California State University, Sacramento
Department of Computer Science

CSC 133: Object-Oriented Computer Graphics Programming – Spring 2018

COURSE SYLLABUS

Instructor: Dr. Doan Nguyen

Classroom:	RVR 1008 (Sect 2) ARC 3004 (Sect 4)
Class Days:	T Th
Class Times:	3:00PM-4:15PM (Sect 2) 1:30PM-2:45PM (Sect 4)

Office Hours: T/Th: 10:30AM-1200 Noon and by appointment
Office: Riverside Hall 3002
E-Mail: doan.nguyen@csus.edu
Course Materials: Syllabus, outline, notes, and other materials are available at the "Content" section of Canvas

Important reminder:

During class time, please keep your cell phones turned off and please refrain from browsing, social networking, gaming, messaging etc.

Catalog Description:

An introduction to computer graphics and to advanced topics in object-oriented (OO) programming. The OO paradigm is used throughout, utilizing computer graphics as the vehicle for solidifying basic OO concepts, studying the implementation of event-driven systems, and for developing a thorough understanding of advanced OO concepts such as inheritance and polymorphism. Topics include fundamental concepts of object-oriented programming, software design patterns, graphic devices, line and surface drawing, simple 2D and 3D representation, and use of User Interface components. Prerequisites: CSC 130 and CSC 131. Units: 3

Prerequisites:

CSC 130 – Algorithms and Data Structures & CSC 131 – Introduction to Software Engineering.

You must have already **completed** both CSC 130 and CSC 131 (or their equivalents at another school), each with a grade of C- or better, in order to take CSC 133. In addition you must have also completed CSC 15, CSC 20, CSC 28, and Math 29 (or their respective equivalents at another school), each with a grade of C- or better, since those courses are prerequisites for CSC 130 and/or CSC 131. Student records will be reviewed to determine whether a student has taken the required prerequisites and if not (s)he will be dropped from the class.
Completion of Math 30, Math 100, and Physics 11A will be very helpful, although not required.

Prerequisites by Topic:

• Three semesters (minimum) experience programming in a high-level language (C, C++, Java, Python, etc.);

• Experience with "Object-based" programming – class definitions, object instantiation, method invocation, public vs. private fields, etc.;

• Implementation of linear lists, including stacks & queues, and of binary trees; use of recursion in a program;

• Familiarity with UML class diagrams and their use in specifying relationships including aggregation, composition, and inheritance;

• Pre-calculus math including trigonometric functions, Cartesian coordinates, points, lines, and planes in space, coordinate transformations, conics, algebraic relations and functions, polynomial equations, inequalities, and matrix operations.

Repeat Policy:

Department policy specifies that students may not repeat a Computer Science course **more than three times** (that is, take a course for **a fourth – or subsequent – time**). Any student who wishes to repeat a course more than three times must submit a petition requesting permission to do so. Student records will be reviewed to determine whether a student is taking this course for a forth (or subsequent) time. Any such student must return an **approved** petition to the instructor **within the first two weeks of class.** Any student who does not submit an approved petition will be dropped from the class. Petitions are available in the Department office (Riverside Hall 3018) and require the signature of both the Instructor and the Department Chair.

Course Structure:

This course has two separates but equally important two main goals: an understanding of the advanced features of the so-called "object-oriented (OO) programming paradigm", and an understanding of the fundamentals of computer graphics (CG) programming. This course also allows students to gain experience in mobile application development by applying the introduced OO and CG concepts to this development environment.

We will cover the OO concepts of abstraction, encapsulation, inheritance, and polymorphism, including discussion of their variations both conceptually and in terms of language-specific implementations. We will also look at formalisms for specification of OO systems (specifically, Unified Modeling Language or UML). In addition, we will cover various design patterns for OO systems which will be emphasized as an underlying theme throughout the course.

In the CG area, we will cover hardware characteristics of graphics systems, and go through basic CG operations such as line and polygon drawing, 2D object modeling, geometric transformations, and transformations for mapping between "world" and "screen" space. We will also cover CG-based user interfaces (GUIs), including how event-driven components for GUI implementation work.

The OO and CG topics in the course will not be covered separately, but rather will be closely integrated and frequently co-mingled. It will be quite common to spend one class period (or even just part of one period) talking about some OO-related topic, and then switch to a discussion of a CG-related application of that OO topic. One reason for this is to insure equal emphasis on both of the course goals; another is simply that many of the OO topics have a strong relationship to various CG topics. Covering the topics in parallel thus has the extra benefit of reinforcement.

There is a potential drawback to organizing a course as just described: students who regularly **miss class**, or who habitually **come in late**, will likely find that the penalty for doing so is significantly greater than it might be in some other courses. You might find, for example, that a graphics concept is being explained by utilizing an OO concept previously explained (but which was missed due to lateness or non-attendance). This makes it considerably more difficult to understand the new topic, which in turn tends to have negative impact on both the amount of time it takes to do assignments and on your ability to do well on exams. The course organization described above should make it easy for students who attend class regularly and on time to keep up and do well.

This course utilizes mobile technology as a tool for teaching course topics and requires students to solve their assignments using this emerging technology. These allow students to relate course topics (e.g. OO programming, design patterns, graphical user interface design, event handling, animation, and computer graphics techniques) to mobile application development process, a skill that is increasingly demanded by the job market. Since the prerequisite course sequence at CSUS uses the Java programming language, as a mobile application development environment we will be using a Java-based framework called Codename One (CN1). Other advantages of CN1 includes: it is a cross-platform framework (i.e. the applications developed in this framework run on various mobile platforms including Android, iOS, and Windows); it is free for academic use and it is open-source; and it comes with a simulator environment (i.e. to develop and run the application you do not need to have a physical mobile device).

Texts and References:

The following text materials are **required**:

CSC 133 Lecture Note Slides, Spring 2018: available at the "Content" section of Canvas

Codename One Developer Guide - Revision 3.6: available at the "Content" section of Canvas

Codename One JavaDocs of APIs: https://www.codenameone.com/javadoc/index.html

The following text materials are **recommended**:

> **Object-Oriented Design & Patterns, 2nd Ed., by Cay Horstmann**; Wiley; ISBN 0-471-74487-5
>
> **Schaum's Outlines Computer Graphics, 2nd Ed., by Xiang & Plastock**; McGraw-Hill; ISBN 0-07-135781-5
>
> They are available in the Hornet Bookstore as well as numerous on-line booksellers.

Supplemental (online) materials:

> **Basic Debugging With Eclipse:** https://www.youtube.com/watch?v=PJWtO5wrptg
>
> Basic functions for using the Eclipse debugger with Java, such as breakpoint, step in/over functions, changing variable values, etc.

Since the mobile application development environment we will be using in this course (i.e. Codename One) is a Java-based framework, students not already familiar with Java may want to acquire a book on Java. A separate bibliography of popular Java texts, as well as a bibliography on "Java for C++ Programmers" topics, is available from the instructor. Some time will be spent during the semester focusing on important differences between Java and C++; however, these discussions will not be sufficient to comprise a tutorial on Java; it will be the responsibility of each student to be proficient in using the basic constructs of Java (this should not be a problem for students with solid background in object-based programming in C++).

Assignments:

Throughout the semester, programming assignments will be given which are not necessarily weighted equally. They are required to be solved with Java-based mobile application development environment called Codename One. Assignments will be cumulative; thus students should not skip an assignment.

Late assignments will be accepted (the instructor would rather have you do the work late than not at all) up until **ONE WEEK** past the original due date, but **a penalty of 5% per day will be applied to all late work.** Hence, assignments submitted one week after the deadline (and beyond) will not be accepted and the maximum late penalty will be 50%. School holidays and weekends will be counted as regular days in computing lateness of assignments, so for example

if an assignment was due on a Friday and was submitted on a Monday, it would be counted as three days late.

Assignment submissions will be done using Canvas; submission time as recorded in Canvas is considered the time at which assignments are submitted. Assignment submissions can be replaced (updated) prior to the due date, but **assignments which have been submitted cannot be replaced once the due date has passed.** Technically, Canvas allows you to submit new versions indefinitely. However, the abovementioned assignment submission policy dictates that the version submitted right before the due date will be graded. If no such version exists, the version submitted right after the due date will be graded (as late assignment). Hence, if you are planning to do a late submission, you should not submit a version of your assignment before the due date (because if you do a submission before due date, versions submitted after due date will be ignored). Also, you should submit your late submission when it is ready (because after you submit the first late version, the submissions you make later will be ignored).

Exams:

There will be a midterm exam and a final exam. Study guides will be provided before the exams. However, you are responsible for knowing all the content of CSC 133 Lecture Note Slides. Makeup exams are not given except under the most extreme (and documented) circumstances, in which case every effort should be made to contact the instructor in advance.

There will attendance quizzes. Attendance quizzes cannot be made up.

Grading:

The following scale is used in determining the grades:

Range	Letter Grade
94-100	A
90-93	A-
87-89	B+
84-86	B
80-83	B-
77-79	C+
74-76	C
70-73	C-
67-69	D+
64-66	D
60-63	D-
59 or Less	F

Overall course grades will be computed using the following policy:

Programming Assignments	40%
Midterm Exam	20%
Final Exam	25%
Attendance & Quizzes	15%

In addition to grades being computed as described above, one further constraint applies: **in order to achieve a passing grade of at least C- for the course, it is a requirement that you achieve at least passing completion (that is, D- or better) of (1) the average of Programming Assignments; (2) the average of the exams (midterm and final exams); and (3) Attendance & Quizzes.**

At the end of the semester, a final score of each student will be calculated according to the above policy. These scores will then be curved based on final scores of all students in the class to determine the final grades. A positive curve will be used. That is, the grading scale listed above indicates the minimum grade that a student will receive based on his/her un-curved final score.

Roll will be taken regularly and the attendance grade will be proportional to the number of classes that the student is present. The student will be counted as absent for the class that (s)he does not attend unless the instructor is notified that there is an unusual (and documented) circumstance.

Students are required to keep backup (soft) copies of all submitted work until after final grades are posted.

Computers & Communication:

There are several computer options available for doing class assignments. Students may work on any school machine onto which Eclipse IDE for Java Developers (Neon version) and Codename One (CN1) have been installed or on their personal machine provided that they install the abovementioned software.

Eclipse IDE for Java Developers (Neon version) and CN1 are installed to ECS Open Labs [RVR 2011, SCL 1234, SCL 1208 (24 hour lab)], ECS Teaching Labs [ARC 1014/1015 (classroom instruction only labs)], CSC Labs [RVR 1013/2005/2009/2013/5029], and ECS Windows Terminal Server called Hydra. Instructions for reaching Hydra terminal can be found at https://www.ecs.csus.edu/computing/help_docs/Connecting_to_Hydra-Windows_and_Unix.pdf. For better performance please use lab machines or your own machine instead of remotely logging into Hydra.

CN1 installation instructions can be found at http://www.codenameone.com/download.html. CN1 is installed as a plugin to one of the following Integrated Development Environments (IDEs) which run on various operating systems: Eclipse, NetBeans, or IntelliJ IDEA (all of which are free). However, for solving the CSC 133 assignments, the students are **required** to use CN1 plugin which is installed to **"Eclipse IDE for Java Developers - Neon version"** (available at http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/neonr)**.** In addition, the instructor **recommends** running **Windows** as the operating system**.** Please note that prior to installing the Eclipse and CN1, the students must install latest version of Java SE Development Kit (JDK) - version 8 - which is freely available at http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html.

If you choose to work on the assignments on your personal machine, prior to submitting them, you are **strongly encouraged** to build and test them on one of the lab machines and then submit the files tested on the lab machine. Since the assignments will be graded on a lab

machine, this step will ensure the validity of your submission (e.g., if your personal machine is running Linux, MacOS or a different version of Windows other than Windows 7, this step will make sure that your assignments also work as expected on a lab machine which runs Windows 7). To build and test your assignment in the lab, you can copy your assignment directory located in the Eclipse workspace directory of your machine to the Eclipse workspace directory of the lab machine and use "File -> Import -> General -> Existing Project into Workspace" option of Eclipse to import your project to the lab workspace.

Regardless of which computer option you choose for doing class assignments, you are expected to be familiar with and regularly read "Discussions" section of Canvas. It is assumed that notices sent to the "CSC 133 – Communication Forum" located under "Discussions" are received and read by you (Tip: You can use "Subscribe" functionality of the forum to get a notification e-mail sent to your SacLink account whenever a new notice is posted to the forum.). You need to have SacLink account to login to Canvas. If you do not have SacLink account, create one through mysaclink.csus.edu.

Using the "CSC 133 – Communication Forum" you can send questions or discussion items regarding topics in CSC 133 to everyone in the class. This is a good way to ask your fellow students for clarifications about assignments, lecture topics, etc. Keep in mind that each such notice you send is read by everyone in the class (including the instructor). If you need to communicate privately with an instructor, use the instructor's individual email address as given above.

In addition to posting them to "CSC 133 – Communication Forum", the instructor may also send some of the important announcements directly to your SacLink e-mail.

Lateness/Absence/Drops:

It is very important for students to attend all classes and to be on time. Students who miss a class are responsible for all material discussed or handed out in the class.

Please inform yourself of the Department, College, and University policies on dropping courses. Policy information is available in the Computer Science Department office, Riverside Hall 3018.

Ethics:

When a student submits work to the instructor, it constitutes a contractual agreement that the work is solely that of the student. Further, submitted work carries with it an implicit agreement that the instructor may quiz the student in detail about the work.

Since the class is graded on a curve, if a student attempts to raise their grade through unethical means it is in essence causing a lowering of the grades of other students in the class. Further, any student who gives such help is equally guilty of unethical behavior for the same reasons. Therefore, all students are hereby notified that this course is being taught by instructors who will pursue attempts at cheating, since students who are cheating are cheating on you.

The **minimum penalty** for even a **single incident** of cheating in this course is **automatic failure of the course**; additional more severe penalties may also be applied. Note that cheating is grounds for dismissal from the University.

Please refer to the instructors' policy on academic integrity entitled "Ethics in Computer Science Classes", to the Computer Science Department's document entitled "Policy on Academic Integrity", and to the University's document entitled "Policy Manual section on Academic Honesty", which are all available online via the "Content" section of Canvas, for additional information. It is the responsibility of each student to be familiar with, and to comply with, the policies stated in these documents.