User limits

Disclaimer: Using the Docker Workspace might lead to an issue where the process limit is not counted correctly, such as getting a negative value for that field. This is fine.

Now that you have dusted off your C skills and gained some familiarity with the CS 162 tools, we want you to understand what is really inside of a running program and what the operating system needs to deal with. The operating system needs to deal with the size of the dynamically allocated segments: the stack and heap. How large should these be? Poke around a bit to find out how to get and set these limits on Linux. Modify `limits.c` so that it prints out the maximum stack size, the maximum number of processes, and maximum number of file descriptors. Currently, when you compile and run `limits.c` you will see it print out a bunch of system resource limits (stack size, heap size, etc.). Unfortunately all the values will be 0. Your job is to get this to print the ACTUAL limits (use the soft limits, not the hard limits). *Hint: run `man getrlimit`*

Your output should look something like this:

```
stack size: 8000000

process limit: 10000

max file descriptors: 1000
```

The values above were obtained using the deprecated VirtualBox VM setup. Don’t worry if your values differ, especially if you’re using a different setup. For those using the VM, you may instead get a value of around 1 million for the “max file descriptors”; this is fine.

You can run `make limits` to compile your code. To run the program please use `i386-exec ./limits`.