San José State University
Department of Computer Science

CS 144
Advanced C++ Programming

Section 1
Spring 2019

Course and Contact Information

Instructor:	Ron Mak
Office Location:	ENG 250
Email:	ron.mak@sjsu.edu
Website:	http://www.cs.sjsu.edu/~mak/
Office Hours:	TuTh 3:00 - 4:00 PM
Class Days/Time:	TuTh 9:00 – 10:15 AM
Classroom:	MacQuarrie Hall MH 233
Prerequisites:	CS 46B and CS 49C (with a grade of C- or better in each), or equivalent knowledge of object-oriented programming and C, or instructor consent.

Course Format

This course will be taught primarily via classroom presentations.

Faculty Web Page and Canvas

Course materials, syllabus, assignments, grading criteria, exams, and other information will be posted at my faculty website at http://www.cs.sjsu.edu/~mak and on the Canvas Learning Management System course login website at http://sjsu.instructure.com. You are responsible for regularly checking these websites to learn of any updates. You can find Canvas video tutorials and documentations at http://ges.sjsu.edu/canvas-students

Course Catalog Description

Advanced features of C++, including operator overloading, memory management, templates, exceptions, multiple inheritance, RTTI, namespaces, tools.

Instructor's Description

We will also examine some of the features of modern C++, such as lambda expressions, smart pointers, and move semantics. We will develop interactive GUI applications and explore multithreaded programming. We'll learn how to write efficient C++ programs that adhere to good design principles while avoiding pitfalls of the language.

Course Learning Outcomes (CLO)

Upon successful completion of this course, you will be able to:

CLO 1: Apply **object-oriented features** of C++, including polymorphism and recursion.

CLO 2: Apply **advanced features** of C++, including operator overloading, memory management, templates, the Standard Template Library (STL), exceptions, multiple inheritance, runtime type identification (RTTI), namespaces, etc.

CLO 3: Apply **modern features** of C++, including lambda expressions, smart pointers, move semantics, etc.

CLO 4: Develop **interactive GUI-based applications** in C++ that use inversion of control and callback functions as event handlers.

CLO 5: Understand the concepts of **multithreaded programming**.

CLO 6: Use **high-level software development tools**, including an integrated development environment (IDE), compilers, linkers, and source-level debuggers to implement and debug C++ applications.

CLO 7: Write **efficient programs** in C++ that adhere to good design principles while avoiding pitfalls of the language.

CLO 8: Document program design with **Unified Modeling Language** (UML) diagrams.

Academic Integrity

You may study together and discuss the assignments, but what you turn in must be your **individual work**. Copying code from another student's program or sharing your program code are equally serious violations of academic integrity.

Never use code you find on the web, unless you have the instructor's permission, and then you must give proper attribution (where did you find that code?) in your comments. This is similar to giving attribution to a quote that you use in a term paper.

Assignment submissions will be checked for plagiarism using Moss from the Department of Computer Science at Stanford University. See http://theory.stanford.edu/~aiken/moss/

See http://www.cs.sjsu.edu/~mak/Moss/ for a report from an actual Moss run. Moss is not fooled by renaming variables, reformatting code, or re-ordering functions.

Violators of academic integrity will suffer severe sanctions, including academic probation. Students who are on academic probation are not eligible for work as instructional assistants in the university or for internships at local companies.

Recommended Texts

Title:	**C++ How to Program**, *10th edition*
Author:	Paul J. Deitel and Harvey Deitel
Publisher:	Pearson International, 2017
ISBN:	978-9332585737
Title:	**Effective Modern C++** **42 Specific Ways to Improve Your Use of C++11 and C++ 14**
Author:	Scott Meyers
Publisher:	O'Reilly Media, 2014
ISBN:	978-1491903995
Title:	**The C++ Standard Library** **A Tutorial and Reference**, *2nd edition*
Author:	Nicolai M. Josuttis
Publisher:	Addison-Wesley Professional, 2012
ISBN:	978-0321623218

Software to install

This class will use the GNU C++ compiler. You should install and use an interactive development environment (IDE) such as Eclipse. During the semester, you will download, configure, build, and install the Multiple Precision Integers and Rational (MPIR) package (http://mpir.org) and the wxWidgets library for GUI-based development (https://www.wxwidgets.org).

These builds and installs are relatively straightforward on the Mac and Linux platforms. However, the Windows platform often has significant compatibility challenges. Therefore, if you're on Windows, we highly recommend that you download and install the VirtualBox virtual machine manager, and then install and run Ubuntu (a variant of Linux) in a virtual machine.

Some useful tutorials:

- "Install and Configure VirtualBox on Windows"
  http://www.cs.sjsu.edu/~mak/tutorials/InstallVirtualBox.pdf
- "Install and Configure Ubuntu on a VirtualBox Virtual Machine"
  http://www.cs.sjsu.edu/~mak/tutorials/InstallUbuntu.pdf
- "Install and Configure Eclipse on Ubuntu for Java and C++ Development"
  http://www.cs.sjsu.edu/~mak/tutorials/InstallEclipse.pdf
- "Install MPIR on Ubuntu"
  http://www.cs.sjsu.edu/~mak/tutorials/InstallMPIR.pdf
- "Install and Configure wxWidgets on Ubuntu"
  http://www.cs.sjsu.edu/~mak/tutorials/InstallwxWidgets.pdf

Course requirements and assignments

This class will progress rapidly, and you must work hard to keep up. There will be in-class quizzes during to check your understanding, and multiple programming assignments each week.

Each assignment will be worth a specified maximum number of points, depending on difficulty, and it will be due before the start of the next class. No assignment will be accepted after its solution is presented in class (it will get a 0 score). Each assignment will include rubrics for its grading criteria.

This is a challenging course that will demand much of your time and effort throughout the semester.

Learning to program in a new language requires much practice. Each week, there may be several short practice programs that emphasize specific language features that you will need to master to complete that week's main programming assignments. All the practice programs and many of the main assignments will be graded automatically online.

The university's syllabus policies:

- University Syllabus Policy S16-9 at http://www.sjsu.edu/senate/docs/S16-9.pdf.
- Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

"Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally 3 hours per unit per week with 1 of the hours used for lecture) for instruction or preparation/studying or course related activities including but not limited to internships, labs, clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus."

Exams

The quizzes, midterm, and final examinations will be closed book. The exams will test understanding (not memorization) of the material taught during the semester. Instant messaging, e-mails, texting, tweeting, file sharing, or any other forms of communication with anyone else during the exams will be strictly forbidden.

There can be no make-up quizzes and midterm examination unless there is a documented medical emergency. Make-up final examinations are available only under conditions dictated by University regulations.

Grading Information

Your final class grade will be weighted as follows:

50%	Assignments
15%	In-class quizzes
15%	Midterm exam
20%	Final exam

Each assignment and exam will be scored (given points) but not assigned a letter grade. The average score of each assignment and exam will be available in Canvas after it has been graded.

Final course grades will be based on a curve. The median total score will earn a B–. Approximately one third of the class will earn higher grades, and another one third will earn lower grades.

Classroom Protocol

It is very important for each student to attend classes and to participate. Cell phones in silent mode, please.

University Policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

CS 144
Advanced C++ Programming

Section 1
Spring 2019

Course schedule

Subject to change with fair notice.

Week	Dates	Topics
1	Jan 24	Introduction C++ basics Flow of control Simple input and output (I/O)
2	Jan 29 Jan 31	Procedural abstraction Functions
3	Feb 5 Feb 7	I/O streams Introduction to classes and objects Arrays Strings Vectors
4	Feb 12 Feb 14	Pointers Dynamic arrays
5	Feb 19 Feb 21	Structures Classes Public and private members Constructors and destructors Friend functions Abstract data types (ADT)
6	Feb 26 Feb 28	Analysis precedes design Where do classes come from? UML class and sequence diagrams Separate compilation Namespaces Inlining
7	Mar 5 Mar 7	A class design example Accessors and mutators Immutable classes The Law of Demeter and the Principle of Least Knowledge Cohesion and consistency The Liskov Substitution Principle

Week	Dates	Topics
8	Mar 12 Mar 14	***Midterm exam Tuesday, March 12*** Class hierarchies Inheritance Overriding and overloading functions Operator overloading The Open-Closed Principle
9	Mar 19 Mar 21	Copy constructors The assignment operator The "Big Three" A "safe" array type Linked lists Stacks Queues
10	Mar 26 Mar 28	Polymorphism Virtual destructors Abstract classes and interfaces Multiple inheritance Runtime type identification (RTTI) The Principle of Coding to the Interface The Principle of Favoring Delegation over Inheritance
	Apr 1 - 5	***Spring break***
11	Apr 9 Apr 11	Recursion Binary search Mergesort Exception handling Template functions and classes
12	Apr 16 Apr 18	Standard Template Library (STL) STL Containers STL vectors STL linked lists STL iterators STL sorting
13	Apr 23 Apr 25	The Model-View-Controller architecture Interactive programming with a graphical user interface (GUI) Introduction to wxWidgets Inversion of control Callback functions Events and event handlers Lambda expressions The **auto** keyword The **decltype** pseudo-function

Week	Dates	Topics
14	Apr 30 May 2	Constructor and destructor calls How does an STL vector grow? Why did my program crash? Shallow vs. deep copy Pointers vs. references Raw pointers vs. unique and shared smart pointers Move semantics
15	May 7 May 9	Introduction to multi-threaded programming Critical regions, mutexes, and semaphores Introduction to algorithm analysis Recurrence relations Proof by induction Big-O notation Rates of growth and scalability The quicksort algorithm
Final exam	***Friday, May 17***	Time: 7:15 - 9:30 **AM** Room: MH 233