Concordia University
Department of Computer Science and Software Engineering

Advanced Program Design with C++
COMP 345 --- Fall 2021

Contact information

Instructor :	Dr. Joey Paquet, paquet@encs.concordia.ca
Web page :	www.cse.concordia.ca/~paquet/
Youtube channel	www.youtube.com/playlist?list=PLAqc0kXi5yE3SVH2AGn5rAzrKqqjDWl0-
Lectures :	Section D -T-J--- 13:15 - 14:30 Tuesdays in-person – Thursdays online Section N M-W---- 16:15 - 17:30 Mondays online – Wednesdays in-person

Calendar description

COMP 345 - Advanced Program Design with C++ (4 credits) Prerequisite: COMP 352 previously or concurrently. Introduction to C++. I/O with stream classes. Pointers and their uses. The Standard Template Library (STL): containers, algorithms, iterators, adaptors, function objects. Class design: constructors, destructors, operator overloading, inheritance, virtual functions, exception handling, memory management. Advanced topics: libraries, locales, STL conventions, concurrency, template metaprogramming. Applications of C++: systems, engineering, games programming. Project. Lectures: three hours per week. Laboratory: two hours per week.

Rationale

Most of our courses are taught using the Java programming language. C++ programming is pervasive in many key areas of the software industry. Though C++ and Java have many similar syntactical elements and structures, C++ has many subtleties and features that differ from Java. This course aims at teaching C++ to an audience well-trained in computer programming and putting the newly acquired knowledge into practice through a challenging project.

Learning objectives

- Master the theoretical and practical concepts underlying the implementation of the C++ language.
- Deliver operational C++ programs given an open-ended problem description and design restrictions.
- Develop C++ programs of significant complexity and size.
- Use C++ language constructs, libraries and tools to develop well-structured solutions.
- Deliver operational software in an oral presentation.

Prerequisite knowledge

It is assumed that all students have extensive experience with computer programming, though no prior knowledge or experience of C++ is assumed.

Course Deliverables

Assignments: There will be 3 programming assignments. For each assignment, you will have to (1) demonstrate your code to the marker (demo time slots will be scheduled a few days before the assignment due date) and (2) submit it on moodle. Assignments will build on earlier assignments to form a bigger project. Assignments are done in teams. Failure to give a demonstration will incur a mark of zero for this assignment. All assignments are to be submitted electronically at the latest at 11h59pm on the due date. No assignment will be accepted after the due date.
Examinations: There will be one midterm examination, focusing on the concepts covered in the lectures. The midterm will be around the middle of the term; the exact date will be announced in class and posted on the course web page. The final examination will be divided in two parts: a regularly scheduled final examination focusing on the concepts covered in the lectures (scheduled during the final examinations period), and a practical programming examination where you will be asked to solve programming problems in a lab environment. The practical programming examination will be scheduled toward the end of the semester and will be announced on the course web page.

Suggested textbooks (non mandatory)

Walter Savitch, *Absolute C++*. Sixth Edition, Addison-Wesley, 2015. ISBN-13: 978-0133970784
Bjarne Stroustrup, The C++ Programming Language. Fourth edition. Addison-Wesley, 2013. ISBN-13: 978-0-321-56384-2

Evaluation

Midterm examination	20%
Final examination (written and programming)	(35% + 15%) = 50%
Assignments (3)	3 X 10% = 30%

There is no standard predefined relationship between percentages and letter grades assigned for this course. Letter grades are calculated using a normal curve based on the numerical class average. In order to pass the course, you must obtain a passing grade (i.e. at least 50%) for the overall mark, and you must pass the final examination, as well as the assignments.

Online/in-person teaching

Exceptionally, due to the COVID-19 pandemic, most of this course will be offered online, i.e. the material for all lectures, and labs will be made available online. The examinations and assignment demonstrations will be made online. As we are gradually going back to in-person teaching, this course will also offer some lectures and lab sessions in-person.
Lectures: All lectures are pre-recorded and made available online for the students to view. The instructor will then be available during the regular class time for discussions on the topics covered during the current week. Such discussions will be done alternatively online and in-person (see “Contact Information” above). It will be expected during these discussions that the students have already viewed the weekly lecture videos and reviewed the labs’ programming examples that correspond to them.
Labs: Each weekly lab session is pre-recorded and made available online for the students to view. The lab instructors will then be available during the scheduled lab times for online discussions. There will also be some weekly in-person lab sessions identified on the course web page. The labs have two main uses: 1) discussions about the program examples related to the current weekly material covered in class, as presented in the weekly lab videos 2) programming coaching for the assignments. The labs are setup for online remote access.
Assignment demonstrations: Markers will organize with each team a live video conference private demonstration for each assignment. Students will use a shared screen to demonstrate the functionality of their code versus the requirements stated in the assignment handouts.
Examinations: There are 3 examinations: 1) a midterm online written examination scheduled during regular class time around the middle of the semester, 2) a final online programming examination scheduled during the last week of regular laboratory sessions of the semester, and 3) a final online written examination scheduled during the examination week. All these examinations are held using a moodle quiz.

Graduate attributes

As part of both the Computer Science and Software Engineering program curriculum, the content of this course includes material and exercises related to the teaching and evaluation of *graduate attributes.* Graduate attributes are skills that have been identified by the Canadian Engineering Accreditation Board (CEAB) and the Canadian Information Processing Society (CIPS) as being central to the formation of Engineers, computer scientists and information technology professionals. As such, the accreditation criteria for the Software Engineering and Computer Science programs dictate that graduate attributes are taught and evaluated as part of the courses. This particular course aims at teaching and evaluating 3 graduate attributes. The following is a description of these attributes, along with a description of how these attributes will be incorporated in the course.

Knowledge-base: Application of advanced programming principles using a mid-level programming language. Use of design patterns and architectural design. Manual/explicit memory management. Language/compiler/runtime system implementation concepts and details.
Design: The project in this course is presented in an open-ended fashion, and its size and complexity is such that it needs to be tackled in a series of assignments Individual assignments provide a platform for designing at a smaller level, and provide the additional difficulty of having to be integrated in the larger design of the project.
Use of tools: The course teaches the use of the C++ language, and leaves the students free to select what programming environment and libraries that they will use in the assignments and project. Selection and use of the right tools and libraries is a crucial aspect of accomplishing the practical work.
Individual and team work: Work on a relatively large project, divided in a sequence of assignments.
Communication skills: Proper code documentation using code comments, as well as simple description of design decisions. Oral presentations for project deliveries.

General Notes

In the event of circumstances beyond the instructor’s or the University’s control, the content and/or evaluation scheme of this course is subject to change. In particular, should there be a change for the worse in the COVID-19 situation, the course may have to proceed entirely online without any in-person lectures nor labs.

Students are expected to be aware of the University’s Code of Conduct, what constitutes any violation to this code, and what are the consequences of violating this code.