

# Course Home Page and Syllabus

## UTEP CS 4375: Theory of Operating Systems

(and sometimes EE 4374: Operating Systems Design)

**Quick links:** [Course description](#), [Course objectives](#), [Text & Email](#), [Labs](#), [Exams](#), [TA](#), [Instructor](#), [Course outline and lecture notes](#), [policy](#).

### Contents

[1 Course Description](#)

[2 Instructors:](#)

[3 Teaching assistants](#)

[4 Exams, labs, and grading](#)

[5 Accommodations for Students with Disabilities and Exceptional Circumstances](#)

[6 Standards of Conduct and Academic](#)

[Honesty](#)

[7 Course Grading Policy](#)

[8 Expectations of UG/Grad Students](#)

[9 Tools](#)

[10 Linux under virtualization](#)

## Course Description

Most students enjoy this course because it exposes so much of the "magic" of how programs and resources are managed on modern network-connected computer systems.

The course's [current learning outcomes](#) were approved in Fall 2019. They are organized around the following inter-related themes

- Virtualization: how achieved, the relationship between policy and mechanism.
- Concurrency: what problems it solves, how it is exposed, hazards of poorly coordinated concurrency, and strategies for its effective coordination..
- Address families and namespaces. how the imposition of suitable namespaces enables effective and scalable management of memory, communication, and computation
- Encoding, persistence, and communication: How data is represented and stored, and communicated. Includes an examination of the abstraction layering which enables filesystems and (end-to-end) communication protocols.
- Mature programming: the ability to design comprehensible programs that use appropriate data structures, algorithms, and abstractions.

These functions of a modern computing systems are primarily derived from the composition of a few fundamental concepts. This course teaches these key concepts and "makes them real" by engaging students in the creation of small programs that implement their key functionalities.

**The relationship between lectures, labs, and quizzes.**

Lectures for this course will survey the major ideas enabling the construction of operating systems and networking, and connect them to the abstractions and interfaces they provide to programmers.

Labs for this course are intended to enable students to focus on the design of small programs that exploit operating system and network services. In Order to facilitate high level analysis (and to minimize the risk of students becoming overwhelmed by nuisance implementation details), lab assignments must be implemented using the Python (v3.\*) programming language. An short initial programming assignment introduces students to aspects of Python relevant for this course. Labs will be graded for correctness. Lab-related learning outcomes (primarily the ability to exploit OS interfaces to achieve pragmatic goals) will be substantially measured by test instruments (e.g. in class quizzes and exams).

The course includes frequent quizzes, a super-quiz (that serves as a midterm) and a final exam. All grading is explicitly mapped to course learning outcomes, which are generally assigned Boolean scores indicating whether the students' work indicates an appropriate level of mastery. In order to appropriately assign credit for skill mastery in the context of clerical errors, "Clerical accuracy" is also included as an additional skill that is graded separately to enable us to provide full credit for skill mastery when clerical errors result in incorrect answers.

## [Text & Email list](#)

- **Text:**
  - Required
    - Operating Systems: Three Easy Pieces, by Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau.
      - Available free online at <http://pages.cs.wisc.edu/~remzi/OSTEP/>.
      - Hardcover (\$36), softcover (\$24), and electronic (\$10) copies are also available from the same link.
    - Peter Dordal's [Introduction to Computer Networks \(free & web viewable\)](#)
  - Supplementary
    - Prior years: Tannenbaum's "Modern Operating Systems."
      - The 4<sup>th</sup> edition is current.
      - The 2<sup>nd</sup> or 3<sup>rd</sup> editions are very similar (and adequate for this course). In June 2015, Amazon was selling the 2<sup>nd</sup> edition at \$0.88.
    - Kernigham & Ritchie, *The C Programming Language*, any edition.
    - Michael Kerrisk's The [Linux Programming Interface](#) (No Starch Press)
    - Nick Parlante's, [Pointers and Memory](#) (*pdf download*)
- Communication:
  - Public conversations among students & teaching team: Via microsoft teams (you will be automatically added)
  - To communicate with the teaching team: 2019 Fall OS teaching team: [f19-os-teaching-team@googlegroups.com](mailto:f19-os-teaching-team@googlegroups.com)

## [Instructors:](#)

- [Dr. Eric Freudenthal](#)
  - email: efreudenthal at utep.edu
  - phone: 915 747 6954

- Office Hours: See my [homepage](#)
- Daniel Cervantes
  - email: drcervantes@miners
  - Office Hours: MW 10-11:30am

## Teaching assistants

- All TA Office Hours will be held in the CS TA Lab (CCSB 1.0706)
- TAs will be present for the first 10 minutes of their office hours and depart if nobody is present or expected.
- If you need to meet a TA at a particular time, please send them email.
  - If nobody is present or expected, they may leave.
  - If you need to meet a TA at another time, send them email.

TA Schedule	Aneesa Judd
Email	ahjudd@miners
Office hours	MW 2:00 - 3:00 pm F 1:00 - 3:00 pm (CCSB 1.0704)

## Exams, labs, and grading

- Course schedule
  - The [course outline and schedule](#) indicates the approximate sequence of assigned readings, labs, and homework exercises. Students are expected to
    - Skim readings prior to the class where they are discussed including attempting listed exercises.
    - Notify the instructor prior to or during class about topics and exercises they found confusing.
    - Review readings & homeworks after the class they are discussed.
- Exams/quizzes:
  - Quizzes
    - Quizzes assess individual students' abilities to demonstrate knowledge, to design solutions to realistic problems, and to present these solutions in a clear and professional fashion. Quizzes will be graded "by skill" (see below), can cover any concept or skill previously taught in the course, are generally offered at the rate of once per week, and unannounced (so students must be continuously prepared).
  - Final Exam
    - The final exam date is scheduled by the university based upon lecture time. Like quizzes, the final exam will be graded "by skill" (see below) and can cover any concept or skill previously taught in the course.
  - Rules
    - Solutions must be prepared individually without communication with or assistance from anybody except the instructor or proctor.
- Labs
  - Frequency & Deadlines
    - Labs will be assigned during the lab course

- Due 1 week after assignment unless otherwise indicated.
- Intention:
  - Labs are intended to provide an opportunity for students to practice and explore the application of concepts presented in class within programming assignments.
- Students are expected to act professionally
  - By reading whatever resources they find relevant
  - By attributing credit to any person or reference materials that substantively contributed to their solutions
  - By only submitting solutions they fully understand.
  - Professionalism includes honesty, clarity, and accuracy.
  - Students are encouraged to help each other select and design problem-solving approaches, but they may not present solutions to each other for duplication or paraphrasing.
- Requirements
  - **Functional:** Assignments will either require students to create complete programs or modify programs provided by the instructor.
  - **Documentation:** Submissions should include documentation that facilitates the grader's determination of
    - How to compile, use, and test
    - Principles of operation (e.g. comments & other descriptive prose)
    - Elements of the submission that were developed by others. This includes both algorithms and code. Vague attributions of credit (e.g. "Assistance was received from Jim Smith.") are unacceptable.
  - **Originality:** Any code not provided by the instructor must be "original work" by the student. As in written "essay" assignments,
    - Each instance of a specific algorithm, code fragment, comment, or explanation composed by or with another person must be individually documented.
    - The majority of "original work" must not be composed of verbatim or paraphrased copies of code or comments composed by somebody else.
  - **Completeness:** Students who labs that do not substantially satisfy functional and documentation requirements will receive no credit.
- Homework:
  - Most class sessions will conclude with an assignment due at the beginning of the next class session (unless otherwise indicated).
  - While most assignments are neither collected nor graded, student mastery of relevant skills will be tested within quizzes and tests.
- **More notes on grading policy**
  
- Submission policy
  - Labs will only be accepted on the published due date unless other arrangements are made.
  - All lab assignments must be completed to earn a passing grade.
  - All labs must be submitted using the version control system. Labs will not be accepted via email.

# Accommodations for Students with Disabilities and Exceptional Circumstances

Individuals with disabilities have the right to equal access and opportunity. Please contact Dr. Freudenthal or the [UTEP Office of Disabled Student Services \(DSSO\)](#) if you have a special circumstance such that an accommodation would be helpful in permitting you to excel or demonstrate mastery of the material covered in this course.

## Standards of Conduct and Academic Honesty

- **Standards of Conduct:** Students are expected to conduct themselves in a professional and courteous manner, as prescribed by the Standards of Conduct: [http://hoop.utep.edu/Student\\_Affairs\\_Chapter\\_One-HOP.htm](http://hoop.utep.edu/Student_Affairs_Chapter_One-HOP.htm) Graded work should be unmistakably your own. You may not transcribe or copy a solution taken from another person, book, or other source ( e.g., a web page). Copying other's work will not be tolerated. Professors are required to report academic dishonesty and any other violation of the Standards of Conduct to the Dean of Students.
- **Permitted collaboration:** Students may discuss requirements, background information, test sets, solution strategies, and the output of their programs. However, implementations and documentation must be prepared individually. Students are strongly encouraged to document advice received from others and all other resources utilized in the preparation of their assignments.
- **If academic dishonesty is suspected:** You will receive an incomplete for the lab, and your case will be referred to the Dean of Students for adjudication. The Dean of Students has published a website with complete details concerning the UTEP Academic Honesty policy at the following arcane URL: <http://studentaffairs.utep.edu/Default.aspx?tabid=4386>.

## Course Grading Policy

- **Overall course grading:** It's posted on this web page: [Exams and Grading](#)
- 

## Expectations of UG/Grad Students

Both graduate and undergraduate students will may attend this course. Graduate students are expected to demonstrate a higher level of technical competency, analytical maturity, and communication skills than undergraduates as demonstrated by (1) class participation, (2) exams and (3) lab assignments. Some laboratory assignments will have advanced sections that only graduate students will be required to submit. Finally, the [official course outcomes](#) specify a range of mastery levels of the topics covered in this course that must be demonstrated by students in order to earn high marks. In order to be assigned similarly high marks, MS candidates are expected to demonstrate higher levels of mastery.

## Tools

[Getting Started in Linux/UNIX and SVN](#)

## Linux under virtualization

This is another way to run linux - just pickup a virtualization system such as vmware, configure a virtual machine, and instal linux onto it.

-