CSCE 3600 Syllabus

Instructor: Philip Sweany
Office: NTDP F262
Phone: 940-369-7427
Office Hours: To be determined
Email: sweany@cs.unt.edu

Textbook:
System Programming in C
Adam Hoover

Course Description:

CSCE 3600 maintains a focus on systems programming, both from the standpoint of learning about how computer system software works and learning/improving students’ programming skills in K&R C and bash. The course will include a significant programming project that will be completed in groups as well as several smaller programming assignments to be done individually.

Course Outcomes:

Course outcomes are measurable achievements to be accomplished by the completion of a course. Outcomes are evaluated in ABET’s accreditation process.

1. Write robust, efficient, readable and correct system software using the C programming language.
2. Demonstrate an understanding of processes and threads by developing applications using multiple processes and multi-threaded activities in a Linux environment.
3. Demonstrate an understanding of deadlocks and synchronization through the development of application(s) that utilize a variety of mutual exclusion mechanisms.
4. Develop shell scripts and utilities that demonstrate an understanding of memory, file and process management and interaction, including concepts such as virtual memory and disk scheduling.
5. Create a Linux-based application that utilizes inter-process communication mechanisms such as pipes and sockets to communicate information between independently running processes on one or multiple platforms.
6. Demonstrate an understanding of the use and interaction among compilers, macro processors, assemblers, linkers and loaders through their use in creating the applications described in previous outcomes.

Policies:
• Most programs are due at 11:59pm on the due date. Sometimes, when we use the class time as a lab session, program(s) may be due by the end of class.
• No late programs will be graded
• All programs will be submitted to Moodle.
• Except for the start of the term, attendance will not be taken in lecture. However, your attendance is strongly recommended to improve your opportunity to meet course outcomes.

Grading: The components of your grade will be weighted as follows:
• 1 midterm exam, 20%
• 1 final exam, 30%
• Minor programming assignments and labs, 30%
• One Major Program, done in groups, with 3 during the term, 20%

Make-Up Policy:

There will be no make-up exams, labs, or programs given in this class. However, for documented excused absences or emergencies on a day of an exam or a lab the exam and/or lab grades will be replaced by an average of the other exam or lab scores. There is one exception to this rule. Under NO circumstances will more than one exam or lab score be replaced by an average of the other scores. For a second (and subsequent) missed exam or lab, even if all are excused, students will receive a 0 for the missed work.

Excused Absences

Students are expected to schedule routine appointments and activities so as not to conflict with attending class. However, some absences cannot be prevented. In the event of a medical emergency or family death, students must request an excused absence as quickly as feasible following the emergency. Use common sense. Students must provide documentation that verifies an emergency arose.

Emergencies

By definition, emergencies cannot be planned for. Your instructor attempts to make accommodations in these instances that allow for making up missed work and completion of the course in a timely manner. Among these emergencies are

• A death in your immediate family
• An accident or illness requiring immediate medical treatment and where a doctor has indicated attending class is impossible or inadvisable.
• Employees who are on call 24/7 fall in this category but must document that they were called during a scheduled class.

Collaboration and Cheating:

On minor programs assigned in lecture, you should work alone. Do NOT work with other students on shared program solutions. Do NOT get help with algorithms or coding from anyone other than Dr. Sweany or the 3600 grader or peer mentor. Do NOT use even partial program solutions from the internet unless those partial solutions are provided to you by Dr. Sweany as part of the assignment description. Failure to adhere to these strict standards will be cause for disciplinary action that could be as severe as expulsion from the university.

It IS permissible to obtain help from whoever you wish to fix syntax errors. But remember, for anything but syntax errors, getting programming assistance from any source other than Dr. Sweany or the 3600 grader or peer mentor will be considered cheating and dealt with harshly.

And, of course you need to do your own work on quizzes and exams as well. Here there should be no ambiguity at all.

In case the above description, and in-class discussion of my views on appropriate and inappropriate collaboration does not answer all of your questions, please look at the university Student Rights and Responsibilities web page.

SETE

The Student Evaluation of Teaching Effectiveness (SETE) is a requirement for all organized undergraduate classes at UNT. This short survey will be made available to you at the end of the semester, providing you a chance to comment on how this class is taught. I am very interested in the feedback I get from students, as I work to continually improve my teaching. I consider the SETE to be an important part of your participation in this class.

ADA

UNT complies with all federal and state laws and regulations regarding discrimination, including the Americans with Disability Act of 1990 (ADA). If you have a disability and need a reasonable accommodation for equal access to education or services please contact the Office of Disability Accommodation.
## Tentative Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
<th>Major Program Submission</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/13/14</td>
<td>Introduction, Networks</td>
<td></td>
</tr>
<tr>
<td>1/20/14</td>
<td>Regular Expressions and DFAs</td>
<td></td>
</tr>
<tr>
<td>1/27/13</td>
<td>Minimal DFAs</td>
<td></td>
</tr>
<tr>
<td>2/3/14</td>
<td>Program Execution</td>
<td></td>
</tr>
<tr>
<td>2/10/14</td>
<td>Processes</td>
<td></td>
</tr>
<tr>
<td>2/17/14</td>
<td>Concurrency</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; Program report and submission 2/21/14</td>
</tr>
<tr>
<td>2/24/14</td>
<td>Concurrency</td>
<td></td>
</tr>
<tr>
<td>3/3/14</td>
<td>Midterm Exam</td>
<td></td>
</tr>
<tr>
<td>3/17/14</td>
<td>Streams, Pipes, Files</td>
<td></td>
</tr>
<tr>
<td>3/24/14</td>
<td>Bash</td>
<td>2&lt;sup&gt;nd&lt;/sup&gt; Progress report and submission 4/4/14</td>
</tr>
<tr>
<td>3/31/14</td>
<td>Bash</td>
<td></td>
</tr>
<tr>
<td>4/7/14</td>
<td>Process Scheduling</td>
<td></td>
</tr>
<tr>
<td>4/14/14</td>
<td>Compilers</td>
<td></td>
</tr>
<tr>
<td>4/21/14</td>
<td>Virtual Memory</td>
<td></td>
</tr>
<tr>
<td>4/28/14</td>
<td>Dynamic Memory Management</td>
<td>Final submission 5/2/14 @ 11:59PM</td>
</tr>
<tr>
<td>5/5/14</td>
<td>Finals</td>
<td></td>
</tr>
</tbody>
</table>