CIS 241 – System Level Programming and Utilities

Winter 2019

School of Computing and Information Systems

Instructor Information

Instructor: Erin Carrier
Office: MAK C-2-211
Office Hours: Tuesdays 5:00pm – 6:00pm (Eberhard 618a – downtown)
Wednesdays 11:00am – 12:00pm (Student Success Center)
Fridays 2:00 – 3:00pm (MAK C-2-211)
Phone: 616-331-2439
Email: erin.carrier@gvsu.edu

Course Description

Introduction to software infrastructure underlying development of computer programs. Topics include Linux utilities, shell scripting, processes, facilities for programming, the C programming language, libraries, structures, pointers, dynamic memory management, and system calls. Course offered winter semester.

Prerequisite: CIS 163 or CIS 260

Credits: 3

Textbooks – None Officially Required

The following textbooks are available for free online through GVSU's library. You may find a combination of them useful:

- Linux Pocket Guide, 3rd edition by Barrett (2016)
- The Linux Command Line, 2nd edition by Shotts (2019)
- Linux in a Nutshell, 6th edition by Love, Figgins, Siever and Robbins (2009)
- Understanding and Using C Pointers by Reese (2013)
- C in a Nutshell by Crawford and Prinz (2005)
- Head First C by Griffiths and Griffiths (2012)
- C Programming Absolute Beginner's Guide, 3rd edition by Miller and Perry (2013)

Course Objectives

After successful completion of the course a student should be able to:

- **Application/Use:** Use command-line interface to do basic system operations such as create a directory, modify file permissions, create a link, manage processes, and access a computer remotely.
- **Mechanism/Manipulate:** Manipulate basic features of a command-line environment such as the 'PATH', which specifies the locations of executable programs, and the 'CLASSPATH', which specifies the locations of Java classes.
- **Synthesis/Write:** Write basic meta-programs in a shell or scripting language, e.g. to compile statistics for numbers stored in text files.

- **Application/Use:** Use build-system utilities such as 'make'/'cmake', 'gdb', or 'SCons' to edit, compile, test, time, debug, and profile a C program.
- **Synthesis/Construct:** Construct programs that access data by using the range of pointer operations and manual dynamic memory management to implement basic ordered binary trees a.k.a. binary search trees (BSTs).

Grade Breakdown

Activity	Percentage
Midterm Exam	25%
Final Exam	25%
Programming Projects	30%
Mini-labs / HW	20%

Grading Scale

Grade	Percent Range
A	[93, 100]
A-	[90, 93)
B+	[87, 90)
B	[83, 87)
B-	[80, 83)
C+	[77, 80)
C	[73, 77)
C-	[70, 73)
D+	[67, 70)
D	[60, 67)
F	[0, 60)

Need Help?

If you have questions, please ask! Either during class, in office hours, or via Piazza, I am here to help. Feel free to stop by office hours with non-class related questions, such as questions about graduate school – I'm happy to chat!

Piazza

We will be using Piazza for communication. Important announcements will be posted on Piazza and it is your responsibility to monitor Piazza for these announcements. If you have a question or a concern, please post it on Piazza. Most questions can be posted publicly. However, if you are asking a question about your specific grades or some other private matter, you may post the question/note privately. Do not post solutions (answers, code etc.) publicly on Piazza.

Please post on Piazza in place of emailing the course staff. This is both to assist other students who may have similar questions and to ensure you receive the fastest response as emails are far more likely to be buried in an inbox with hundreds of other emails.

While Piazza is a fantastic resource, it is typically not appropriate for helping students debug code, which requires a more back and forth conversation. If you need assistance debugging your code, please come to office hours.

Course Policies

- All assignments and assessments, unless otherwise specified by the instructor, are to be completed individually.
- An assignment is considered **late** if you turn it in past the posted due date and time. Late assignments will not be accepted.
- Assignments requiring submission will be submitted through Github Classroom and PrairieLearn. More details will be provided with the first assignment; however, you should create a github account ASAP if you do not already have one. Some assignments (tentatively mini-labs) may not require submission and will instead be checked during class.
- All code submitted should be properly commented and any code submitted that does not compile will receive a 0.
- Any questions, concerns, or misgivings regarding a specific grade must be raised within one week of the release of the grade. After one week has passed, the assigned grade is considered accepted and no further complaints will be considered.
- The instructor reserves the right to modify course policies, the course calendar, and due dates.

Mini-Labs

In the latter portion of many/most class periods, short activities will be assigned and time provided to complete them. These "mini-labs" are designed to allow for a chance to apply the concepts discussed during lecture in an environment where help is easily available. Once you have completed the mini-lab, please let the instructor know and they will verify and mark your completion. Mini-labs are meant to be completed in-class and makeups will not be allowed for students who miss class on the day of the mini-lab. However, the 4 lowest mini-lab scores will be dropped, which should be reserved for unexpected absences such as sickness or family emergencies. For some minilabs, you may be asked to walkthrough your attempt with the class.

Academic Honesty

All students are expected to adhere to the academic honesty standards set forth by Grand Valley State University. In addition, students in this course are expected to adhere to the academic honesty guidelines as set forth by the School of Computing and Information Systems, the details of which can be found at https://www.cis.gvsu.edu/academic-honesty/.

All assignments are to be completed individually unless explicitly stated otherwise by the instructor. While, learning from others is encouraged, the work you submit must be your own. You may *discuss* your approach (but not your code) with your peers. However, you must have typed/written every part of your assignment yourself. In some cases, code/pieces of work from elsewhere *may* be allowed -- if so, this will be stated in the instructions. In that case, you *must* acknowledge the source.

Special Needs?

If there is any student in this class who has special needs because of a disability, please contact Disability Support Resources (DSR, https://www.gvsu.edu/dsr/) at 616-331-2490.

GVSU Course Policies

This course is subject to the GVSU policies listed at http://www.gvsu.edu/coursepolicies/.

In Case of Emergency

Fire: Immediately proceed to the nearest exit during a fire alarm. Do not use elevators. More information is available on the University's Emergency website located at http://www.gvsu.edu/emergency.

Tentative Schedule

(This is *very likely* to change throughout the term.)

Week	Topic
1 Jan 6	Syllabus, Overview, Linux basics: ssh, directory structure, …
2 Jan 13	Linux basics: manpages, path, processes, …
3 Jan 20	**MLK Day (no class on Mon 01/20)** Version control
4 Jan 27	More Linux: file permissions, linking, piping, redirection
5 Feb 3	Regular expressions
6 Feb 10	sed, awk, grep
7 Feb 17	Scripting (Bash, Python)
8 Feb 24	Introduction to C C basics (basic data types, functions, control flow)
9 Mar 2	**Spring Break (no class)**
10 Mar 9	**Midterm Exam on Monday (03/09)** Memory management, Pointers, Arrays/Matrices, Strings
11 Mar 16	Revisiting functions with pointers, reading from stdin
12 Mar 23	Strings, 2d Arrays
13 Mar 30	Debuggers / Valgrind, Structs/Headers
14 Apr 6	Makefiles, IO, Bitwise operators
15 Apr 13	Libraries, Structs, Unions, Arguments
Finals Week	**Final Exam: Monday, April 20** **12:00pm – 1:50pm**