CS 288: Intensive Programming in Linux

Honors Section, Fall 2024

Instructor

- Xiaoning Ding
- **E-mail**: xiaoning.ding@njit.edu (**Note**: Before you email me, please review the email policy later in this syllabus)
- **Phone**: (973) 596-3390
- **Office location**: GITC 4203

Course

- **Title**: Intensive Programming in Linux
- **Course number**: CS 288-H01
- **Location**: CKB 222
- **Meeting time**: TR 4:00PM - 5:20 PM. First meeting: Tuesday, September 3, 2024. Last meeting: Tuesday, December 10, 2024.
- **Office hours**: TR 5:20PM - 6:20 PM

Course Description

This course focuses on advanced programming within the Linux environment, where students will use extensively Linux command lines (shell scripts) and C to develop substantial applications. Through hands-on experience, students will enhance their programming skills and deepen their understanding of computer systems, particularly the concepts, designs, utilities, and ecosystems of Unix-like platforms. By the end of the course, students will be equipped with the knowledge and tools to create efficient, scalable real-world applications that fully leverage the capabilities of the Linux OS and its ecosystem. The course also introduces writing programs for parallel and distributed computing environments on Linux, as well as the foundational concepts of parallel computing and distributed computing.

Specific Goals for the Course

- **Master Linux Command Line and Shell Scripting:** Enable students to proficiently navigate the Linux command line and automate tasks using shell scripting, enhancing their efficiency in a Unix-like environment.
- **Develop Proficiency in C Programming and Application Development:** Equip students with advanced skills in C programming within the Linux environment, focusing on writing efficient, scalable code and building substantial, complex applications. Strengthen their problem-solving abilities by tackling real-world challenges and developing solutions for practical scenarios in Linux programming.
- **Understand System Concepts and Designs**: Provide a strong understanding of key system concepts and designs specific to Unix-like platforms, such as memory management, file systems, and permissions.

- **Understand the Ecosystem of Unix-like Systems**: Deepen students' understanding of the broader ecosystem of Unix-like systems, including common utilities, tools, and best practices in Linux programming environment.
- **Introduce Concepts in Parallel, Distributed, and Cloud Computing**: Introduce students to the fundamental concepts of parallel, distributed, and cloud computing, equipping them with the foundational knowledge needed to understand and effectively utilize these models, and preparing them for more advanced topics in these areas.
- **Develop Programs for Parallel, Distributed, and Cloud Computing Environments**: Teach students how to write and optimize programs that run efficiently on Linux systems in parallel, distributed, and cloud computing environments.

Student Outcomes

Students will be able to

- Proficiently navigate and use the Linux command line and shell scripting
- Write efficient and scalable C programs in the Linux environment.
- Develop and deploy substantial applications to solve real-world problems.
- Understand key system concepts like memory management, file systems, and permissions.
- Gain a comprehensive understanding of the Unix-like system ecosystem.
- Understand fundamental concepts of parallel and distributed computing.
- Write and optimize programs for parallel and distributed computing environments.
- Enhance problem-solving skills in Linux programming scenarios.

Prerequisites

- CS 100 Roadmap to Computing
- CS 280 Programming Language Concepts
- The course is about programming in Linux systems, and you will be using intensively Linux command line interface, not GUI (graphic user interface). You need to know how to use Linux systems and common Linux commands to understand course materials and finish assignments.
- The course uses intensively C language. You need to know how to read and write C programs to understand the related course materials and finish assignments.

Textbook

There is no formal textbook for this course. But you will find the following books useful.

- The C Programming Language, Kernighan and Ritchie, Prentice Hall, 2nd ed., ISBN: 978-0131103627
- Matthews, S. J., Newhall, T., & Webb, K. C. (2021). Dive into Systems: A Free, Online Textbook for Introducing Computer Systems. *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 1110-1116).

  Dive Into Systems Different Versions

  Dive Into Systems Development Version (This class will use the development version)

- Supplementary materials will be posted on Canvas course homepage.

Course Work and Grading

Your grade in the course will be determined by the following breakdown:

- **Attendance (12%):** Students are expected to attend every class. By registering for this course, you confirm that there are no time conflicts with other courses or obligations (e.g., work).

- **Practice programming assignments (20%):** The course includes about 10 programming assignments primarily designed for hands-on practice. It's essential to fully understand the problems and develop solutions independently. Additionally, you're encouraged to explore alternative solutions, and consider variations of the problems and their corresponding solutions.
- **Midterm test (34%):** One midterm exam will be scheduled in the semester. Most questions in the midterm exam will be derived from examples and programming assignments.
- **Final exam (34%):** Final exam will be scheduled by the University. Check online for the time and location. Most questions in final exam will be derived from examples and programming assignments.

Grading Scale

Grades	Significance	Percentage Range
A	Superior	85~100%
B+	Excellent	75~84%
B	Very Good	70~74%
C+	Good	65~69%
C	Acceptable	60~64%
D	Minimal	50~59%
F	Inadequate	<50%

Course Navigation

The course is structured into a few topics, including Linux bash scripting, raw data format and organization, radix-sort, state-space searching, multi-threading and MPI parallel processing, and cloud native computing (gRPC, Serverless Computing, and micro-services). Each topic consists of lecture notes, pointers to the related book chapters/sections and required online materials, homework assignments, and pointers to the supplementary materials recommended for interested students. The course will proceed by completing the topics, with each major topic taking about 1~3 weeks and having 1~2 homework assignments. Check the tentative schedule below.

Each week, students will attend two lectures in class. The lectures will mainly discuss the key points on the topics, student questions, and skills and tips needed to solve HW problems. Thus, students are expected to get prepared for the lectures by reading the related book chapters/sections and required materials before the week starts. The lecture discussions will use extensively example programs and code snippets. To get deeper understanding, students are encouraged to understand, change, and test them after each class session.

Track “Announcement” section in Canvas for any updates to the courses.

Tentative Schedule

*Note: Though I will seek to follow this schedule, please understand that I also always reserve the right to modify the schedule, depending upon class progress.