Advanced Programming
CMPS 109

Dr. Karim Sobh
Computer Science Department
Jack Baskin School of Engineering
ksobh@cs.ucsc.edu

Fall 2016

Basic Information

- **Course Title:** Advanced Programming (CMPS 109)
- **Prerequisites:** Introduction to Programming and Data Structures (CMPS 13H), or Introduction to Data Structure/ Data StructuresLab (CMPS 12B/ CMPS 12M).
- **Lectures:** TuTh 01:30 PM - 03:05 PM (Steven Acad 175)
- **Instructor:** Dr. Karim Sobh (ksobh@ucsc.edu)
  Office Hours: 16:00 - 17:30 MTW
- **TAs:**
  Karthik Mohan Kumar (kmohanku@ucsc.edu)
  Office Hours: 10:00-11:30 MWF
  Sharath TS (sturuvek@ucsc.edu)
  Office Hours: 9:30-11:30 W, 15:00-17:00 Th
- **Home page:** `https://cmps109-fall16-01.courses.soe.ucsc.edu/`
- **Discussion:** `https://piazza.com/ucsc/fall2016/cmps109`
- **Git Repository:** git@git.soe.ucsc.edu:classes/cmps109/fall16/cruzid
- **Git How-to:** `https://git.soe.ucsc.edu/~git/index.html`

Catalog Description

An introduction to object-oriented techniques of software development including data abstraction, inheritance, polymorphism, and object-oriented design. Extensive practice using a computer to solve problems, including construction of graphical user interfaces and a multi-threaded client/server applications.

Textbooks and References

It is required to use the first book, and any C++ book or reference not discussing C++11 should be considered obsolete. We will also cover some material from the last book, Grady Booch, related to the OO design notation. You can use also any of the listed references as complementary sources.

1. Bjarne Stroustrup : Programming Principles and Practice Using C++, second edition. Addison-Wesley, 2014. ISBN 0-321-99278-4. This is an elementary textbook for a first course in C++. The previous edition will also work, or any other C++ textbook you may already have. It is strongly recommended that you read a book advertising the use of C++11. Any C++ book not discussing C++11 should be considered obsolete.

2. Paul Deitel and Harvey Deitel: C++ How to Program (Early Objects Version), Prentice Hall, 9th edition (February 22, 2013), ISBN-13: 978-0133378719.
3. Bjarne Stroustrup: The C++ Programming Language (4th Edition), Addison-Wesley Professional, 4th edition (May 19, 2013), ISBN-13: 978-0321563842. `http://www.stroustrup.com/4th.html`
4. Stanley B. Lippman, Josee Lajoie, Barbara E. Moo: C++ Primer, 5th edition. Addison-Wesley, 2013. A good primer discussing C++11.
5. Frank B. Brokken: C++ Annotations Version 10.5.0, Center of Information Technology, Published at the University of Groningen, ISBN 9036704707, `http://www.icce.rug.nl/documents/cplusplus/`.
6. Nicolai M. Josuttis : The C++ Standard Library, 2nd edition : A Tutorial and Reference. Addison-Wesley, 2012. A specific tutorial on the library, with C++11
7. Bjarne Stroustrup FAQs
   - http://www.stroustrup.com/C++.html
   - http://www.stroustrup.com/C++11FAQ.html
8. P.J. Plauger, Alexander Stepanov, Meng Lee, David Musser : The C++ Standard Template Library. Prentice-Hall, 2001. Detailed description of the implementation of the STL, showing detailed code examples.
9. Grady Booch et al.: Object-Oriented Analysis and Design with Application, Third Edition. Addison-Wesley, 2007. ISBN-13: 978-0201895513.

Course Goals

1. Understand the basic object-oriented programming concepts.
2. Understand the basic object-oriented modeling concepts.
3. Develop hands-on experience in advanced object-oriented programming.
4. Understand the importance of code reuse.
5. Be able to design and implement a medium-sized object-oriented software.

Major Topics Covered

The course is mainly about object-oriented programming and design using C++/C++11.

1. **Fundamentals:** C vs. C++, Declaration and Definition, Data Types, Function calls and Return Values, Pointers to functions, Order Evaluation, Memory Allocation, Code Organization, make files and compilers.
2. **Standard Library:** Containers and Iterators.
3. **Object-Oriented Programming in C++:** Main, OO Characteristics, Classes, Objects, Constructors, Inheritance, Virtual Methods and Polymorphism, Virtual Methods, Templates, Exception Handling, Operator Overloading, Static Methods, and Virtual Tables.
4. **Object-Oriented Design and Modeling:** CASE-Tools, UML, Booch, OO Design and Modeling Process and Diagrams.
5. **GUI:** QT UI Components, Signal/Slot Communication Model.
6. **Multithreading and Concurrency:** C++11 Native Multi-threading, POSIX Threads, and QT Threads.
7. **Networking APIs:** BSD Sockets.

Exams

Students must attend all exams, the midterms and the final. In case of absence due to emergency, the student needs to let the professor know before the exam scheduled time. Moreover, the student will be required to present a legitimate justification as proof such as a **doctor's note or letter from the funeral home** before taking a makeup for the exam.

Assignments

Must be submitted electronically with a README file explaining how to compile and run the programs. All code must work on Unix and be compiled using the GNU GCC compiler. A report with a detailed explanation of the work will need to be submitted whenever it applies and instructed. The due dates are announced with each assignment, and late assignments will be penalized 1% on every day, for a maximum of three late days after which the assignment will not be accepted, and the student will lose all points for the assignment.

Notes and Class Participation

Students are required to take notes and participate in class. Part of the grade is attributed to how active the student is. Weekly notes must be turned in on eCommons latest by Sunday 9 PM of the week after the material was covered. The notes can be hand written or typed by **the student**, and they should not be copied form textbooks, class slides, other students, or the internet.

There are many forms of participation, mainly class attendance, actively participating in class discussions, visiting course staff during their office hours, and/or participating in the piazza discussion. The student is encouraged to perform all of these as it is of great benefit to the student and might enhance the final grade.

Grading

The course overall grade is split over two components; exams and programming assignments. A student needs to get at least 50% of the total grade of each component to pass the course; Each component needs to be passed. For example the following cases will definitely fail the course:

- Scoring 55% in the exams and 51% in the assignments.
- Scoring 100% in the exams and 30% in the assignments.
- Scoring 28% in the exams and 90% in the assignments.

The final grade will be calculated as follows:

• Homework Assignments:	45%
• Midterm Exam:	20%
• Final Exam:	30%
• Participation & Attendance:	5%

The following are tentative approximate ranges of the overall scores:

- A: 89-100%
- B: 79-89%
- C: 69-79%
- D: 60-69%
- F: below 60%

Individual exams, or assignment can be curved based on the situation and the overall distribution of the grades. Also, extra bonus work might be offered near the end of the course based on the grades distribution but with no guarantees.

Attendance

Students are required to attend the lectures. Exam questions will be designed to refer to lecture discussions and examples, and will require knowledge of specific details discussed in the lectures. Students failing to attend the lectures have a high chance of loosing marks on the exams. By not attending the lectures the student will also find it difficult to take good notes. Lab section attendance is not required, although important material on the programming projects will be missed if the student does not attend, since that will be where projects will be discussed in detail.

Getting Help

The student is advised to always ask for help as early as possible and not to wait. One way of getting help is through engaging in informative discussions with the course staff during their office hours. However, for these discussions to be fruitful the student should:

- Attend classes and lab sections.
- Read the course Web page for information on assignments.
- Read and post to the class discussion forum, hosted at piazza.com

The student should try to avoid dropping by outside the office hours as the course staff might be busy and might not be able to avail the time to help at the time. If the student cannot attend office hours, the student should arrange a meeting in advance by emailing the person he/she would like to meet with. Questions can also be posted to course staff via email as long as they can be replied to in short answers.

Students with Disabilities

If the student qualify for classroom accommodations because of a disability, sthe student should submit the Accommodation Authorization Letter from the Disability Resource Center (DRC) to me **as soon as possible**, preferably within the first week of the quarter. Contact DRC by phone at 831-459-2089 or by email at drc@ucsc.edu for more information.

Academic Honesty

Academic Honesty is a very important aspect in academic life. The students are expected to respect and adhere to the highest levels of academic integrity standards. There are many forms for cheating and all of them are considered a violation, which means that plagiarism is not accepted by any means. Consequently, students are not allowed by any means to do any of the following or anything similar:

1. Work on assignments with anyone.
2. Share code with anyone.
3. Share material or notes with anyone.
4. Share written class notes with peers or anyone.
5. Obtain help in assignments from anyone other than the assigned course staff.
6. Use of online past material that might help in solving the assignments.
7. Request help through online forums to help solve assignments.

If the student obtains help in any aspect of the practical work of the course the student is encouraged to document the source. In such cases it will not be considered or reported as cheating, nevertheless it might lead to a lower grade for that. So documenting the sources when turning the work in is highly advisable, as it will not count at all if it is caught afterwards.

By **anyone** we mean friends, family, former or current Computer Science students, other humans, animals, zombies, sparkling vampires, materials found on the Internet. Any cheating attempt in the assignments or the exams will result in failing the course. The Student is encouraged to read `http://registrar.ucsc.edu/navigator/section1/academic-integrity.html`. Moreover, the Academic Misconduct Policy for Undergraduates will be applied; the student should refer to `https://www.ue.ucsc.edu/academic\_misconduct` for more information. The policies will be applied on all parties participating in a misconduct, both the transmitter and the receiver of the information and material in any form.

The bottom line is: do not ever cheat, or you will definitely fail the course.