CSE220 Programming for Computer Engineering

Fall 2012—Course Information

Lecture Section: 71233
Date and Time: MWF 10:30–11:20
Room: COOR L1-10

Catalog Course Description

Introduces procedural and object-oriented programming languages (C/C++) and Linux/Unix software development. Fee. Prerequisites: A grade of C or better in CSE120/EE120 Digital Design Fundamentals *and* a grade of C or better in CSE205 Concepts of Computer Science and Data Structures. Three (3) credit hours. Lecture/No lab.

Expanded Course Description

This is a required course for the B.S.E. in the Computer Systems Engineering program. The C and C++ programming languages are used extensively in industrial computer engineering and a proper understanding of these languages is essential for a graduate in computer systems engineering. Furthermore, the material learned in this course will be required in several of your future CSE courses (most specifically in CSE325 Embedded Microprocessor Systems, CSE310 Data Structures and Algorithms, CSE430 Operating Systems, and CSE423/424 Systems Capstone Project I and II) where the instructors will assume you can write complete, functional programs in C and C++. Furthermore, The GNU/Linux and Unix operating systems are used extensively by professional engineers and scientists so at the least, gaining a rudimentary understanding of this operating system is an important part of your undergraduate education.

Course Objectives

1. To be able to effectively use the GNU/Linux and Unix operating systems at a basic level:
    - Log in to a console window and use basic Bash shell commands.
    - Create, delete, and move files using Bash shell commands.
    - Understand file and directory permissions and use the **chmod** command to set permissions.
    - Write and use relatively simple Bash shell scripts.
    - Use I/O redirection and pipes.
    - Use process management commands to start, pause, and kill jobs. Move jobs between foreground and background.
    - Use the **tar**, **gzip**, and **bzip2** programs to create file archives.
    - Edit a C/C++ program using a GNU/Linux text editor.
    - Use the GNU tool chain to compile, execute, and debug a C/C++ program.
    - Use **make** and makefiles to automate software builds.
    - Setup a repository for program development.
2. To develop an intermediate understanding of the C programming language:
    - Use various data types including the fundamental data types, structs, 1D- and 2D-arrays, and pointers.
    - Master flow-of-control constructs: **while/for** loops, **if** statements, **switch** statements, nested loops, and nested **if** statements.
    - Write programs using multiple functions and multiple source code files.

- Understand local, global, and multi-source code file scope rules.
- Understand and use the **static** and **extern** reserved words.
- Understand the difference between declaration and definition of variables and functions.
- Master dynamic memory allocation and pointer variable usage.
- Effectively use structs.
- Master text and binary file I/O.
- Understand and use basic concurrency constructs such as fork/join.

3. To develop an intermediate understanding of the C++ programming language:
    - Understand differences and similarities among C, C++, and Java.
    - Use functions and classes from the C++ Standard Libary.
    - Design and code a program using proper object-oriented constructs.
    - Master text and binary file I/O using streams and manipulators.
    - Effectively use dynamic memory allocation, including proper use of destructors.
    - Write multi-class programs involving multiple objects.
    - Write methods, and properly use pass-by-value and pass-by-reference parameter passing techniques.
    - Write classes using single inheritance.
    - Understand polymorphism and how and when to use it.
    - Write generic code using function and class templates.

Major Topics

The material to be covered in this course includes:

1. GNU/Linux and Unix
    - Brief history of Unix and GNU/Linux.
    - GNU/Linux distributions. Cygwin.
    - Obtaining an account on **general.asu.edu**.
    - Using general.asu.edu and X Windows.
    - FTP, SFTP, and transferring files with WinSCP/Filezilla.
    - Shells. Brief history. Bash shell. Commands and options.
    - Directory structure and common GNU/Linux directories.
    - Pathnames. Absolute pathnames. Relative pathnames.
    - Common file and directory commands.
    - Permission attributes and setting file/directory permissions.
    - Filename globbing.
    - Creating file archives with **tar**, **gzip**, and **bzip2**.
    - Help system. Man pages. **apropos** command. Info.
    - I/O redirection. **stdin** and **stdout**. Pipes.
    - Processes and process management commands. Starting, pausing, killing jobs. Foreground/background jobs.
    - Basic Bash shell scripts.
    - Introduction to the GNU GCC tool chain. Using **gcc/g++** to compile C/C++ programs.
    - Automating software builds with **make** and makefiles.
    - Debugging with **gdb**.
    - Introduction to subversion, version control.
2. C Programming
    - A first C program: "Hello world."

- Header files. Preprocessor. **#include** and **#define** directives.
- The C Standard Library. Common header files: **ctype**, **math**, **stdio**, **stdlib**, **string**, **time**
- Fundamental data types. Defining variables and constants. 1D- and 2D-arrays.
- Input/output with *printf()*, *scanf()*, *putc()*, *getc()*, *fgets()*, and *fputs()*.
- Arithmetic, relational, logical, bitwise, assignment, and conditional operators.
- Expressions. Precedence rules. Typecasting.
- Flow-of-control constructs: **if** statement, **for/while/do-while** loops, and **switch** statement.
- Defining functions. Passing parameters by value. Returning values from functions. Declaring functions.
- Local and global variables. Static reserved word.
- Recursive functions.
- Multi-source code file programs. The **extern** reserved word. Scope rules.
- Memory addresses and pointer variables.
- Pointers and function arguments. Pointers and arrays. Pointer arithmetic. Pointer arrays. Pointer to pointers.
- Multidimensional arrays.
- Command line arguments.
- Pointers to functions.
- Structs. Arrays of structs. Pointers to structs.
- Self-referential structs.
- Typedef reserved word.
- Text and binary file I/O.
- Concurrent programming. *fork()* and *join()* library functions.

3. C++ Programming
    - Comparison of C++ to C and Java.
    - The C++ Standard Library. Common header files: **cmath**, **fstream**, **iomanip**, **iostream**, **string**.
    - Classes and objects.
    - Declaring and implementing a class. Scope rules.
    - Input with **cin** and output with **cout**. Stream manipulators. Text file I/O.
    - Defining and creating objects.
    - Calling constructors. Member initializer lists.
    - Dynamic memory allocation/deallocation with **new** and **delete**. Destructors.
    - **const** reserved word. Proper use of const.
    - Calling functions. Pass-by-value. Reference variables. Pass-by-reference.
    - Function overloading. Operator overloading.
    - Single inheritance.
    - Polymorphism. Virtual functions. Abstract classes.
    - Binary file I/O.
    - Function and class templates.

Office Hours & Support

The instructor will be available during his office hours to answer any questions you may have about the course, the material, or the lab projects. I'm a friendly guy (except when I'm not), and willing to talk with you and help you as much as I can, so stop by if you want, but please be aware that I also teach other courses and have other duties as well, so I am quite busy. When

you come in for help, please be prepared with a printout of your program or project solution ready for discussion, bring your book, and be ready to ask pointed and specific questions about what you do not understand or are having difficulty with. It is next to impossible for me to help you if you come in and say, "I don't understand anything." Also, a final word of advice. I may not be available as much as you would like the day before or the day a project is due. I will give you plenty of time to complete the projects as long as you start working on them well before the due date. This will give you ample time to meet with me to discuss your questions as you are working on the project. Start early!

The School of Computing, Informatics, and Decision Systems Engineering (SCIDSE) provides free tutoring in the BYENG 214 computer lab, on the second floor of the Brickyard engineering building (the tall one; the two-story classroom building is BYAC). The tutors will be located in the 214AD/AE cubicles. Note that this lab is open 24/7 but you must swipe your ASU Suncard to gain entry. *Note: these tutors are hired primarily for CSE100 (C++), CSE110 (Java), and CSE205 (Java). They may or may not know very much about programming in C, and their C++ programming knowledge is probably less than their Java knowledge. But, you may still find one or more of them to be helpful, and I strongly recommend you visit with them when you need help.* Tutoring Schedule (note: schedule may change, check often).

Finally, the Fulton Schools of Engineering provides free tutors for CSE100, CSE110, CSE205, and CSE240 in the Engineering Tutoring Center, ECF 100. Their tutoring schedule does not list CSE220 as being one of the courses for which tutoring is available. However, you may find that one of the tutors for CSE205 and/or CSE240 has sufficient C/C++ knowledge to be able to assist you, although I cannot guarantee that. Tutoring Schedule (note: schedule may change, check often).

Course Materials & Resources

- *Linux and UNIX Programming Tools* by S. Sarwar and K. Al-Saqabi, © Addison Wesley 2002. ISBN-10: 0201773457. ISBN-13: 978-0201773453. (Moderately Recommended)
- *C How to Program, 6th Edition*, by P. Deitel and H. Deitel, © Prentice-Hall, 2009. ISBN-10: 0136123562. ISBN-13: 978-0136123569. (Required)
- *C++ How to Program, 7th Edition*, by P. Deitel and H. Deitel, © Prentice-Hall, 2009. ISBN-10: 0136117260. ISBN-13: 978-0136117261. (Very recommended)
- My ASU Courses (Blackboard)
- See the Tools section below.

Assessment

Various methods will be used to present the material and assess the student's understanding and comprehension.

Lab Projects

There will be several lab projects (probably four to six) covering Linux, and C/C++ programming. The lab projects will be weighted at 40% of your final course percentage. Projects are accepted for grading on a variable deadline:

Submitted	Bonus/Penalty
> 48 hrs before the deadline	Bonus of 20% of earned pts.
> 24 hrs and ≤ 48 hrs before the deadline	Bonus of 10% of earned pts.
≤ 24 hrs before the deadline	Accepted for grading with no bonus/no penalty.
< 24 hrs after the deadline	Penalty of 10% of assignment pts.
≥ 24 hrs and < 48 hrs after the deadline	Penalty of 20% of assignment pts.
≥ 48 hrs after the deadline	Not accepted for grading. Score will be 0 pts.

Projects that are submitted more than 24 hours before the deadline will be awarded bonus points. For example, if the project is worth 50 points and you earn 33 points by submitting the project more than 48 hours before the deadline, then you will be awarded 33 x 20% = 6.6 bonus points, for a total project score of 40 points. Projects that are submitted less than 48 hrs after the deadline will be penalized points. For example, if the project is worth 50 points, and you earn 33 points by submitting your project one minute after the deadline, then you will be penalized 50 x 10% = 5 points, for a total project score of 28 points. In no case will a project be graded when submitted 48 hours or more after the deadline. Your **lab project percentage** *lab* will be calculated as the number of project points you earned (including any bonus points) divided by the number of lab project points that were possible. Your project percentage— including bonus points—will be limited to a maximum of 100%.

Examinations

There will be **two in-class midterm examinations**; the exam dates are listed on the monthly calendars in the Schedule section. No exam scores will be dropped, and only in rare circumstances (documented illness, death or sudden illness in the family, incarceration, official school trip, job travel) will a makeup exam be given. If you know that you are going to be absent on the date of an exam, for a good reason, then you must arrange to take the exam *before* the date of the exam. Each of the midterm exams will be weighted at 20% toward your final course percentage. There will be a **final exam** given during the final exam period, weighted also at 20%. **Note: you will be required to show proper photo identification (a driver's license, military ID card, or ASU Suncard) when handing in your examination for grading**. If the instructor does not recognize you, and you cannot produce photo identification, your exam will not be accepted for grading.

Calculating Final Letter Grades

Your final letter grade will be based on your **final course percentage** *FCP* which is calculated as a **weighted sum** of your scores on lab projects and examinations.

$FCP = (lab \times 40\%) + (exam_1 \times 20\%) + (exam_2 \times 20\%) + (final \times 20\%)$

The **final letter grade** will be based on *FCP*:

FCP	**Letter Grade**
$87.5\% \leq FCP \leq 100\%$	A
$75\% \leq FCP < 87.5\%$	B
$62.5\% \leq FCP < 75\%$	C
$50\% \leq FCP < 62.5\%$	D
$FCP < 50\%$	E

The instructor may round course percentages up at his discretion at the end of the term after all student percentages have been determined (e.g., a 74.6% C may be rounded up to a 75% B, but then again, it might not. The only guarantee is that percentages will not be lowered).

Grade Appeals

If you believe a graded assessment (e.g., lab project or exam) was marked incorrectly, then you may file a **grade appeal** using this form. Please read the form instructions carefully. Basically, you have **one week** from the date the assessment was returned to file your grade appeal. No appeals will be accepted after the one week period for **any reason**. You must submit the **original** lab or exam, along with your grade appeal form **in person** to the instructor; emailed forms will be discarded. I will regrade your assessment, and will return the original to you along with the completed grade appeal form. If your score was changed, the new score will be entered in the Blackboard Grade Center.

Academic Misconduct

In general, the instructor believes learning is a **collaborative activity**, that students learn best when they work together, and that students should be **encouraged** to learn from and teach each other. These activities would include discussing solutions to homework exercises, working together on lab projects, and jointly studying for exams. In completing homework assignments and labs, student collaboration is encouraged and will be permitted as long as each member of the team **contributes equally** to the work. Collaboration is only acceptable between members of the same team; **inter-team collaboration** is forbidden and violators may be sanctioned. Collaboration on quizzes and examinations is **not permitted**; each quiz or exam must be completed by the individual student. Failure to abide by these rules will result in a score of zero being assigned to one or more members of the team (i.e., if I have a reasonable hunch that one student did all of the work on an assignment and the other student(s) simply put his/her name on it, then the student who did all of the work will receive the assignment score and the othes student(s) will be given a score of zero).

The ASU Academic Integrity Policy will be enforced. I suggest you acquaint yourself with this policy to avoid any intentional or unintentional violations. Note that the allowable sanctions available to the instructor include: a reduced or failing grade on the assignment, or a reduced or failing grade for the course. Violators may be referred to the School of Engineering Dean's office, and the most severe penalty may be expulsion from the University. Any incidents will be handled on an individual case-by-case basis, and different sanctions may be imposed for different reasons for different offenders.

Classroom Behavior

The ASU Student Services Manual (SSM 201-10) permits the instructor to withdraw a student from a course for disruptive behavior with a mark of **W** or **E**. Note that "disruptive behavior" is defined **by the instructor**, not by the University or the student.

In my courses, appropriate classroom behavior is expected and required. Inappropriate behavior includes, but is not limited to: talking without being given permission to do so; verbally or sexually harassing another student or the instructor; being disrespectful or rude toward another student or the instructor; complaining/whining about assignments, grades, course policies, etc.; using a computer or other electronic device for anything other than an assigned task; using electronic portable stereos or entertainment devices, disruptive pagers or cell phones, using any tobacco or alcoholic product, or being under the influence of alcohol or illicit drugs or

nonprescribed prescription drugs. This list is not inclusive and any other behavior that is deemed inappropriate by the instructor is also forbidden.

Violation of any of these unacceptable behaviors will result in the offender being removed from the classroom and notification of the offense to the Fulton Schools of Engineering's Dean's Office. A warning may or may not be provided.

Note that in general, you may **sit** where you wish. However, the instructor has **the right** to ask you to sit in specific seat or move to a different seat at any time during the semester. In the past, I have moved students who I suspected were **cheating** on a quiz or exam, and I will do so in this course if I believe you are looking at another student's paper during an exam.

Attendance Policy

There is a strong and well-established correlation between class attendance, learning, and performance; therefore, regular class/lab attendance and participation is expected. I (or in the labs, the TA) intend to begin class each day on time, and we expect you to be present with your book open and notes ready at that time. However, you are adults, and you (or someone who may or may not love you) are paying for your education, and ultimately, it is **your** education. If you want to squander this opportunity, then no gimmick we devise to try to get you to come to class and participate will be successful, so attendance will not count toward your final course grade.

Requirements for Success in this Course

The instructor assumes that you are mature and responsible adults, that you are enrolled in this course because you wish to learn the material, that you will read any assigned readings before class begins, that you will come to class prepared to discuss the reading and ask questions, that you will complete the assignments to the best of your ability on time, that you will actively participate in class discussions, and that you will ask questions about material you find confusing. The instructor believes that college students must be actively involved in their own learning process, that they cannot just sit and listen to lectures and expect to learn the material, that one of the purposes of college education and the Arizona State University mission is for the student to self-develop skills such as problem solving, independent learning, critical thinking, and effective written and spoken communication. To succeed in this course you must:

- Be prepared for every class, attend every class, and pay attention.
- Read the textbook and other assigned readings prior to class.
- Begin and complete the assignments/labs well before the due date.
- Prepare thoroughly for and complete every exam.
- Do any additional exercises you must to understand the material.
- Ask questions in class. If you do not feel comfortable asking the question in class, talk with me outside of class.
- If you do not complete an assignment/lab by the deadline, complete it anyway later.
- If you miss points on an assessment, determine why your work was graded incorrect and learn how to do it correctly.
- Seek help from the instructor, TA, or the tutoring center before you are too far behind on your understanding of the subject.
- Read your email every day; I often send important announcements via email.
- Check the course website and Blackboard every day for new announcements, material, and updates.

Having said all that, I want you to know that I **care** about all of my students and their education. I want all of you to **succeed**, to feel you have gained something from the course, to have some **fun** in the process, and I will do all I reasonably can to **assist** you in your efforts!

Statement on Accommodations

The Disability Resource Center (480-965-1234; Matthews Center; email: disability-q@asu.edu) is the central location for students requiring accommodation. Any student requiring accommodation must contact and register with the Center before any accommodation requests can be granted by the instructor. If you require accommodation, please contact the Center as soon as possible so the instructor can work with you to ensure your success.