Operating Systems

CIS*3110 (W18)

Assignments

There are three assignments. Each assignment is due by **23:55h (i.e. 5 minutes before midnight)** on the day it is due. Assignments, including any documentation, are submitted electronically. Late assignments are not accepted, and will be assigned a grade of 0. Assignments are to be done individually.

	Topic	**Due date**
Assignment 1	Writing a Shell	*Monday, Jan. 29, 2018*
Assignment 2	CPU Simulation	*Monday, Feb. 26, 2018*
Assignment 3	Mutual exclusion and Memory management	*Monday, Mar. 26, 2018*

Guidelines and Policies for Assignment Submission

- Assignment Submission :

  The following should be handed in for each assignment submission:

  - **Source Code**: Turn in all source code you have written (including any code provided to you) along with any other files necessary to compile and run your program (i.e. Makefile, etc.). Source code should be appropriately commented and follow reasonable style guidelines (see note below). You should only submit files required for the compilation of your code - temporary files, back-ups or any other spurious files should not be present.
  - **Documentation**: In addition to any documentation that may be specified by an assignment, a file called *README* should always be present containing, for each file submitted, a one sentence description of its contents; your *README* file may also include any relevant notes for the marker.
  - Assignments are submitted electronically via the department Moodle server (a submission dialog appears at the bottom of the assignment link under Moodle once submissions are open):
    1. organize your submissions directory (include all files that are to be submitted and ensure your name and student ID appear in the heading of every file you submit).
    2. run `tar` in that directory to turn everything into a single file named as Lastneme_firstname_Assignment#, such as Li_Xining_a1.

3. run `gzip` to compress the resulting tar file.
4. use a web browser to log in to the *Moodle* system and choose the appropriate assignment for submission; you will be prompted to select the file that you are submitting after which you will confirm the submission.
5. **note**: you can view the status of your submission at any time through the *Moodle* system, so you can verify that you have submitted correctly; for the record, we will not accept "I thought I submitted it" as a reason for non-submission.

- **Other Requirements**: Any other requirements as specified by a given assignment must be satisfied completely.

- Requests for re-grade:

  If you wish to have an assignment re-graded, you have seven (7) calendar days from the day results are made available to the class to request one. Re-grade requests are made in-person, during scheduled TA office hours. **No requests for re-grade will be handled by e-mail**. You must make any materials relevant to your request available to the TA at the time of the request. If the TA is unable to resolve your problem to your satisfaction, you may then appeal to your instructor during office hours within seven (7) days of your initial re-grade request.

  Absolutely no re-grades will be considered for any reason after the initial seven day period.

- Coding Style Guidelines :

  Part of your mark will reflect coding style. Although there is no single correct coding style, you should strive to make your code as readable and internally consistent as possible. Such programs are easier to test, debug, maintain and modify. Please consider the following points when writing code:

  - All source files should have a comment header describing the purpose of the module/program; similarly, all subroutines should have a comment header clearly describing its function.
  - All identifiers (variables, subroutine names, etc.) should have descriptive names, with the possible exception of loop indices.
  - Literal constants (or so-called "magic numbers") are to be avoided---used named constants for counting limits, array sizes and so forth.
  - Indentation and use of whitespace should be consistent throughout the code.
  - Ideally, a subroutine is a small block of code that performs a single function. If a

subroutine is getting too large or is doing too many things you should probably separate it into several separate subroutines.
    - Commenting should be appropriate (i.e. you would not comment an increment statement as what it is doing is obvious), and assist someone unfamiliar with your code to understand what it is doing.

- Operating Environment :

  Programming assignments are to be implemented only in the language or with the specific tools specified by the assignment in question, and are required to compile and run using the software available under Linux in the undergraduate lab environment provided by the department. You are free to develop your programs on any platform you wish, however it is entirely your responsibility to ensure that by the time it is handed it runs on the department lab systems.

  Code that does not compile, cannot be run productively in the specified university lab environment or that uses forbidden features/functionality will be assigned a grade of 0.

- Late submissions:

  Late assignments are not accepted, and will be assigned a grade of 0.