UNIVERSITY OF South Carolina

CSCE 240: Advanced Programming Techniques

CRN: 40673, Section: 002

instructor: Biplav Srivastava, Ph.D.
email: biplav.s@sc.edu
office: AI Institute, Room 515, 1112 Greene St., Columbia, 29028
office hours: By Appointment in-person or Blackboard (11:30 am - 12:30 pm), M and W

teaching assistant: Yuxiang (Cherry) Sun,
office: Room 1211, Storey Innovation Center, 550 Assembly Street
office hours: In-person or Blackboard (10:30 am - 11:30 am), Tu and Th

Course Details

Section: 002

Meeting Time: TuTh 8:30—9:45 PM

Meeting Location: In person: INNOVA 1400, Virtual: Blackboard Collaborate Ultra

Credit Hours: 3

Bulletin Description: Pointers; memory management; advanced programming language structures: operator overloading, iterators, multiple inheritance, polymorphism, templates, virtual functions; Unix programming environment.

Prerequisite(s): Grade of D or better in CSCE 215, grade of C or better in CSCE 146.

Learning Objectives

1. Develop language-independent understanding of programming concepts by being exposed to multiple languages (C++, Java, Python)
2. Independently design and implement programs of language choice (C++, Java or Python based on choice) in a Unix environment
3. Demonstrate mastery of pointers, iterators, memory management including object creation and destruction, and parameter passing in C++
4. Demonstrate mastery of object-oriented programming concepts including: inheritance, polymorphism, operator overloading, template functions and classes, and the use of STL containers.
5. Develop object-oriented models using UML
6. Able to work in programming teams with code review and walk throughs

7. Solve practical problems that matter by designing individual programs to fit together for a larger societal project goal.

All learning outcomes of the remote section are equivalent to face-to-face (F2F)section outcomes.

Text

C/C++

- (Authoritative) Brian Kerninghan and Dennis Ritchie, The C Programming Language, https://en.wikipedia.org/wiki/The_C_Programming_Language
- (Authoritative) Bjarne Stroustrup The Annotated C++ manual, https://www.stroustrup.com/arm.html
- The C++ Programming Language (4th Edition), Addison-Wesley ISBN 978-0321563842. May 2013, https://www.stroustrup.com/C++.html
- Walter Savitch, Absolute C++ 6th ed., Pearson, 2016
- Free books
  - C++ Essentials, Sharam Hekmat, https://freecomputerbooks.com/Cpp-Essentials.html
  - Fundamentals of C++ Programming , by Richard L. Halterman, https://archive.org/details/2018Fund
  - C++ Today, https://www.jetbrains.com/cpp/cpp-today-oreilly/

Java

- (Authoritative) The Java Programming Language, 4th Edition 4th Edition by Ken Arnold, James Gosling, David Holmes, ISBN-13: 978-0321349804
- Effective Java - 3rd Edition, by Joshua Bloch, ISBN-13: 978-0134685991
- Free books
  - Essential Java by Krzysztof Kowalczyk (HTML), https://www.programming-books.io/essential/java/
  - Teach Yourself Java in 21 days, https://cs.cmu.edu/afs/cs.cmu.edu/user/gchen/www/download/jav

Python

- (Authoritative) https://docs.python.org/3/tutorial/
- Fundamentals of Python Programming, Richard L. Halterman, https://freecomputerbooks.com/Fundamen of-Python-Programming-by-Richard-Halterman.html
- Think Python, Allen Downey, https://greenteapress.com/wp/think-python-2e/

All reading/materials comply with fair/use policies.

Course Overview

This course will be partially asynchronous with lecture material posted prior to a live session with the Instructor.

Student-to-Instructor Interaction Students will attend in-person classes on Tuesday–Thursday. Due to uncertainty around COVID and individual health situation, the instructor will allow students to view online lectures in lieu of in-class presence with prior intimation. The lectures will be recorded so that students can refer to them later. The in-class session allows opportunities for student-to-student interaction, as well as questions and further explanation of lecture topics. Though attendance is not strictly required and lectures will be recorded, students missing more than one each week will be penalized in their attendance grade. If a student is unable to attend the live session s/he should contact me on those days (at least 2/week) to ensure s/he is keeping up with course content.

Student-to-Content Interaction Students will complete readings, programming assignments, and projects.

Students should expect instructor responses to all feedback within a reasonable time (one business day). The same reasonable response time is expected of students. Students can expect feedback on assignments within two business days of the final due date.

Technology

This course is taught using Unix and open source software. All software required by the course is free for download. Students are expected to be familiar with the Unix operating system (familiarity is a prerequisite for this course via CSCE 215) and help cannot be offered with other Operating Systems or software, i.e. Mac OS, Windows, Visual Studio, Eclipse, etc.

Topics Covered[1]

1. Unix Programming Environment: Unix tools, C preprocessor, Make, Shell, I/O redirection, debugging.
2. Hello World in multiple languages: C++, Java, Python.
3. Unit and System Testing.
4. Input and output Management
5. Pointers: Pointer manipulation, functions and function pointers, virtual functions.
6. Basic class management: constructors, destructors, data hiding, container classes.
7. Memory management: object creation and destruction, memory leaks.
8. Advanced C++ features: operator overloading, iteration, special containers, inheritance, code reuse, multiple inheritance, virtual functions, polymorphism, templates, template libraries.

[1] This list may be extended or abridged at my discretion.

9. Introduction to UML and object oriented modeling: usecase models, object identification, specifying static behavior, activity diagrams, collaboration diagrams and sequence diagrams, specifying relationships: generalization/specialization, aggregation, associations including multiplicity and roles, dynamic behavior using state diagrams.
10. Introduction to Source Control and Distributed Source Control, for example, using git.
11. Software development in teams: code walkthrough (peer review), independent testing
12. Project management to solve real-world problems

Assignments

Project: The class will have independent programming homeworks. Once completed, the homeworks can be assembled to finish a *chatbot* to help people in a domain.

Specifically, the project will be involve having a chatbot that can answer questions about a South Carolina member of state legislature from:

https://www.scstatehouse.gov/member.php?chamber=H

Each student will choose a district (from 122 available). Programming assignment programs will: (1) extract data from the district, (2) process it, (3) make content available in a command-line interface, (4) handle any user query and (5) report on interaction statistics. The project will test the student's ability to use the knowledge and skills s/he has learned in the lectures and homework assignments. This will also lead to a tangible software that solve a real need.

For further information on project expectations, see the Program Expectations below.

Programming Assignments: There are five programming assignments in this course to support the project as described above. Students are encouraged to ask the instructor for guidance and to seek help in online information repositories. Programming assignments must be completed by the student submitting the assignment without programmatic help from individuals other than the instructor. Students will be given a reasonable amount of time to complete assignments for full credit.

For further information on homework expectations, see the Program Expectations below.

Home Work: Homework assignments will be testing content taught in class. They will be peer-reviewed in class. Not graded but class activity (doing home assignments, peer reviewing and testing) will count towards overall grade.

For further information on homework expectations, see the Program Expectations below.

Program Expectations: The following expectations hold for both Homework and Project submissions.

- Programs must run correctly on one of the reference Linux machines in the SWGN 1D43 or SWGN 3D22 computer labs. Your program running on your personal com-

puter is not considered valid. It must run on an official computer chosen by the "client" (the instructors or peer). Note, these are the same machines used in the prerequisite course.[2]

- Programs that do not compile will not receive credit.
- Programs that crash during execution will not receive credit.
- Programs that go into infinite loops during execution will receive zero credit. Programs will be given 10s execution time for this metric, as I have not solved the halting dilemma.
- Programs that fail to have your name and copyright information in the header documentation of ALL files (header and implementation code) will receive zero credit.
- Programs are expected to be able to process any input file that meets the same format description as the sample data provided to you. For grading purposes your program will be run on different data with the same format.
- It is not enough for your programs to compile and pass correctness tests, they must also be styled correctly and documented. We will be following the Google Style guidelines found at:
  https://google.github.io/styleguide/cppguide.html.
  You may download the Google's free cpplint Python3 application to ensure that you are correctly following the style guidelines.
- Submissions will be graded roughly as follows: 80% for correct execution, 20% for correct organization, style, and documentation. It is also true that we tend to be a little more lenient on organization, style, and documentation for the first homework exercise than on later homework so as not to penalize you excessively for an initial misunderstanding of our standards.
- All submissions from all sections will be submitted to the MOSS website at Stanford for plagiarism detection purposes.
- Grading corrections must be brought to my attention within two (2) business days of the returned grade/results.

Grading

Participation:	10%
Homework:	10%
Prog. 1-5	50%
Project Ass.:	10%
Quiz and Exams:	20%

A	≥ 90.0	C	≥ 70.0
B+	≥ 85.0	D+	≥ 65.0
B	≥ 80.0	D	≥ 60.0
C+	≥ 75.0	F	< 60.0

Course Policies

Blackboard: This course will extensively make use of blackboard.cse.sc.edu. You should check regularly and will be held responsible for anything posted on the site.

[2] If you are unable to remotely connect to these machines, you should resolve that as soon as possible.

Attendance: Though live session attendance is not required, you should make regular check-ins. Students regularly missing live sessions will be contacted. Students not replying to said contact will be reported to the college's Advising office as a *student of concern*.

Academic Integrity: The faculty treats cheating and plagiarism seriously and all instances will be reported to the Office of Academic Integrity. All submissions in this course are expected to be a student's own work. This course upholds all tenets of the USC Honor Code. You should look over that document if you intend to work with or ask others for help. A note, it takes more than one person to share work. Any student found sharing her or his work is engaged complicity as defined by the Honor Code and is as guilty of violation as a student copying said work.

Any project or homework assignment found to be in violation will be awarded -100% of its points. It is better to take a zero than to cheat.

Any student found guilty of uploading instructor-created material to a website such as Chegg.com will receive a failing grade for the course. This is a copyright violation as well as a blatant disregard for the Carolinian Creed.

Disability Services: "Every student deserves equal access to all aspects of the University of South Carolina experience," (Student Disability Resource Center). This course supports students requiring additional accommodations. To that end, you must ensure you are registered with SDRC prior to benefiting from any of the aforementioned accommodations. You must comply with their time requirements.

Performance-Based Grading: Grades will be awarded based on the evaluation of student performance. Grades will not be assigned or adjusted based on any other criteria. If a student chooses to request a grade change for any reason other than her or his performance, that request will be passed along to the correct party for evaluation.

Intended Purpose: The student's work should reasonably match the instructor's intended purpose. Efforts to deceive, violate, circumvent, or avoid the purpose of a question will result in zero points awarded. See the USC Honor Code for further clarification.