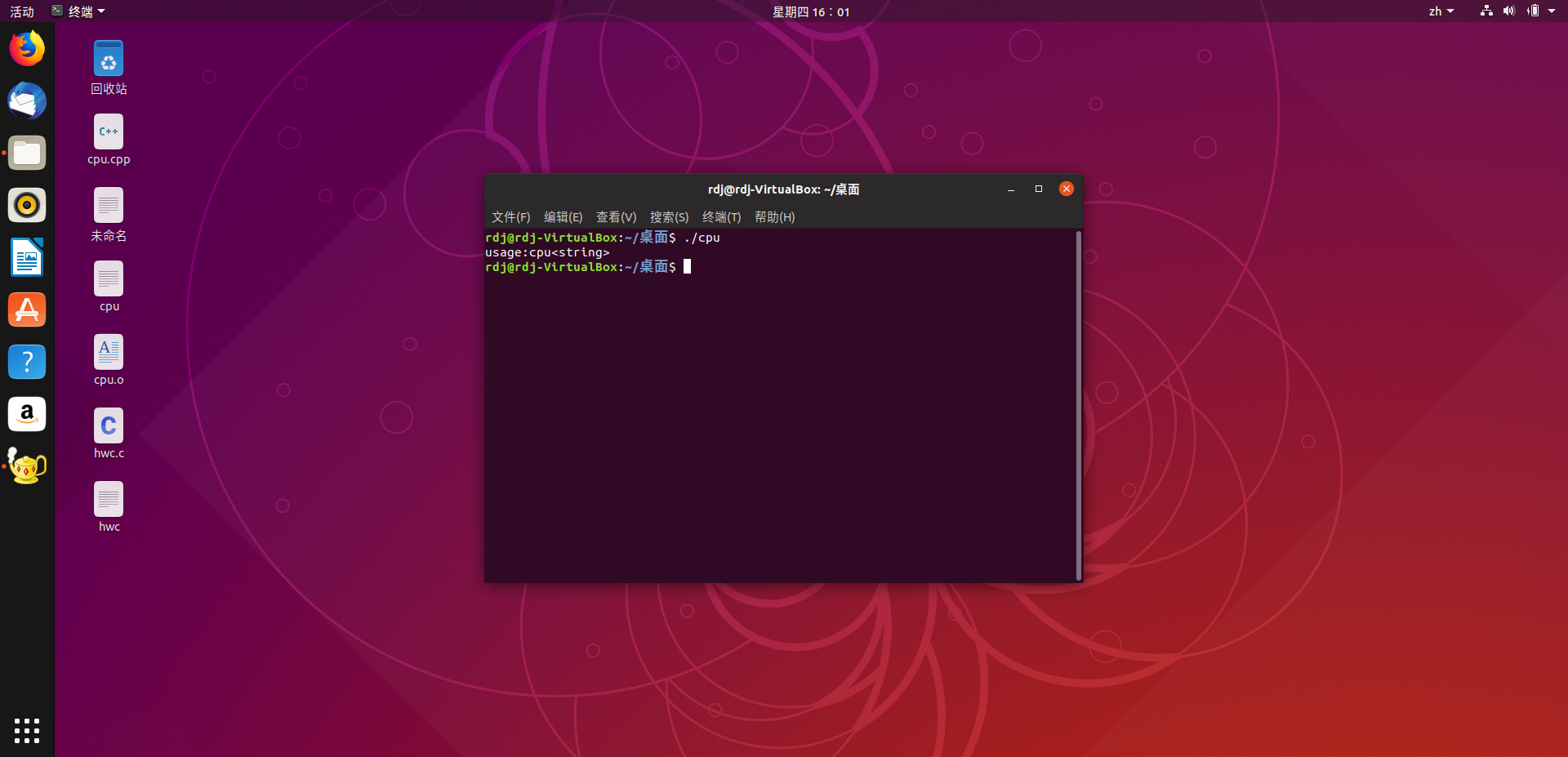


**二、并发实验**

实验代码如下：

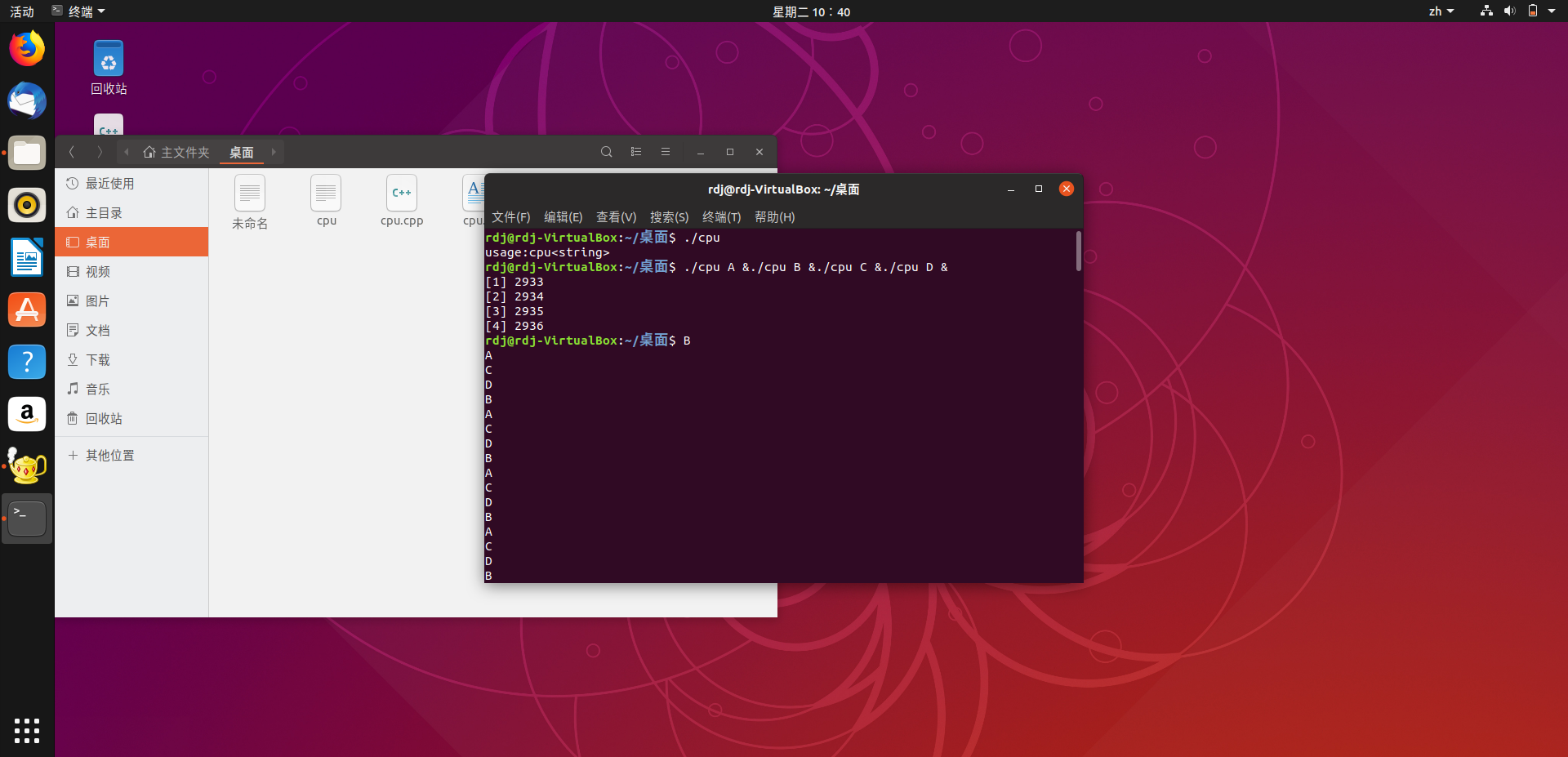


编译并在终端运行：



这个程序的功能是虚拟化cpu，将单个cpu虚拟为多个，使四个程序同时进行。当你输入不为空时，每隔一秒循环输出你所输入的字符。

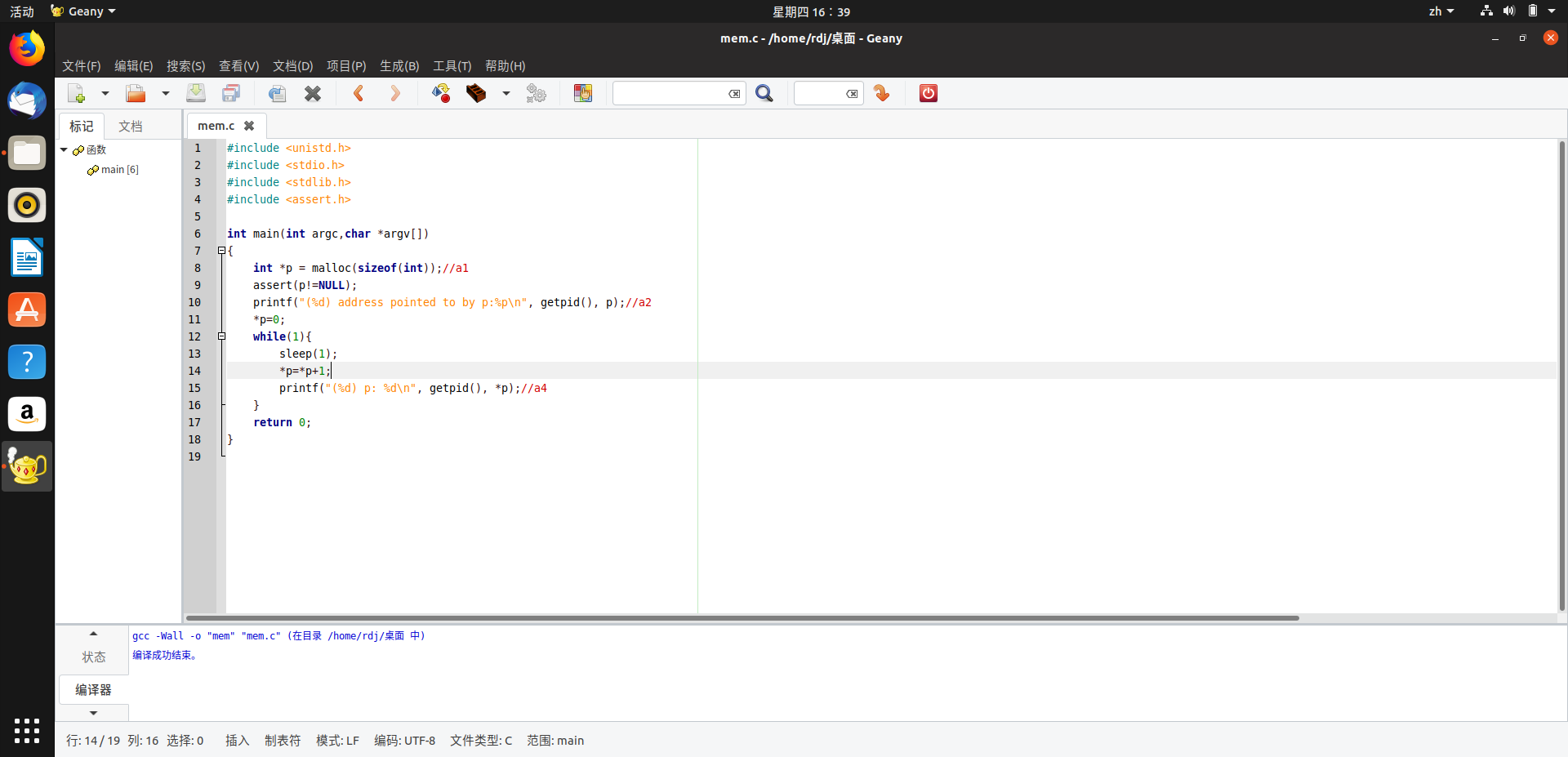
输入命令./cpu A & ./cpu B & ./cpu C & ./cpu D &，运行结果如下图：



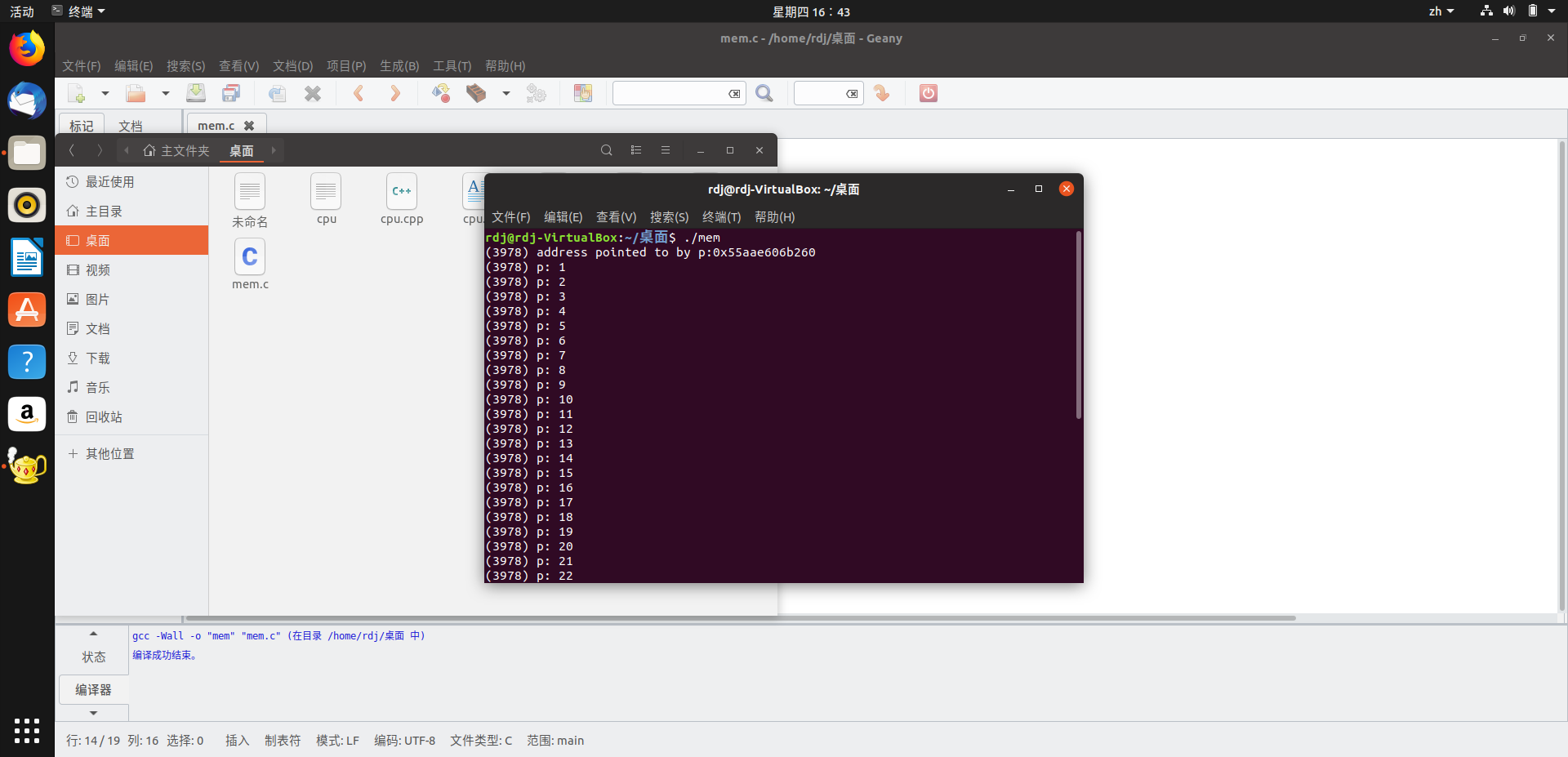
程序cpu运行了4次，分别是输入A、B、C、D这四个字母所执行的这四次。顺序是BACD，执行顺序是随机的、无规律的。程序并发执行是一个程序段的执行尚未结束，另一个程序段的执行已经开始。这个程序的功能是虚拟化cpu，将单个cpu虚拟为多个，使四个程序同时进行。当你输入不为空时，每隔一秒循环输出你所输入的字符。当输入ABCD这四个程序并发执行时，前一个程序还未执行完，即时间间隔中下一个程序开始执行，四个程序以此类推交叉执行，于是便出现了BACD四个字母交替出现的结果。

三、内存分配实验

实验代码如下：

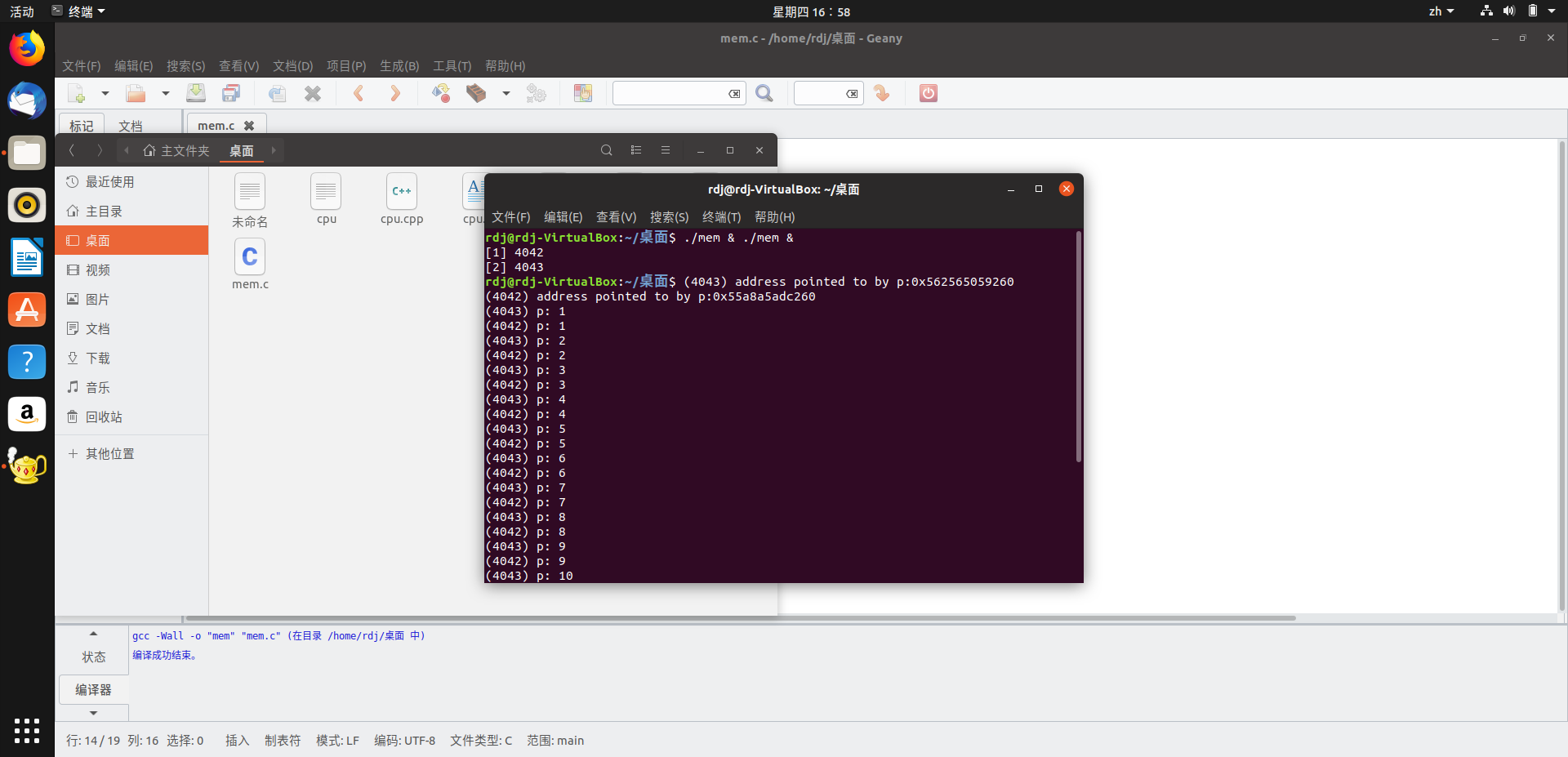


编译并在终端运行：



首先给p分配内存空间，第一行输出为：前面括号内的是进程识别码，先分配内存，然后输出指针p的首地址。令p指针内容为0，然后每隔一秒令p指针内容加一，前面括号内的仍为进程识别码，冒号后输出p指针的内容，即1、2、3……

输入命令./mem & ./mem &，运行结果如下：

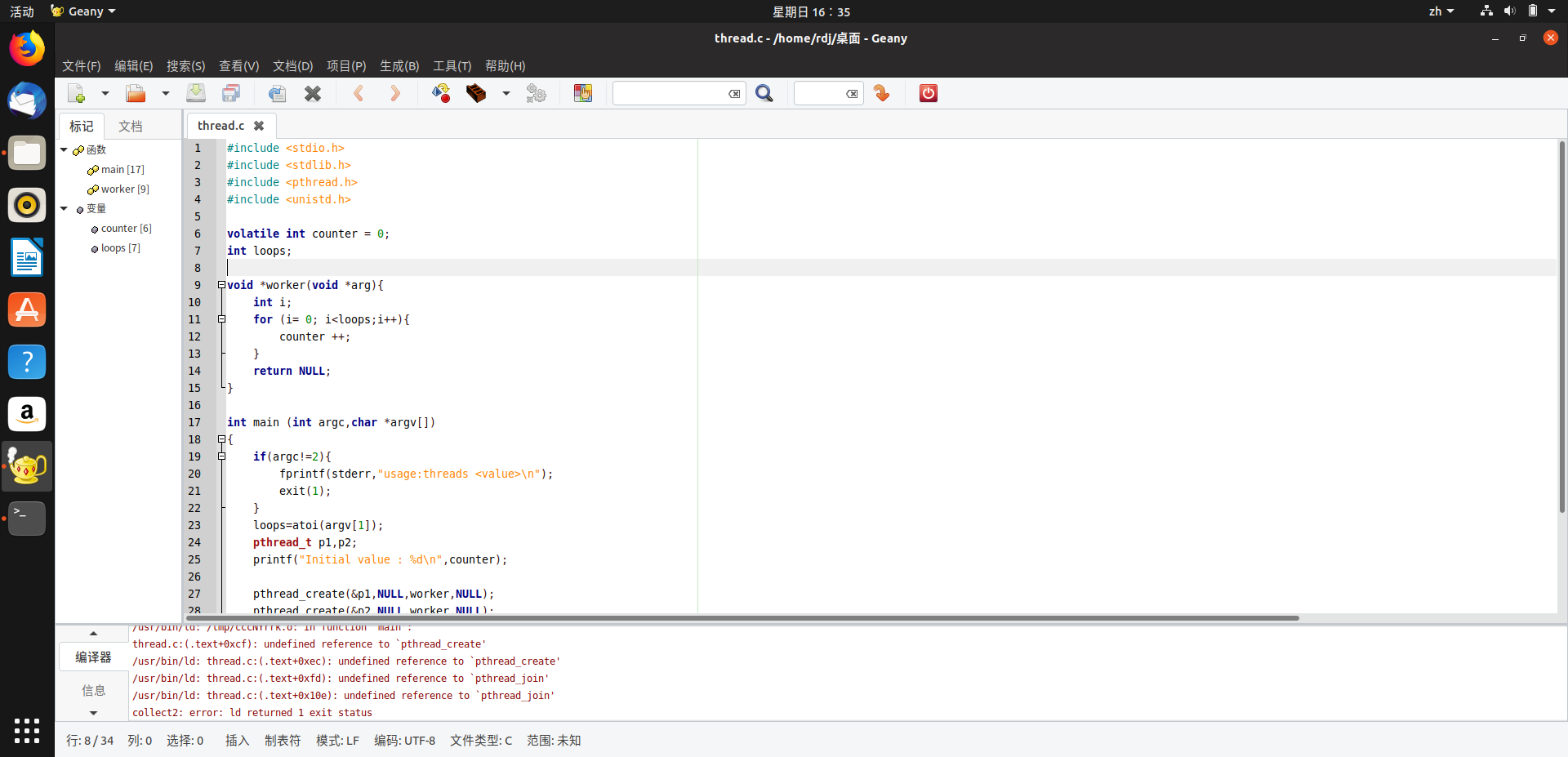


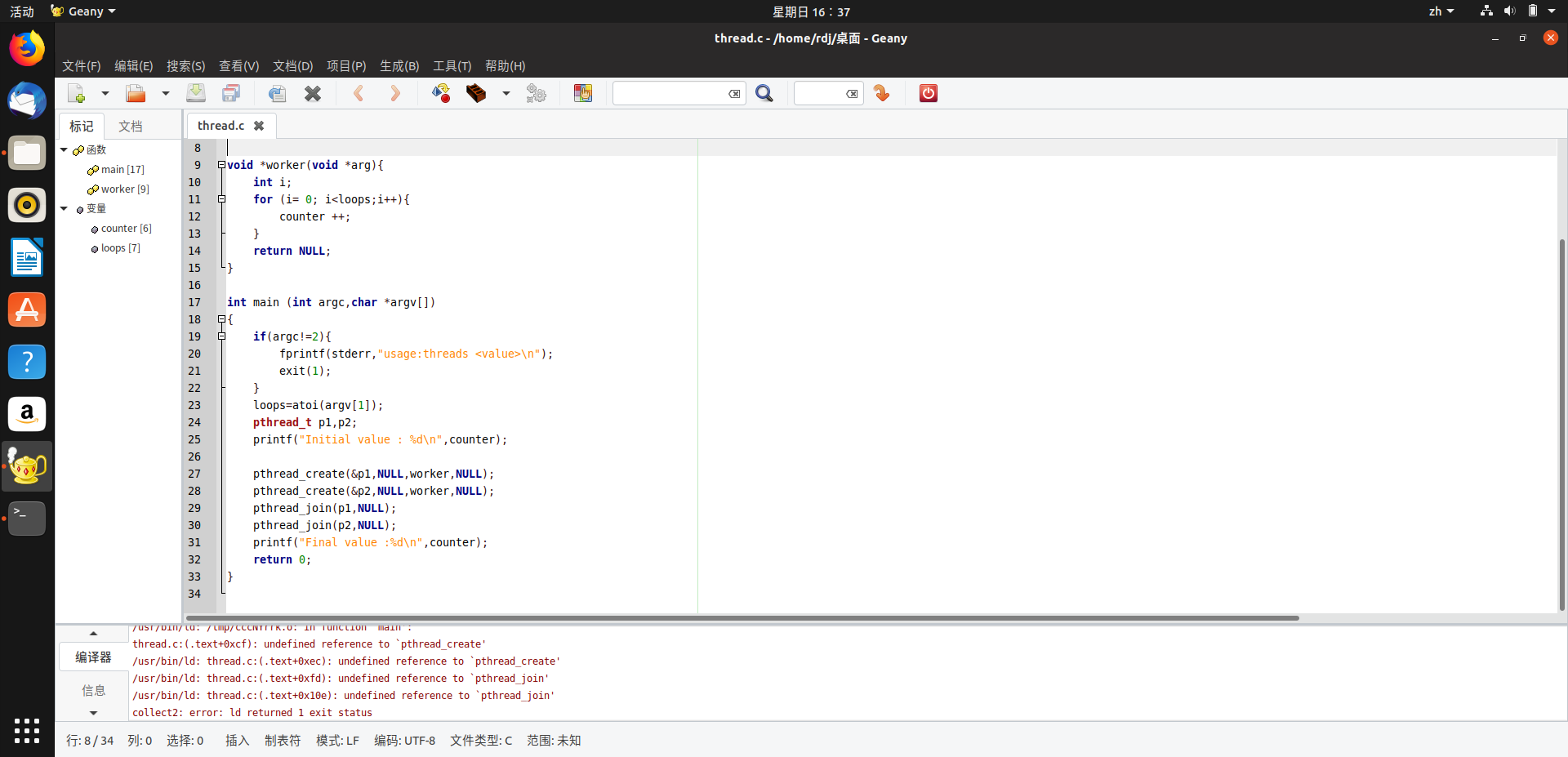
此时为两个mem程序并发执行，每个程序的进程识别码分别显示在前面的括号中，每秒钟显示两个程序各自的p的内容。

两个程序分配的内存地址不相同，不共享同一块物理内存区域。从第一行显示的首地址可以看出两个指针的首地址不同。

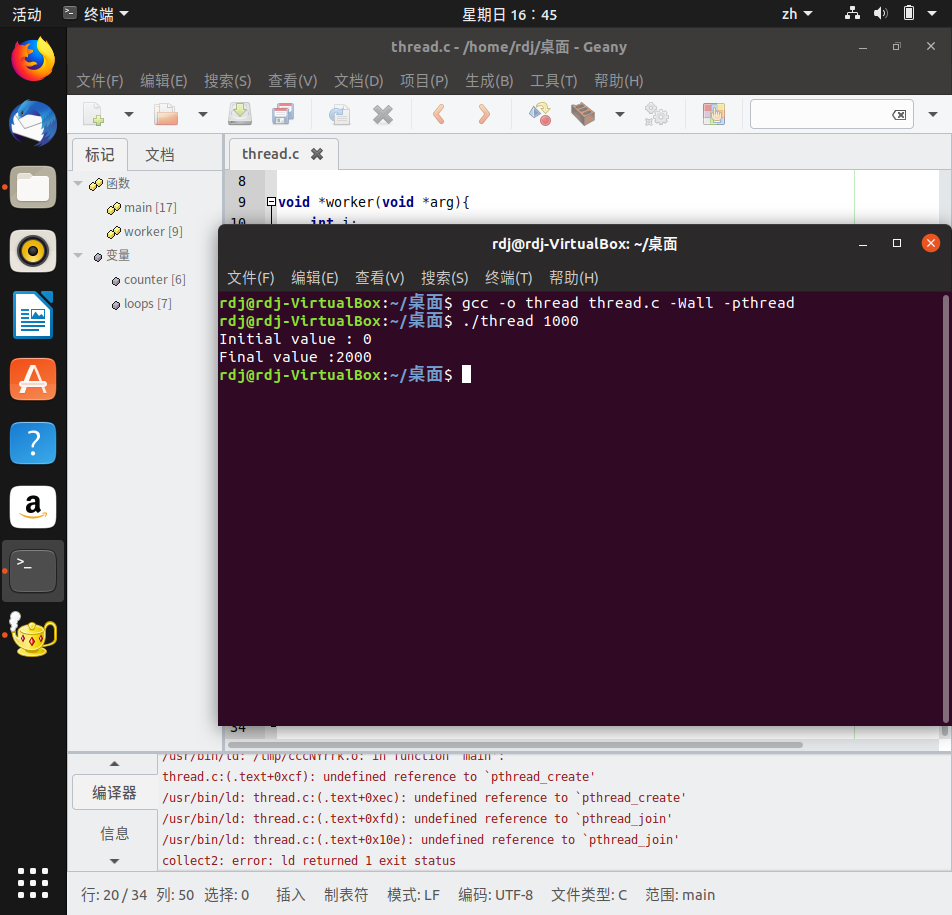
四、共享的问题

实验代码如下：

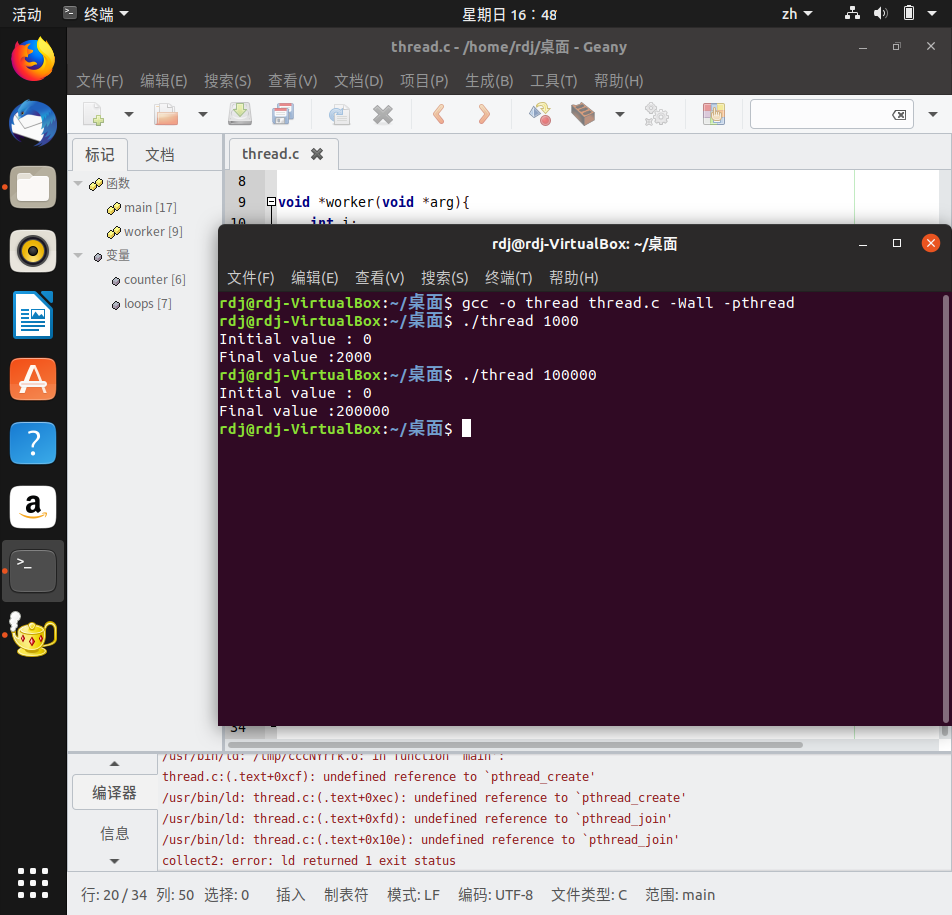




在终端编译并执行命令1：



执行命令2：



此程序是多线程程序，主程序使用pthread.create()创建两个线程，每个线程在一个名为worker（）的历程中进行，计数器统计循环次数循环递增。

执行结果与输入的参数成同一数量级

变量p1、p2是各个线程共享的