**实验报告**

**Lab 0**

姓名：周训哲

班级：302教室（张凯）

学号：20307110315

**实验报告填写要求**

1.请在每个exercise之后简要叙述实验原理，详细描述实验过程。可以使用中文进行描述，不对语言做要求。

2.请将你认为的关键步骤附上必要的截图。

3.有需要写代码的实验，必须配有代码、注释以及对代码功能的说明。

4.你还可以列举包括但不局限于以下方面:实验过程中碰到的问题你是如何解决的、实验之后你还留有哪些疑问和感想。

5.如果实验附有练习，请在每个练习之后作答，这是实验报告评分的重要部分。

6.Challenge为加分选作题。每个lab可能有多个challenge,我们会根据完成情况以及难度适当加分，具体情况会在课上说明。这部分的实验过程描述应该比exercise更加详细。（请注意，Lab0为基础环境配置，不设置挑战问题。）

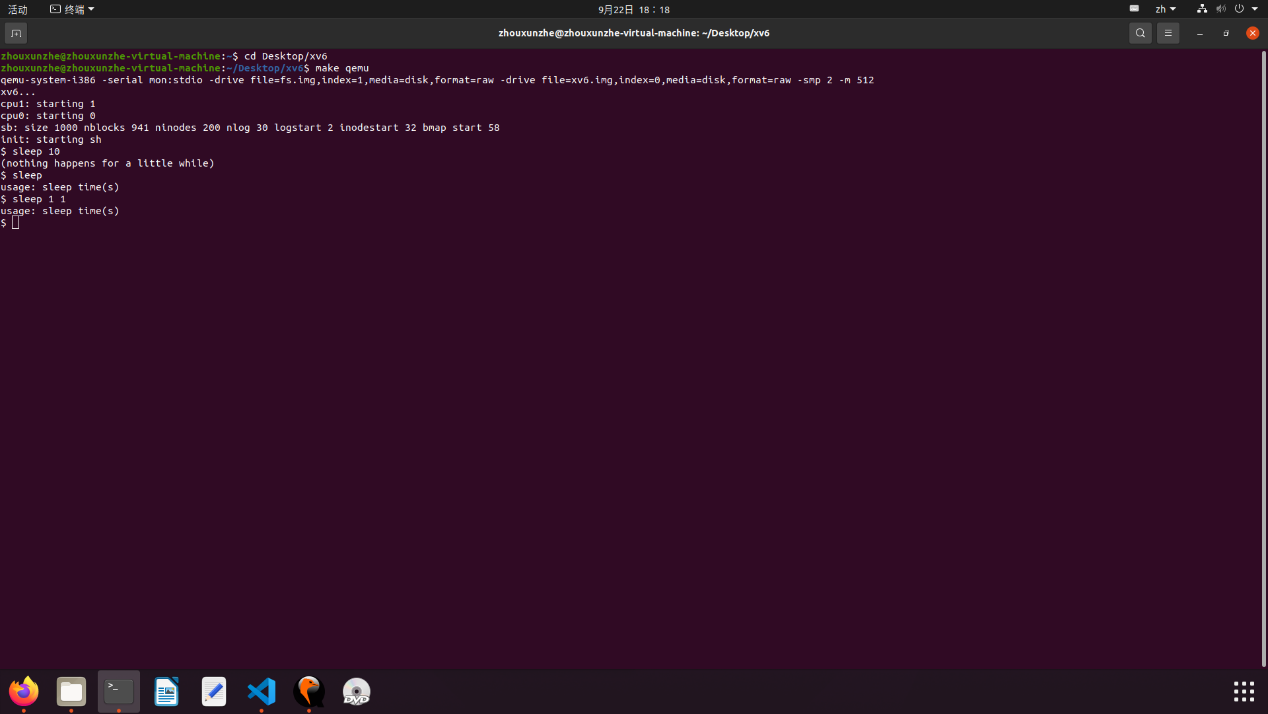
7.切勿抄袭亦或是去互联网复制粘贴答案。

【练习题模板】

1. Question
2. Screenshot
3. Difficulties and solutions

**实验练习1**

Exercise：Implement the UNIX program sleep for xv6; your sleep should pause for a user-specified number of ticks. A tick is a notion of time defined by the xv6 kernel, namely the time between two interrupts from the timer chip. Your solution should be in the file user/sleep.c.



实验原理及实现方式：利用sleep（）函数设定睡眠时间。在函数中还增加判断输入格式是否正确的判断，如果正确则输出"(nothing happens for a little while)\n")，否则输出"usage: sleep time(s)\n"。其中，由于用户输入的格式为字符串，如果需要调用sleep函数，则需先用atoi（）函数将数字的字符串类型转换为整型。将程序放到makefile中的UPROGS，make qemu会编译整合程序，用户便可在xv6 shell中就可以调用sleep指令了。

Sleep代码：

#include"types.h"

#include"user.h"

Int

// 模仿echo指令,其中argc表示输入参数的数量

// argv存储输入的参数

main(int argc, char \*argv[])

{

if(argc != 2)

{

printf(1, "usage: sleep time(s)\n");//判断输入是否合理

exit();

}

// 利用atoi() 将输入的数字字符串转换为整型

int time = atoi(argv[1]);

printf(1, "(nothing happens for a little while)\n");

// 利用sleep() 调用kernal的休眠

sleep(time);

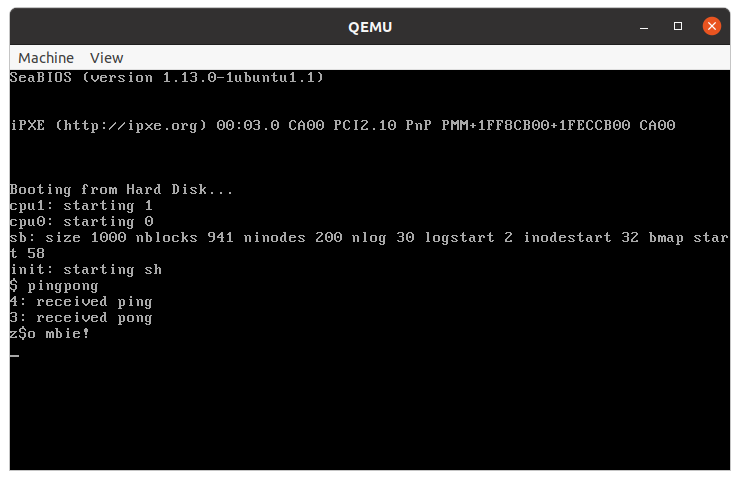
// 最后使用exit() 退出程序

exit();

}

**实验练习2**

Exercise：Write a program that uses UNIX system calls to ''ping-pong'' a byte between two processes over a pair of pipes, one for each direction. The parent should send a byte to the child; the child should print "<pid>: received ping", where <pid> is its process ID, write the byte on the pipe to the parent, and exit; the parent should read the byte from the child, print "<pid>: received pong", and exit. Your solution should be in the file user/pingpong.c.



实验原理及实现方式：首先定义父进程和子进程的输入输出端，然后用pipe（）管道将输入输出端进行连接。fork（）函数创建子进程，若失败则返回值小于0，退出。两个进程分别进入到不同的条件中。在父进程中，首先关闭父到子的读与子到父的写，然后在向子进程传输的管道中写入"ping"，子进程中首先关闭子到父的写进程与父到子的写，然后读取父进程传输的"ping"，并输出<pid>：received %s（ping）。然后子进程再执行写"pong"的操作作为对父进程信号的回应，在父进程中虽然与子进程是同时进行的，但是只有子进程写之后才能执行读的操作，当子进程写好"pong"之后，父进程会读取数据并输出<pid>：received %s（pong）。如此一来，便完成信息在双向管道传输中的一个来回。最后由于没有调用wait（）函数kill子进程，所以会产生僵尸进程。（读写函数分别为read（），write（），<pid>利用getpid（）获取），最后与sleep相同，将程序放入makefile中的UPROGS，便可以在make qemu后在xv6系统中直接运行。

Pingpong代码:

#include"types.h"

#include"user.h"

int

main(int argc, char\* argv[])

{

//声明文件,定义输入输出端

int par\_to\_chil[2], chil\_to\_par[2];

//生成管道

pipe(par\_to\_chil);

pipe(chil\_to\_par);

//利用buf作为缓冲,暂存read()的数据

char buf[8];

//利用fork()生成子进程

//fork()=0: child process

//fork()>0: parent process

//fork()<0: make child process failed

if(fork() == 0) //child

{

//在进行读写之前关闭不需要的端口

close(par\_to\_chil[1]);

close(chil\_to\_par[0]);

read(par\_to\_chil[0], buf, 4);

printf(1, "%d: received %s\n", getpid(), buf);

write(chil\_to\_par[1], "pong", 4);

}

else //parent

{

close(par\_to\_chil[0]);

close(chil\_to\_par[1]);

write(par\_to\_chil[1], "ping", 4);

read(chil\_to\_par[0], buf, 4);

printf(1, "%d: received %s\n", getpid(), buf);

//if delete wait(),child will turn to zombie!

//the purpose of wait() is to kill child process when finished

}

//最后退出程序

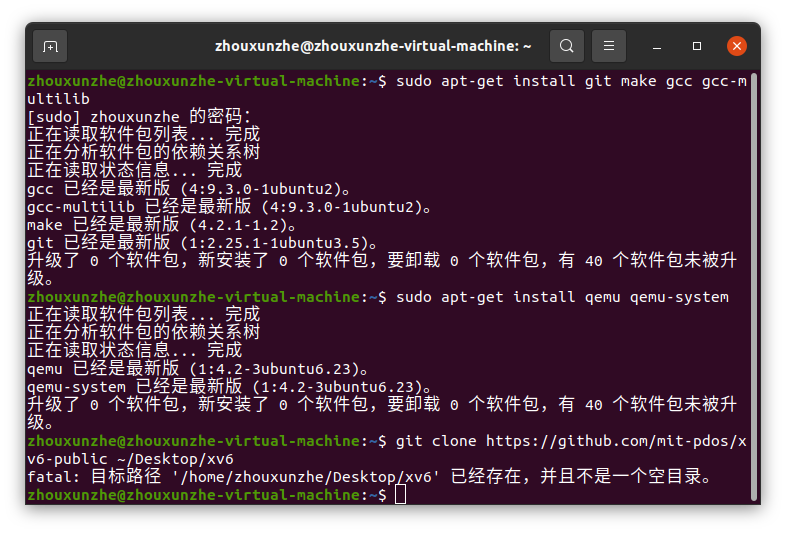
exit();

}

**实验练习3**

**Part 1：Environment Configuration**

**Exercise 1：**Please install your own environment, attach a screenshot of the process and describe it in detail.

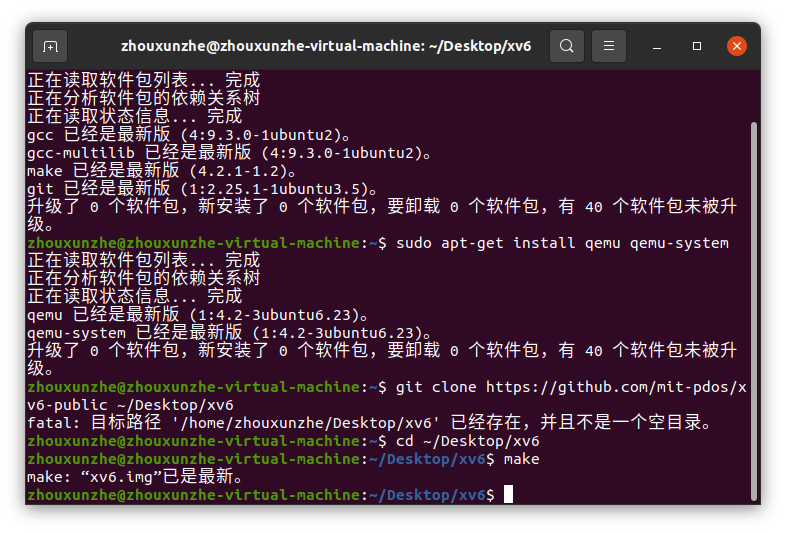


**sudo apt-get install git make gcc gcc-multilib**

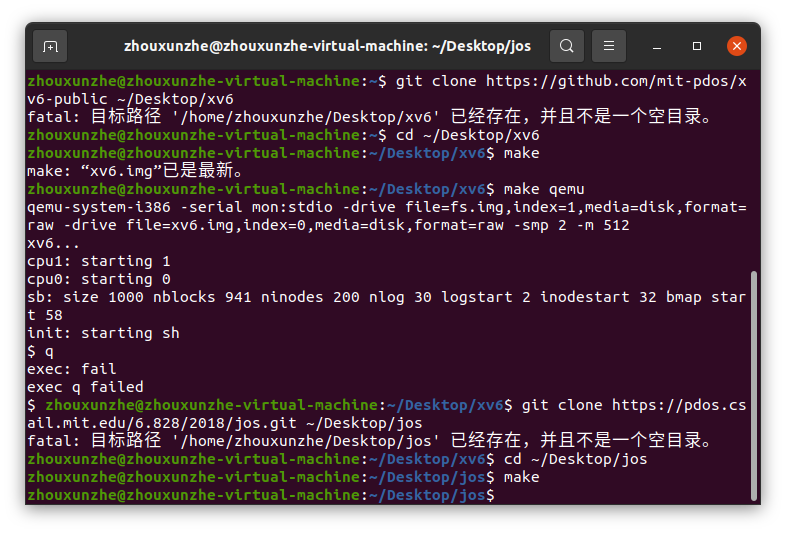
这条指令的含义是安装gcc编译环境，可以将.c文件编译成.o 、.i、可执行文件等，从而让代码正常运行。

**sudo apt-get install qemu qemu-system**

这条指令的含义是安装qemu系统，qemu是一个虚拟机系统，可以在qemu中模拟操作系统执行计算机指令。

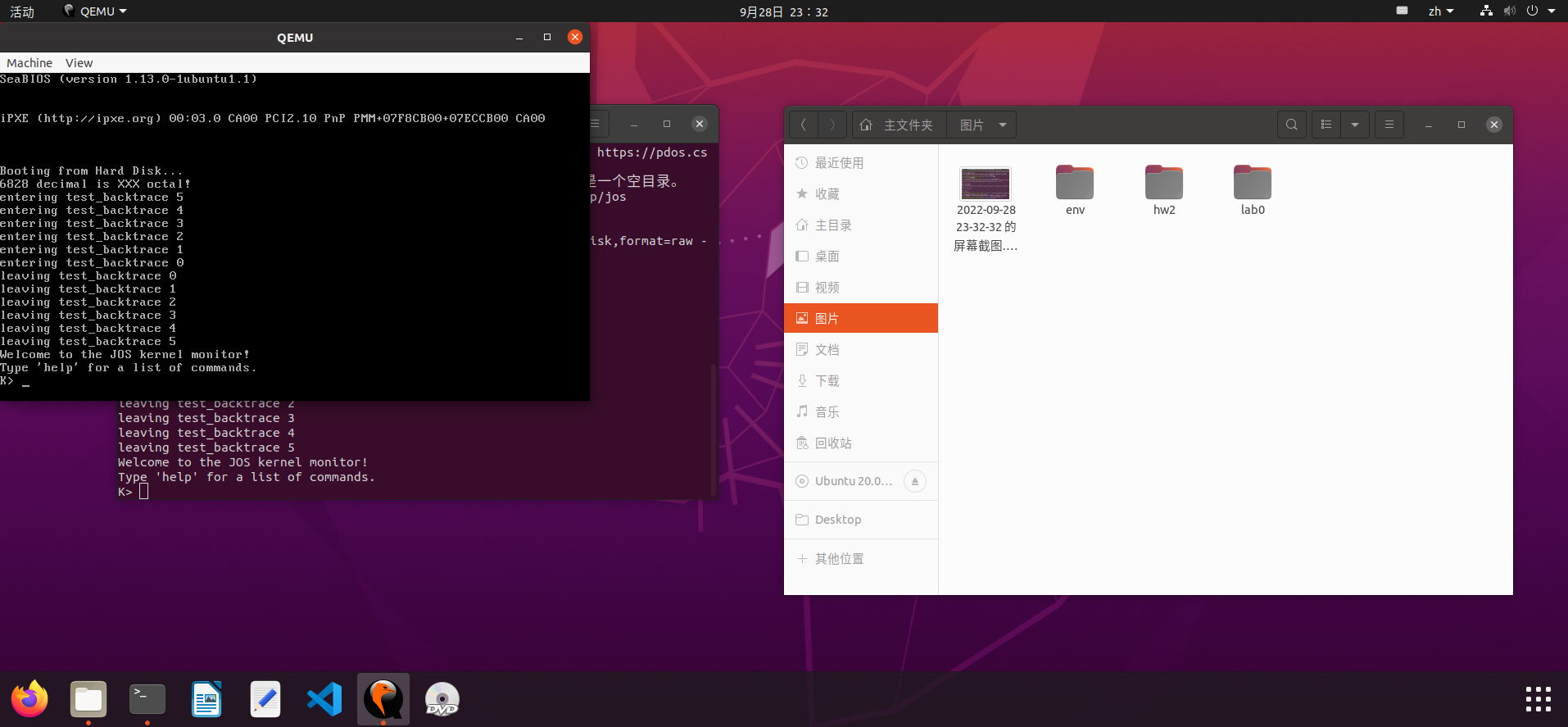
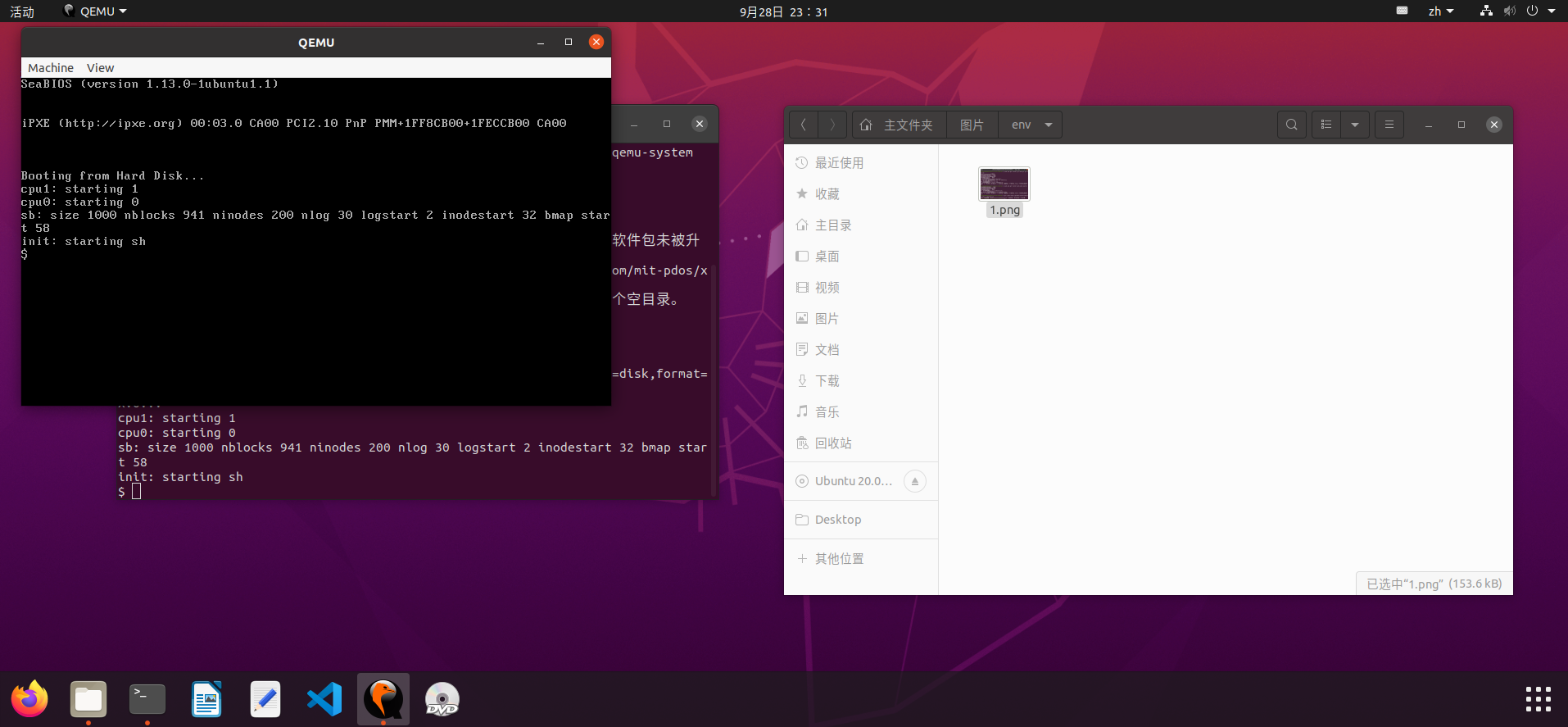


**git clone** [**https://github.com/mit-pdos/xv6-public ~/Desktop/xv6**](https://github.com/mit-pdos/xv6-public%20~/Desktop/xv6)

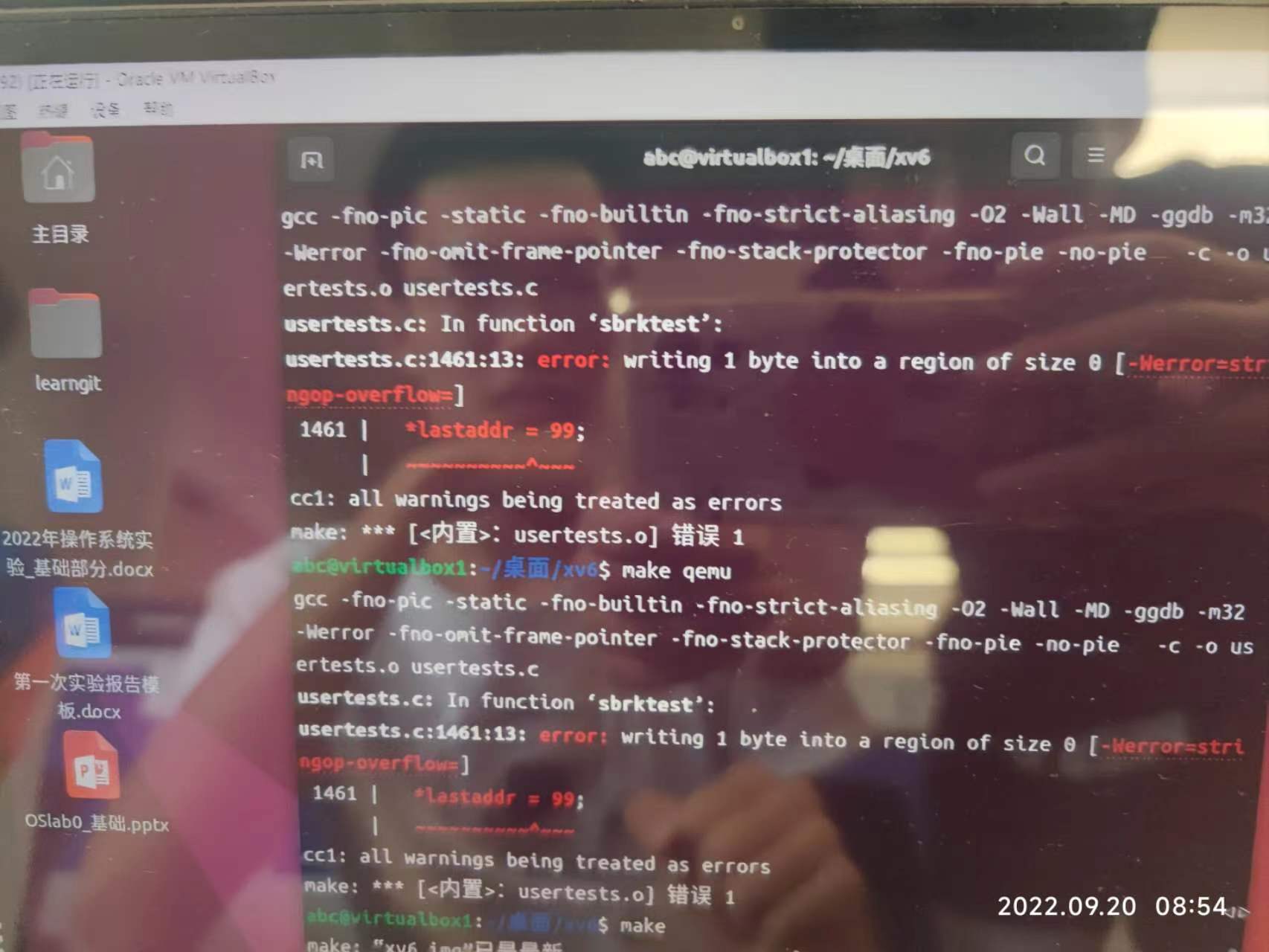


**git clone** [**https://pdos.csail.mit.edu/6.828/2018/jos.git**](https://pdos.csail.mit.edu/6.828/2018/jos.git) **~/Desktop/jos**

这两段指令的含义是将网络上的xv6系统和jos系统克隆到本地文件夹xv6和jos，然后用**make**指令对系统进行编译。**make qemu**指令表示用qemu运行系统。



**Exercise 2 ：**What problems did you encounter, how did you solve them, and if you didn't encounter any problems, what did you learn during the installation process？（此处忘记截图了，更换系统之后才反应过来，此时已经make成功）

一开始无法使用make qemu指令，每次使用报错如上图所示，经网上查阅资料、论坛，发现原因是qemu版本与Ubuntu版本不符，22版本不能运行，将Ubuntu版本换成20版本后可以正常通过qemu运行xv6或jos系统。

**Part 2：Some More Questions about Configuration**

**Exercise 3 ：**What does “operating system virtualization” mean and what important uses does it have？

操作系统虚拟化是指多个操作系统虚拟出来共用同一个电脑的硬件，而互相不干扰影响，从而允许在一台计算机系统上运行不同的操作系统的不同应用程序，并且程序之间互不影响。

用处有: 可以在一台计算机上运行不同的操作系统而互不干扰，从而可以相对隔离的使用不同的操作系统来实现不同的功能，可以很方便的搭建测试环境，而不需要另一台计算机，且运行条件与硬件环境无关。除此而外，虚拟化技术还有节省管理成本、节省硬件投资、省电等产业化管理的优点。

**Exercise 4：**What is Qemu? What are its advantages and disadvantages? What are xv6 and jos?

Qemu是一个用C语言编写的开源虚拟化软件。

Qemu的优点有：支持很多被广泛使用的架构，增加模拟速度，程式同时运行，在其他平台上运行Linux的程式，存储及还原运行状态、虚拟网卡、模拟多cpu等

Qemu的缺点有：对Windows系统的视窗操作系统支持不完善，模拟速度不如VMware等虚拟机软件，比其他模拟软件而言较难安装使用。

Xv6和jos都是MIT开发的用于教学的操作系统，可以借助qemu软件实现操作系统虚拟化。

**Exercise 5：**What's the difference between Qemu and VMware or Virtual Box?

Qemu和Virtual Box都是开源虚拟机平台。其中qemu是linux内核的虚拟机，在Windows系统中不是很完善。virtual box尤其适合Android系统，相比qemu，各方面性能都更好，2d性能强悍，但是3d功能欠佳。

VMware是闭源软件，相比virtualbox，3d功能更好，但是2d功能较差，可以配置桥接网络。

**Exercise 6：**What does the "make" directive mean? What is a makefile？

为了节省编译过程的复杂性以及节省时间，Linux系统提供了make指令，在必要时重新编译所有受改动影响的源文件。Makefile文件为make指令描述如何构建应用程序。

具体过程为：在执行make命令时，make会读取makefile文件获得一个程序的依赖关系，并按照makefile文件中给出的命令来确定目标文件的创建顺序以及正确的规则调用顺序，同时将执行的命令打印到标准输出。

Makefile由一组依赖关系和规则组成，每个依赖由一个目标文件和一个该目标文件所依赖的源文件组成；规则中描述了如何利用依赖文件创建目标文件。

**Part 3：The Ending**

**Exercise 7：**Please tell me what you learned from this experiment, or give a summary of the experiment.

在这个实验中，首先我学会了如何正确的安装虚拟机，利用Vmware以及镜像文件在虚拟机上运行Ubuntu操作系统，并且由于第一次使用linux系统，在配置操作系统的过程中，还学会了如何根据自己的要求在linux系统中安装各种所需的配件。其次，我学会了配置qemu环境以及如何安装xv6和jos系统并编译运行。学会了利用qemu运行xv6和jos系统并且使用gdb调试系统。最后在exercise中，sleep指令中学会了如何自己编写一个程序能够用shell直接调用，并且对kernal进行操作（用makefile）。Pingpong指令中了解了父进程和子进程的概念，以及怎样通过管道进行数据的传输。在回答相关操作系统的question中对OS的一些基本概念又有了更深入的了解。

本次实验作为操作系统的一个引入，让我对操作系统的整体框架以及具体指令编写有了一个基本的概念，为之后更深入的操作调用打好基础。