1. Use your favorite Linux editor to create the `simple` shell script given in class on Monday. (I've posted a "cheat sheet" for `nano` on our class website, if you'd like to use that text editor.) Run it, and see how the content of the script relates to the output. If you already did this on Monday, you're a step ahead!

2. Extend the script so that it generates a random secret number between 1 and 100 and then keeps asking the user to guess the secret number until they guess correctly. The script should give the user hints such as "I'm sorry your guess is too low" or "I'm sorry your guess is too high".

   Hint: `/dev/urandom` is a Linux device that generates a stream of random characters constantly. You can get a random decimal number in the range of 0-255 with the command:

   ```
   od -d -N1 -An < /dev/urandom | head -1
   ```

   What this command does is take the "file" `/dev/urandom` and redirect it as input to the octal dump (`od`) command we talked about earlier. The options to `od` are `-d` to specify a base 10 (decimal) number, `-N1`, only give me one number, and `-An`, don't output an address. You will still need to do some math to get that number between 1 and 100. One way to do math in a shell script is to make use of the calculator, `bc`:

   ```
   echo "your math expression here" | bc
   ```

   16 points total:

Math conversion of random number	4
Loop to allow any number of guesses	4
Loop executes (and exits) correctly	4
Gets user input (correctly)	4

3. Write a shell script called `pidof2` which takes a process name as parameter and returns the process information (not just the process ID) of all processes with that name. You will likely want to see more processes that just yours, so use `ps -lax` to get all processes. A good process to look for is the bash process because everyone who is logged on will be running a bash shell.

   10 point total:

Takes a command line parameter	5
Correctly produces process IDs for that parameter	5

4. Modify your `.bash_profile` script so that your `PATH` includes the current directory (`.`) and so that your prompt variable `PS1` is set to something you like. Run the modified script using `source .bash_profile` and check that the changes you made have been applied to the current shell (type `env`).

4 points total:

$PATH modified correctly	2
$PS1 modified correctly	2

When you are done, you should submit your scripts `simple`, `pidof2` and `.bash_profile` (which you may need to rename to `bash_profile`) to the Moodle dropbox for this assignment. **Put your name in a comment on all files!**