Fall 2005

CEG 333: Introduction to Unix

Travis E. Doom
Wright State University - Main Campus, travis.doom@wright.edu

WRIGHT STATE UNIVERSITY
Wright State University
College of Engineering and Computer Science
Department of Computer Science and Engineering

CEG 333 - Introduction to UNIX

Fall Quarter, 2005

Professor: Travis E. Doom, Ph.D.
Professor's Office: 331 Russ Engineering Center
Office Hours: 2:00-3:00 TR. Other office hours by appointment (via email).
Email: (Preferred contact) travis.doom@wright.edu
Office Phone: (937) 775-5105

Room & Time:
Section 01: 2:45 - 3:35 MW 146 Russ

Course Description:
Introduction to the use of UNIX and UNIX tools as a problem-solving environment. Emphasis on the shell, files and directories, editing files, user process management, compiling, and debugging. Prerequisite: CS 241.

Textbook:
Required: Das, Sumitabha. "Your Unix: The ultimate guide." McGraw-Hill, ISBN 0-07-252042-6.

Prerequisites: CS 241
1. Basic concepts of programming, including conditional statements and iteration.
2. C or C++.

About this course: All operating systems are designed to help the user solve problems. An operating system is a complex program that allows the user to make use of computer hardware in a certain way. If a different operating system is used on the same computer hardware, then dramatically different tools may be available to solve problems using the same computer hardware. Most common operating systems today (particularly Windows) are designed to help a "standard" computer user solve "standard" problems. Thus, these operating systems come with relatively few general-purpose tools. These operating systems are designed primarily to be easy to use - not to help their users solve complex problems. Most "tools" that the users might wish to use to solve problems in these environments are stand-alone applications that have to be purchased, found, or created by the user. UNIX, on the other hand, provides many powerful tools to solve problems, at the cost of being more difficult to learn. You may be asking yourself, "Why should I learn UNIX, when all I really need to know is Windows?". To help you answer this question, consider the following points:

1. Programming languages, such as Pascal, Visual Basic, C++, Java, and PERL, each facilitate a unique set of programming tools and approaches. Some problems are "easier" to solve using the tools provided by one programming language (say Visual Basic) then using the tools provided by

a different programming language (say C++). This doesn't imply that Visual Basic is better than C++, only that it is easier to solving some classes of problems with Visual Basic. An operating system, like a programming language, provides a set of tools for solving problems. Many problems are far easier to solve with the tools available on a UNIX platform. It is always to your advantage to know what your options are before attempting to solve a complex problem.

2. UNIX was designed by computer scientists, for computer scientists. UNIX has exceptionally powerful tools, created by very, very smart people over a period of several decades. Although it is harder to learn than many operating systems, it is correspondingly more powerful.
3. UNIX runs on virtually every computer ever made (including handheld computers, cell phones, watches, and ipods). Although there are variations (different versions) of Unix, each provides fundamentally the same set of commands and tools.
4. Many variations of UNIX are open source. Thus, you can obtain the actual source code for the operating system and see exactly how it works. If fact, you can modify the operating system to meet any special needs that you might have. Because of these facts, UNIX machines are heavily used in both academic and industry settings.
5. Most modern operating systems are based on or use ideas that were first implemented in UNIX. Most modern OSes provide a graphical interface over a very UNIX-like functionality (consider Mac OS-X!). Thus, the UNIX operating system is the de facto standard when discussing operating system strategies, techniques, and performance.
6. Learning multiple operating systems is crucial to being a well-rounded computer scientist. So crucial, in fact, that the Computer Science Accreditation Council mandates that to receive a Bachelor of Science in Computer Science, a student must be familiar with multiple operating systems. Almost every university throughout the world teaches its students UNIX. UNIX is the "standard" for operating systems.

Computer science students everywhere are expected to be able to operate comfortably and program in the UNIX environment. UNIX may take a considerable amount of time and effort to master. Like most powerful tools, it has a steep learning curve. Don't worry. It will be well worth the time and effort.

Objectives: This course has two primary sets of objectives. The first set is content-based. We hope to teach students the fundamental principles of UNIX and design in the UNIX environment. At the end of the course, each passing student should be able to:

1. Describe the basic methodology of UNIX filters, including pipes and redirection of stdin/stdout.
2. Program simple UNIX utilities at the command-line and shell-script level.
3. Discuss the advantages and disadvantages of common user interfaces (such as UNIX vs. PC/NT).
4. Discuss the philosophy of UNIX development and the open source movement.

The second objective is skill-based. Students will exercise their ability to apply these principles in practical application though laboratory projects. At the end of this course, each passing student should be able to:

1. Work comfortably in the UNIX environment.
2. Edit and manage files and user-level security for UNIX development.
3. Use standard UNIX development tools for C or C++.

Grading: A student's demonstration of their ability to discuss issues, solve problems, and demonstrate mastery of course topics will be the underlying metric for the determination of a student's overall grade in this course. Students will be provided the opportunity to demonstrate their mastery through examinations and laboratory projects. Grades will be assigned on a standard A/90%, B/80%, C/70%, D/60%, F/60% scale. Clustering of grades may cause the thresholds to be lowered; they will not be raised. The instructor reserved the right to fail any student who does not a student attain both a passing grade (70%+) in the laboratory and at least a grade of 50% on the final. The overall course grade will be

the weighted sum of the three grades:

30% Laboratory Projects
40% Two Midterm Examinations
30% Final Examination

Laboratory Projects: The laboratory projects are designed to help you learn the course concepts and are the primary course "homework". Points will be deducted for projects submitted late. No points will be awarded for projects that are more than one week late. Corrupt files or other computer problems will not be considered a sufficient excuse to extend this deadline. It is your responsibility to back-up your work. I strongly suggest that you save your work to multiple locations/media to aid in the recovery of corrupt files.

Midterm examinations will occur at the normally scheduled class time and location unless announced otherwise in class. The final examination is cumulative and will take place during the university scheduled time period in the normally scheduled class location unless announced otherwise in class.

Academic Integrity : Student-teacher relationships are built on trust. For example, students must trust that teachers have made appropriate decisions about the structure and content of the courses that they teach, and teachers must trust that the assignments which students turn in are their own. Acts which undermine this trust undermine the educational process. It is the policy of Wright State University to uphold and support standards of personal honesty and integrity for all students consistent with the goals of a community of scholars and students seeking knowledge and truth. Furthermore, it is the policy of the university to enforce these standards. The following recommendations are made for students:

1. Be honest at all times.
2. Act fairly towards others. For example, do not seek an unfair advantage over others by cheating with or by looking at other individual's work during examinations or laboratory assignments.
3. Take group as well as individual responsibility for honorable behavior. Collectively, as well as individually, make every effort to prevent and avoid academic misconduct, and reports acts of misconduct that you witness.
4. Know the policy -- ignorance is no defense. Read the policy contained in the student handbook. If you have any questions regarding academic misconduct, contact your instructor.

Students are encouraged to get together in small study groups to discuss the course topics and homework problems. Small group discussion and collaboration is a vital aid to mastering the concepts presented in this course. Being able to communicate and work in teams is a necessary skill for any engineer. However, **students must work on all graded course assignments and examinations on an *individual* basis.**

Absences: Class attendance will not be a direct factor in your grade but will strongly effect the quality of your education. Students are expected to attend every class. Things may make less sense to students that do not attend class or arrive late. Students who miss class are responsible for the material or announcements presented. Any extenuating circumstances which impact on your participation in the course should be discussed with me as soon as those circumstances are known. Make-ups for laboratory demonstrates may be arranged if a student's absence is caused by documented illness or personal emergency. It is the student's responsibility to provide a written explanation (including supporting evidence) to the instructor in a timely manner. Students registering after the term begins are responsible

for all missed assignments and cannot expect that due dates will be altered.

Additional Needs: Students with disabilities or any additional needs are encouraged to set up an appointment at their convenience to discuss any classroom accommodations that may be necessary.

CEG 333 Web Page: http://www.wright.edu/~travis.doom/courses/CEG333

Dr. Travis Doom, doom@cs.wright.edu.

Last modified: 08/13/05

WRIGHT STATE UNIVERSITY
Wright State University
College of Engineering and Computer Science
Department of Computer Science and Engineering

CEG 333 - Introduction to UNIX

Fall Quarter, 2005

Introduction to UNIX		
DATE	**TOPIC / ACTIVITY**	**HOMEWORK ASSIGNMENT**
Week 1	Introduction to the UNIX Environment; Basic UNIX commands; Editing and printing	Read Ch. 1, Ch. 2, Ch. 5 (vi), Ch. 6 (emacs); Set up UNIX account
Week 2	Files, file managment, and permissions;	Read Ch. 3, Ch. 4; File manipulation lab
Week 3	The shell; filters as tools	Read Ch. 7, Ch. 10; Filter lab
Week 4	Midterm Examination I; Programming Enviroment	Debugging lab
Week 5	Compiling and debugging in UNIX	Read: Ch. 17; Shell lab I (in C)
Week 6	Processes and multitasking	Read: Ch. 8, Ch. 18; Shell lab II (in C)
Week 7	Shell programming	Read: Ch. 13; Shell programming lab
Week 8	Midterm Examination II; Internet utilities	Read: Ch. 14
Week 9	Programming utilities	Read: Ch. 11, Ch. 16
Week 10	Just enough Administration	Read: Ch. 19

Final Examination		
DATE	**TOPIC / ACTIVITY**	**HOMEWORK ASSIGNMENT**
W 11/16	Section 01: Final examination, 3:15 - 5:15	Regularly scheduled class room

NOTE: As this is Dr. Doom's first offering of this course, the schedule and assignments will be announced in class. This page was last modified on Wednesday, 24-Aug-2005 10:43:08 EDT. Assignments prior to this date should be accurate. Assignments listed after this date are *projections* and may not correspond to the actual material and assignments presented in class.

The most recent version of this document is available on the world wide web via:
http://www.wright.edu/~travis.doom/courses/CEG360

Dr. Travis Doom, doom@cs.wright.edu.

Last modified: 08/24/05