Part a

1. Try the following commands as a non-privileged user. What does each do? How do the files f1-f5 compare? How do they compare to /dev/urandom? Remember you can get output from /dev/urandom using cat or dd. If you use cat, make sure to pipe it to less!
   1. cp /dev/urandom f1
      1. was able to run command
   2. cp -a /dev/urandom f2
      1. wasn’t able to run command as didn’t have sufficient permissions to create the special file
         1. error was cp: cannot create special file, operation not permitted
   3. sudo cp -a /dev/urandom f3
      1. was able to run the command, had root permissions cus of sudo
   4. mknod f4 c 1 9
      1. wasn’t able to run command as didn’t have sufficient permissions
         1. error was mknod operation not permitted
   5. sudo mknod f5 c 1 9
      1. was able to run command, had root permissions cus of sudo
   6. all files that were able to be created, f1,f3,f5 all look the same once using cat f1 | less and looks the same to /dev/urandom
   7. cp copies the input files/directorys to the output file/directory
   8. cp -a flag doesn’t copy the file data just the attritbutes
   9. mknod makes a directory entry and corresponding I node for a special file
      1. makes a f5 special file that is a special block file with the major device number 1 and the minor device number as 9

**Note** that some commands may run for a long time and may create large files. If they do, you'll want to terminate them and delete any files they create (after examining them).

1. Make named pipes using mknod and mkfifo. Use them to simulate ls | wc using just the > and < operators.
   1. wc < namedpipe & ls > namedpipe
   2. Runs the first command in background so that both commands run at the same time and aren’t blocked
2. Make a block device myroot representing your machine's root filesystem. Get info on its filesystem using dumpe2fs myroot. Do you need to be root to do these operations? (Hint: if you use df to find the device, you may have to follow a symlink to get to the real block device.)
   1. Don’t think you d need to
3. Use mknod to make a copy of your current terminal's tty. Examine its characteristics using stty --file=mytty. Do the same for the original tty. How can you figure out what your current tty is? (NOTE: On modern Linux systems the copy often doesn't behave like the original. Why do you think this is the case?)
4. Run stty --help to see what you can do with stty. Try disabling local echo. How does the shell behave with echo disabled? How can you restore echo without logging out and back in?