CSC209: Software Tools and Systems Programming

http://www.teach.cs.toronto.edu/~csc209h/winter/

Michelle Craig
BA 4258

csc20918s@cs.toronto.edu

Jennifer Campbell
BA 4238
Communication

• **Piazza**
  – Use first for non-personal communication
  – Informative subject lines help
• **Email:** csc20918s@cs.toronto.edu
  – Email is a formal method of communication:
    • Use proper English.
    • State your question clearly, with enough context.
    • Sign it (Name and utorid are the most useful.)

• **Anonymous Feedback**
  – Link from the website

• In person in **Office Hours**
Inverted/Blended Classroom

• Preparation before class (videos & exercises)
• Hands-on Activities in class
  • Need to bring computers to class (or share)
Course Information

• Check the course syllabus on the course web page for
  – Office hours
  – Assignment schedule
• The course web page is the official source of announcements. (Piazza)
  http://www.teach.cs.toronto.edu/~csc209h/winter
• Make sure you have the prerequisites!
## Marking Scheme

<table>
<thead>
<tr>
<th>type of work</th>
<th>probable topic</th>
<th>Weight</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecture Prep (11)</td>
<td></td>
<td>5%</td>
<td>Mondays 9:30am</td>
</tr>
<tr>
<td>Lab Exercises (10)</td>
<td></td>
<td>10%</td>
<td>Fridays 6:30pm</td>
</tr>
<tr>
<td>A1</td>
<td>Shell &amp; Basic C</td>
<td>5%</td>
<td>Jan 25, 8:00pm</td>
</tr>
<tr>
<td>A2</td>
<td>More C (pointers, files)</td>
<td>10%</td>
<td>Feb 15, 8:00pm</td>
</tr>
<tr>
<td>Midterm</td>
<td></td>
<td>10%</td>
<td>Feb 28</td>
</tr>
<tr>
<td>A3</td>
<td>processes</td>
<td>10%</td>
<td>March 15, 8:00pm</td>
</tr>
<tr>
<td>A4</td>
<td>communication</td>
<td>10%</td>
<td>April 5, 8:00pm</td>
</tr>
<tr>
<td>Final Exam</td>
<td>everything</td>
<td>40%</td>
<td>exam period</td>
</tr>
</tbody>
</table>
Assignments

- You will use git to manage and submit your assignments.
- The repositories are already set up.
- Submit your assignment to your repo well before the due date. Then submit again and again as you improve your solution.
- All code **must** work on teach.cs to receive full marks.
- Code that does not compile on teach.cs will get 0.
- See course syllabus for late penalties.
Weekly Lab Exercises

• Held in BA labs (3175/3185/3195)
  – Wed 8-9 pm
  – Fridays 1, 2, or 3 pm
• Starting this week
• Submit your work on MarkUs for credit
  – 1% per lab
  – No credit for attendance
• Due by 6:30pm Friday (no lates accepted)

Complete Lab 0 (not for credit) Now!
Software Installation

- Lab 0 will help
- Take advantage of the help centre
  – M-F 2-6pm in BA2230
Plagiarism

• “The work you submit must be your own, done without participation by others. It is an academic offence to hand in anything written by someone else without acknowledgement.”
• You are hurting your friend when you give him or her a copy of your assignment.
• You are hurting your friend when you ask him or her to give you a copy of their assignment.
What is cheating?

• Cheating is
  – copying parts or all of another student’s assignment
  – including code from books, web sites, other courses without attribution
  – getting someone else to do substantial parts of your assignment
  – giving someone else your solution

• Cheating is not
  – helping to find a bug in a friend’s code (be careful)
  – helping each other understand man pages or example code
What is this Course About?

• Software Tools
  – Efficiently use the *Unix Command Line*
  – Understand the *shell*
  – Use Basic *Shell* Programming
  – Understand and use *Make*

• Systems Programming
  – C
  – files
  – processes
  – communication
Unix Principles

• **Do one basic thing well**
  – with some basic variations
• **Simple input formats**
  – plain text
  – don’t require interactive input
    • stdin to stdout/stderr
• **Simple output format**
  – expected to be input to another tool
Unix Tools Example

• ls, wc, sort, ...
• standard input/standard output
• pipes
Unix is user-friendly; it’s just choosy about who its friends are.
Basic Tools to Learn

- head, tail
- cd
- mkdir
- ls
- cp
- mv
- rm
- diff
- comm
- cut
- cat
- wc
- grep
- who
Unix manual

- man example
- don’t memorize - look it up!
- it grows on you
Shells

$ gcc -Wall -g -std=gnu99 –o hello hello.c

- The $ is a shell prompt.
- Shells
  - accept commands (programs) as input
  - finds the executable
  - interprets the arguments
  - starts executing the command
- Shells also have some “built-in” commands.
Running a program

$ gcc -Wall -g -std=gnu99 –o hello hello.c
$ hello

• load a program into memory and hand it off to the OS to run the program.
Files and Directories

• “Everything is a file.”
• Unix provides a file interface for all Input/Output.
  – regular files
  – directories
  – devices
    • video
    • keyboard
    • sound
    • network
• File interface = open, read, write, close
File System Hierarchy

- Everything starts in the “root” directory whose name is “/”
- A directory is a file that contains directory entries.
- A directory entry maps a file name to an inode.
- An inode is the data structure that contains information about a file, including which disk blocks contain the file data.
Inodes and Directory Entries

Directory Entry

12345  afile

Inode

12345

- size
- owner UID, GID
- access time
- modified time
- creation time
- link and block counts
- permissions

- direct pointers to file blocks
- single indirect pointer
- double indirect pointer
- triple indirect pointer

Pointers to next file blocks
## Directories and Links

<table>
<thead>
<tr>
<th>Directory</th>
<th>Permissions</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Date</th>
<th>Time</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>4096</td>
<td>Nov 8 17:56</td>
<td></td>
<td>bin/</td>
</tr>
<tr>
<td>..</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>4096</td>
<td>Aug 10 14:46</td>
<td></td>
<td>cdrom/</td>
</tr>
<tr>
<td>u</td>
<td>lrwx------</td>
<td>root</td>
<td>root</td>
<td>6</td>
<td>Sep 2 15:32</td>
<td></td>
<td>/cdf/u/</td>
</tr>
<tr>
<td>home</td>
<td>drwxrwsr-x</td>
<td>root</td>
<td>staff</td>
<td>4096</td>
<td>Feb 8 2002</td>
<td></td>
<td>lib/</td>
</tr>
<tr>
<td>lib</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>4096</td>
<td>Sep 2 15:26</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```bash
% ls -l /
```

```bash
drwxr-xr-x  2 root root 4096 Nov 8 17:56 bin/
drwxr-xr-x  2 root root 4096 Aug 10 14:46 cdrom/
drwxrwsr-x  2 root staff 4096 Feb 8 2002 home/
drwxr-xr-x  6 root root 4096 Sep 2 15:26 lib/
lrwx------  1 root root 6 Sep 2 15:32 u -> /cdf/u/
```
Permissions

- **File permissions**
  - read, write, execute – pretty much what you think

- **Directory permissions**
  - read – you can run `ls` on the directory
  - write – you can create and delete files in the directory
  - execute – you can “pass through” the directory when searching subdirectories.
chmod

- **chmod 755 <filename>**
  - 3 numbers between 0 and 7, the octal value for that category of user
  - Quiz — what is the command to set the permissions of the file classlist to be world readable but writeable only by the file owner and members of the group.

- **Another approach**
  - chmod u+rwx
  - chmod go-x
  - adds or removes permissions for those categories of users
Globbing

- A little like regular expressions but different syntax
- * matches any number of any character
- ? matches any one character
- [list of characters]
- [1-5] or [a-z] or [a-xz]