

# 600.211: Unix Systems Programming

## Midterm 1

Peter H. Fröhlich  
phf@cs.jhu.edu

February 27, 2007  
**Time:** 40 Minutes

**Start here:** Please fill in the following important information using a **permanent pen** before you do **anything** else! Your exam will **not** be graded if you use a pencil or erasable ink on this page.

**Name (print):** \_\_\_\_\_

**Login (print):** \_\_\_\_\_

**Ethics Pledge:** With your signature you **certify** the information above and you also **affirm** the following:  
*“I agree to complete this exam without unauthorized assistance from any person, materials, or device.”*

**Signature:** \_\_\_\_\_

**Date:** \_\_\_\_\_

**Instructions:** Please read these instructions carefully before you start. **Switch off** your phones, pagers, and other noisy gadgets! You are **not** allowed to have anything but a pen (pencil, eraser) and this exam on your desk. You are **not** allowed to talk to anyone during the exam. If you have a question, please raise your hand **quietly**. You must **remain seated quietly** until all exams have been collected. Remember that you can **not** claim grading errors if you do not use a **permanent** pen for your answers.

**Do not open before you are told to do so!**

**You got \_\_\_\_\_ out of 40 points.**

# 1 Binary Warmup

(12 points)

For each of the following statements, determine whether it is either **true** or **false**. (1 point each)

1. The `fork()` function resets the signal disposition of the newly created process to the default action.
2. The `setjmp()` function does not always return twice
3. In C, the instructions `a[i++] = i` and `a[i] = i+1` have the same effect on array `a`.
4. The “execute” bit has meaning for files but not for directories.
5. The `fork()` function does not always return twice.
6. The `lchown()` function is a relic from “the olden days,” it is not used anymore.
7. In C, a function can be declared inside another function.
8. A “hole” in a file is created by seeking beyond the current end of the file and reading data.
9. We cannot change the disposition of the `SIGSTOP` signal.
10. The name of a file is stored only in the directory block.
11. A session is a group of process groups.
12. System calls set `errno` to 0 (that’s zero) on success.

## 2 Tough Choices

(8 points)

For each of the following questions, circle **one** answer out of the choices given.

(2 points each)

1. What is the **key** difference between the `chdir()` and `fchdir()` calls?
  - (a) One takes a path name while the other takes a file descriptor.
  - (b) One follows links while the other does not follow links.
  - (c) `fchdir()` is faster because it performs less error checking.
  - (d) `chdir()` maintains a stack of “previous” directories.
  - (e) None of the above.
2. Which of the following `exec()` calls searches `PATH` **and** allows you to provide an environment?
  - (a) `execl()`
  - (b) `execlp()`
  - (c) `execle()`
  - (d) All of the above.
  - (e) None of the above.
3. Which of the following is **not** inherited from parent to child after a `fork()` call?
  - (a) Process ID
  - (b) Parent Process ID
  - (c) Accumulated execution time
  - (d) All of the above.
  - (e) None of the above.
4. What is so special about the `access()` call as far as **permissions** are concerned?
  - (a) It uses the effective user and group ID.
  - (b) It uses the real user and group ID.
  - (c) It allows us to temporarily get root privileges.
  - (d) It allows us to temporarily avoid permission checks.
  - (e) None of the above.

### 3 Short Answer

(8 points)

For each of the following questions, answer in **one to three** sentences, the shorter the better. (2 points each)

1. What is the difference between a **hard** link created with `link()` and a **symbolic** link created with `symlink()`? Draw a picture if that helps, but be sure to **explain** as many differences as you can think of.
  
2. KERNIGHAN's proposition: "*Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.*" **Discuss.**
  
3. Why are **filters**, i.e. programs that read from the standard input and write to the standard output, such a good idea in the UNIX environment? **Explain.**
  
4. What is a **race condition** as far as UNIX is concerned? How does it manifest? How can you avoid it? **Explain.**

## 4 Piping Shells

(12 points)

Explain how a shell would execute a command line like `ls -l | sort | more` with each “sub-command” running **concurrently** as a separate UNIX process. Feel free to use any and all means at your disposal to make this explanation **easy to understand**. Please provide as much **detail** as you can manage, for example explain **exactly** how pipes and processes are set up and torn down in terms of `pipe()`, `fork()`, and `exec()`. (10 points)

Here are two supplemental questions:

(1 point each)

1. Would it help in any way if the UNIX you were using supported **bi-directional** pipes? **Explain!**
2. How can you tell if a given UNIX returns file descriptors that are **bi-directional** from `pipe()`?

This page is intentionally **mostly** blank in case you run out of space elsewhere. If you ended up here early, please go over **everything** again and remain seated **quietly**! Make sure that the title page is filled out correctly and in **permanent** pen. Maybe you want to "rewrite" your **answers** in permanent pen as well?