Computer Science Department

COMP2040-202 Computing IV

3 contact hours - 3 credits

Fall 2018 Syllabus

General Information

Instructor	Dr. Charles T. (Tom) Wilkes
Office	Dandeneau Hall 323
Phone	978-934-3634
Class Time	MWF 2:00-2:50 PM
Class Location	Southwick Hall 401
Office Hours	MWF 9:00-9:45 AM, 11:00-noon, 3:00-5:00 PM (except Wednesday 9/12, 10/3, 11/7: 3-5 PM hours canceled)
Teaching Assistant	TBD

Required Textbook

- Stanley Lippman, Josee Lajoie, and Barbara Moo, *C++ Primer*, 5th Edition (Addison-Wesley, 2013). ISBN 978-0321714114

Supplementary Textbook / Materials

- Scott Myers, *Effective Modern C++*, 1st Edition (O'Reilly Media, 2014). ISBN 978-1491903995

Course Description

- Advanced C++ programming, which deepens students' understanding of object-oriented analysis and design. Basic software engineering principles and practice, including work with APIs. Topics may include program translation, web software, parsing, and regular expressions.

Course Prerequisites

- COMP2010 Computing III

Course Category

- Required

Additional Course Information

- **Course Overview**
    - We will be writing lots of code
    - We will be doing it in C++
    - We will be using some interesting APIs, including SFML, a free, open-source "simple fast media library" for C++, which is available for Mac, Win, and Linux (translation: it's a gaming library)
    - We'll be working through some awesome problem sets developed over the last 15 years at Princeton, led by Robert Sedgewick.
- **Web Page** – on Blackboard
- **Software** – GCC (g++) and Make on Linux, or Xcode on Mac OS (see assignment PS0)

Additional Faculty Details

- **Office Location** – Dandeneau Hall 323
- **Email** – Charles_Wilkes@uml.edu

Learning Outcomes:

- Employ appropriate object-oriented (OO) techniques in C++ in the development of 500+ line programs
- Analyze technical specifications for a variety of algorithms, and create working code based on specs provided
- Use the Unix shell for C/C++ code development, including use of the gcc compiler and linker tools, and Makefiles
- Use the C++ SFML (simple fast media library) for event handling, graphics, animation, and sound
- Employ unit testing in your software development process
- Use industry-standard C++ API libraries, including Boost (e.g. unit testing, regular expression, and date/time libraries)
- Describe the value (and limitations) of coding standards, and use a static style checker to review your own code
- Document your work for technical written presentations

Methodology

Teaching methods: This course will make use of discussions, examples, and question and answer sessions. Hands-on programming activities using computers during class, in addition to active reading, experimenting with sample programs, and problem solving, will be used during the semester. Independent programming assignments are used to evaluate language and coding skills.

Assessment: There will be in-class exercises, regular programming practice problems and quizzes, programming exercises, and a final portfolio submission in lieu of a final exam.

Grading:

The available points for the various assignments in this class are:

Weekly Homework Assignments	60%
Quiz Average	20%
Final Portfolio	15%
Classroom Participation	5%

The final grade is assigned using the following scale:

A	90.0-100
B+	87.0-89.9
B	83.0-86.9
B-	80.0-82.9
C+	77.0-79.9
C	73.0-76.9

C-	70.0-72.9
D+	67.0-69.9
D	60.0-66.9
F	below 60

At the end of the semester, a course average is computed for each student using the percentage weights given above. The scale for assigning letter grades based on the course average is formed by averaging the grading scales of the assignments, quizzes, portfolio, and participation using the weights given above. Factors such as attendance and improvement may be considered when making the final decision in borderline cases.

Requests for re-grading assignments may be made up to one week after the assignment is returned. The request must be submitted in writing and include a short paragraph outlining the rationale for the re-grade. Acceptable requests include correcting errors in calculating a score, marking a correct answer incorrect, etc.

University Policies and Advising Resources

Students are expected to be familiar with the following policies from the Undergraduate Catalog:

- Academic Policies: https://www.uml.edu/Catalog/Undergraduate/Policies/Academic-Policies/Academic-Policies.aspx
- Kennedy College of Sciences Policies: https://www.uml.edu/Catalog/Undergraduate/Sciences/policy/default.aspx
- Computer Science Department Policies: https://www.uml.edu/Catalog/Undergraduate/Sciences/policy/Continuance-appeal-dismissal.aspx

Also, students are urged to be aware of advising resources provided by the University:

- https://www.uml.edu/Academics/Provost-office/Faculty-success/Professional-Engagement/advising-resources.aspx

Course Policies

Attendance: Formal attendance is mandatory. Three unexcused absences are permitted. Additional absences may result in a failing grade and/or removal from the course.

Academic integrity: The practice of good ethical behavior is essential for maintaining good order in the classroom, providing an enriching learning experience for students, and as training as a practicing computing professional upon graduation. This practice is manifested in the University's Academic Integrity policy. Students are expected to strictly avoid academic dishonesty and adhere to the Academic

Integrity policy as outlined in the course catalog. Violations will be dealt with as outlined therein.

As a general rule, all work submitted for grading must be the student's own work. Students are allowed to help each other solve compiling and linking problems, and may generally discuss issues related to a student's particular program, but students may not share code, write code, or examine another student's code.

In regard to homework and projects, students may discuss the problems (what is being asked for), appropriate material from class lectures or the textbook or acceptable other sources. Students, however, may not share answers or the specifics of how to answer the question.

Use of material from previous classes, solution manuals, material from the Internet or other sources (e.g., parents, siblings, friends, etc.) that directly bears on the answer is strictly prohibited.

At the discretion of the instructor, students may be asked to sign a statement that they have abided by the University's Academic Integrity policy and its application to this class. This statement may appear on homework, tests, or projects.

When in doubt, consult the course professor before doing something that may result in violation of the University's Academic Integrity policy.

Application to this course:
Programming assignments are to be done by the student alone. No outside help is permitted. If you need help on a programming assignment you can only receive aid from the instructor of the course, teaching assistant or approved tutors.

The sanction for the first violation of the Academic Integrity policy or plagiarism policy will result in a minimum failing grade on the relevant assignment and the violation will be reported to the student's department chair. Once the final decision has been rendered and any or all appeals exhausted by any parties involved, the instructor or appropriate parties will carry out the recommended sanction.

Personal conduct: In order to minimize distractions and interruptions, students will be expected to:

1. Arrive to class on-time and fully prepared.

2. Give the instructor full and undivided attention once the lecture has begun.

3. Turn off and stow all cell phones, pagers, and any other personal electronic devices once the lecture has begun.

Failure to adhere to these policies may result in immediate dismissal from class and loss of any in-class credit for relevant assignments or activities.

Computer use: Students are encouraged to use their laptop computer for taking notes and other activities directly related to the course. The manner in which students use the computer in class is considered a matter of honor and professionalism. Students will adhere to the following guidelines:

1. Computer use must be for taking notes or other activities related directly to the course.

2. Computer use must be subtle and must not distract fellow classmates or the instructor.

Inappropriate use of a computer in the classroom will be viewed as disrespectful to the instructor and classmates and will be considered unprofessional. Examples of inappropriate use include, but are not limited to:

- Sending, receiving, or reading e-mail
- Instant messaging
- Web browsing
- Working on assignments
- Playing games
- Listening to music
- Watching movies

Judgments regarding the appropriateness of computer use are at the discretion of the instructor. The consequences for violating this policy are also at the discretion of the instructor, and may include loss of in-class computer privileges, grade reduction, and so forth.

Course readings: Readings are to be completed before the class session for which they have been assigned; material covered in each reading is fair game for class discussions and unannounced quizzes.

Assignments: Assignments will be distributed via a mixture of paper and electronic means. Students are responsible for managing due dates and understanding submission procedures to turn in programming assignments.

Final exam: In lieu of a final exam, students will submit a Final Portfolio no later than December 21st.

Late work: Late assignments or projects will not be accepted without prior approval. Students must consult the instructor at least three days prior to the scheduled due date of any assignment to make alternative arrangements; however, the instructor is under no obligation to grant any such request. Penalties such as a reduced score may be applied at the instructor’s discretion.

In-class exams will not be rescheduled except in the event of an excused absence. The instructor will work with the student to schedule such a makeup exam; however, it is in the student's best interest not to miss the regularly scheduled in-class exam.

Extremely, and very much not set in stone, Tentative Schedule / my best guess

Project Topic	**Week**
Class introduction/overview; programming environment setup; introduction to SFML API	1
Recursive graphics	2
Linear Feedback Shift Register, Unit Testing	3
Encoding Images	4
Smart Pointers	5
Lambda Expressions (part 1)	6
Lambda Expressions (part 2)	7
Concurrency in C++ (part 1)*	8
Concurrency in C++ (part 2)*	9
Kronos Intouch parsing (part 1)	10
Kronos Intouch parsing (part 2)	11-12
Ring Buffer with cpplint, testing, and exceptions; GuitarHero	13-14

***Note: Readings & programming examples provided by instructor.**

Readings in Required Text (Lippman et al., *C++ Primer*)	**Chapter**
Dealing with Types	2.5
Generic Algorithms (Lambda Expressions)	10
Dynamic Memory (Smart Pointers)	12
Copy Control	13
Function Call Operations & Conversions	14.8-14.9
Templates and Generic Programming	16.2-16.5
Multiple and Virtual Inheritance	18.3
Specialized Tools and Techniques	19