Introduction to Operating Systems
CMPS 111

Dr. Karim Sobh
Computer Science Department
Jack Baskin School of Engineering
ksobh@cs.ucsc.edu

Fall 2016

Basic Information

- **Course Title:** Introduction to Operating Systems (CMPS 111)
- **Prerequisites:** Algorithms and Abstract Data Structures (CMPS 101), and Comparative Programming Languages (CMPS 112), or Computer Architecture (CMPE 110)
- **Lectures:** MWF 02:40 PM - 03:45 PM (Merrill Acad 102)
- **Instructor:** Dr. Karim Sobh (ksobh@ucsc.edu)
  Office Hours: 16:00 - 17:30 MTW (E2-255)
- **TAs:** Aneesh Neelam (aneelam@ucsc.edu)
  Office Hours: Tu 12:00 - 13:00, Th 16:30-17:30 (E2-480)
- **Labs:** (Kresge 317)
  Mondays 16:30 - 17:30
  Thursdays 15:30 - 16:30
- **Home page:** `https://cmps111-fall16-01.courses.soe.ucsc.edu/`
- **Discussion:** `https://piazza.com/ucsc/fall2016/cmps111`
- **Git Repository:** git@git.soe.ucsc.edu:classes/cmps111/fall16/cruzid
- **Git How-to:** `https://git.soe.ucsc.edu/~git/index.html`

Catalog Description

Fundamental principles of operating systems: process synchronization, deadlocks, memory management, resource allocation, scheduling, storage systems, and study of several operating systems. A major programming project will be required.

Textbooks and References

The primary textbook is ***Modern Operating Systems***, by Tanenbaum, make sure to get the 4th Edition (ISBN 013-359162-X). It covers the main topics of uniprocessor operating systems in the early chapters, and then covers distributed systems and other advanced topics in the later chapters. The second book, ***The Design and Implementation of the FreeBSD Operating System*** will be needed extensively for the assignments, and we will be referring to specific chapters that will need to read to be able to do the assignments.

- Modern Operating Systems, 4th ed. , Tanenbaum & Bos (ISBN 0-13-359162-X)
- The Design and Implementation of the FreeBSD Operating System, 2nd ed. , McKusick, et al.

Course Goals

Students that take this course should learn the fundamental principles of operating systems. The course covers the various important aspects, characteristics, and design approaches of operating systems in general. The presented concepts will be complemented with examples from real modern operating systems such as FreeBSD.

The is not a theoretical course by any means and the student should be prepared to get his/her hands dirty as there will be a considerable amount of code writing involved. During the course students should learn how to read already existing operating systems code, reverse engineer it, and be able to hook it and add new or modified functionality to it. The assignment projects are designed to implement concepts and implementation details explained in class through amending and modifying the kernel code of the FreeBSD open source operating system.

Major Topics Covered

- Operating system concepts and fundamentals.
- Processes, thread, inter process communication, and scheduling.
- Synchronization and deadlock.
- Memory management.
- I/O devices and interrupt handling.
- File systems.
- Multi-core systems.
- Introduction to distributed operating systems.
- Virtualization.
- Protection and security (if time permits).

Exams

Students must attend all exams, the midterms and the final. The final is a comprehensive exam with emphasis on material that were not covered in the mid-term. In case of absence due to emergency, the student needs to let the professor know before the scheduled exam date. Moreover, the student will be required to present a legitimate justification as proof such as a **doctor's note or letter from the funeral home** before taking a makeup for the exam.

Assignments Projects

The student should submit all the assignment work to the git repository. Most of the assignment projects will involve modifying the FreeBSD kernel and hence extensive use of virtual machines will be needed to be able to perform the required tasks reliably. The student can choose any hypervisor of choice as long as it can run on x86-based architecture, although VirtualBox is recommended as it is free and runs on almost all platforms.

Most of the assignment projects will be done in teams, 3-4 students, which will be assigned by the course staff, and will not be fixed across all the team assignments. **Assigned teams cannot be changed unless more than one team member drops the class**.

Each assignment project will have a due date that will be announced with the assignment, and late submissions of the assignments will result in a deduction penalty of 10% per day, with a maximum of 5 late days (including holidays, and week-ends) after which the assignment will not be accepted. With the same concept, assignments submitted earlier than the deadline will be credited 1% for every 12 hours (half-day), with a maximum of 10% credit. The maximum grade that the student can attain is 100% irrespective of how early assignment project is submitted.

It is highly recommended to start immediately, and as early as possible, as assignment projects will require time to complete as per requested. Each assignment project can only be submitted once, and should have all the needed information that will help the grader evaluate the work such as code, in-code documentation, explanation of the design approach and reasons why specific routes were taken, etc. Simply, it is the responsibility of the student to showcase his/her work.

More details about the assignment projects will be available towards the beginning of the quarter.

Homework

A number of optional ungraded homework problems will be assigned on average every two weeks. This will give the student the chance figure out his/her weak areas for further help by the course staff.

Notes and Class Participation

Students are required to take notes and participate in class. Part of the grade is attributed to how active the student is. Weekly notes must be turned in on eCommons latest by Sunday 9 PM of the week after the material was covered. The notes can be hand written or typed by **the student**, and they should not be copied form textbooks, class slides, other students, or the internet.

There are many forms of participation, mainly class attendance, actively participating in class discussions, visiting course staff during their office hours, and/or participating in the piazza discussion. The student is encouraged to perform all of these as it is of great benefit to the student and might enhance the final grade.

Grading

The course overall grade is split over two components; exams and programming assignments. A student needs to get at least 50% of the total grade of each component to pass the course; Each component needs to be passed. For example the following cases will definitely fail the course:

- Scoring 55% in the exams and 51% in the assignments.
- Scoring 100% in the exams and 30% in the assignments.
- Scoring 28% in the exams and 90% in the assignments.

The final grade will be calculated as follows:

- Homework Assignments: 45%
- Midterm Exam: 20%
- Final Exam: 30%
- Participation & Attendance: 5%

The following are tentative approximate ranges of the overall scores:

- A: 89-100%
- B: 79-89%
- C: 69-79%
- D: 60-69%
- F: below 60%

Individual exams, or assignment can be curved based on the situation and the overall distribution of the grades. Also, extra bonus work might be offered near the end of the course based on the grades distribution but with no guarantees.

Attendance

Students are required to attend the lectures. Exam questions will be designed to refer to lecture discussions and examples, and will require knowledge of specific details discussed in the lectures. Students failing to attend the lectures have a high chance of loosing marks on the exams. By not attending the lectures the student will also find it difficult to take good notes. Lab section attendance is not required, although important material on the programming projects will be missed if the student does not attend, since that will be where projects will be discussed in detail.

Getting Help

The student is advised to always ask for help as early as possible and not to wait. One way of getting help is through engaging in informative discussions with the course staff during their office hours. However, for these discussions to be fruitful the student should:

- Attend classes and lab sections.
- Read the course Web page for information on assignments.
- Read and post to the class discussion forum, hosted at piazza.com

The student should try to avoid dropping by outside the office hours as the course staff might be busy and might not be able to avail the time to help at the time. If the student cannot attend office hours, the student should arrange a meeting in advance by emailing the person he/she would like to meet with. Questions can also be posted to course staff via email as long as they can be replied to in short answers.

Students with Disabilities

If the student qualify for classroom accommodations because of a disability, sthe student should submit the Accommodation Authorization Letter from the Disability Resource Center (DRC) to me **as soon as possible**, preferably within the first week of the quarter. Contact DRC by phone at 831-459-2089 or by email at drc@ucsc.edu for more information.

Academic Honesty

Academic Honesty is a very important aspect in academic life. The students are expected to respect and adhere to the highest levels of academic integrity standards. There are many forms for cheating and all of them are considered a violation, which means that plagiarism is not accepted by any means. Consequently, students are not allowed by any means to do any of the following or anything similar:

1. Work on assignments with anyone.
2. Share code with anyone.
3. Share material or notes with anyone.
4. Share written class notes with peers or anyone.
5. Obtain help in assignments from anyone other than the assigned course staff.
6. Use of online past material that might help in solving the assignments.
7. Request help through online forums to help solve assignments.

If the student obtains help in any aspect of the practical work of the course the student is encouraged to document the source. In such cases it will not be considered or reported as cheating, nevertheless it might lead to a lower grade for that. So documenting the sources when turning the work in is highly advisable, as it will not count at all if it is caught afterwards.

By **anyone** we mean friends, family, former or current Computer Science students, other humans, animals, zombies, sparkling vampires, materials found on the Internet. Any cheating attempt in the assignments or the exams will result in failing the course. The Student is encouraged to read `http://registrar.ucsc.edu/navigator/section1/academic-integrity.html`. Moreover, the Academic Misconduct Policy for Undergraduates will be applied; the student should refer to `https://www.ue.ucsc.edu/academic\_misconduct` for more information. The policies will be applied on all parties participating in a misconduct, both the transmitter and the receiver of the information and material in any form.

The bottom line is: do not ever cheat, or you will definitely fail the course.