California State University, Sacramento
Department of Computer Science

CSC 133: Object-Oriented Computer Graphics Programming – Fall 2022

COURSE SYLLABUS

Instructor:	Dr. Pınar Muyan-Özçelik
E-Mail:	pmuyan@csus.edu
Course Materials:	Syllabus, outline, notes, recorded lectures, and other materials are available on Canvas
Office Hours:	Wednesday & Friday 8:45 – 10:15 am (held via Zoom, see the "Announcements, Class Meetings, and Links to Lectures/Office Hours" module of the "Home" section of Canvas for the Zoom link)

	Section-1	**Section-2**
Classroom:	Alpine Hall 235	Mariposa Hall 1002
Class Days:	TR	TR
Class Times:	10:30AM-11:45AM	12:00PM-1:15PM

Class Modality: CSC 133 is delivered using flipped-classroom methods this semester. Most weeks, we will meet at the scheduled class start times listed above only once a week (mostly on Thursdays). Some weeks (like the first week of instruction) we will meet both on Tuesday and Thursday. **All class meetings will be in-person.** On the days we meet, a part of the (or the whole) lecture will be conducted in class and/or questions about the programming assignments and course topics will be discussed. **Our first meetings will be held on the lecture days of the first week of instruction (Aug/30/2022 and Sep/01/2022).** The other days that we will meet will be announced on Canvas. **However, the majority of the lectures will be completely pre-recorded and posted to Canvas** (hence, on the days of those lectures, either we will not meet or if we will meet, questions about programming assignments or course topics will be discussed)**. See "Recordings and Meetings" section below for more info.**

Important reminders:

You are not allowed to save a copy of the recorded lectures. Also, you are not allowed to share the links of these recordings and office hours with anyone else.

You are not allowed to make audio or video recordings of the class meetings and office hours.

During class time, please refrain from browsing, social networking, gaming, messaging etc.

Catalog Description:

Introduction to computer graphics and to advanced topics in object-oriented programming. Mobile application development; implementation of event-driven systems; advanced object-oriented concepts such as inheritance and polymorphism; implementation of software design patterns; graphical user interface development; fundamentals of 2D graphics systems. Application of these topics to mobile programming. Prerequisites: CSC 130 and CSC 131. Units: 3

Important Dates:

September 5 (Monday)	Labor Day (campus closed)
October 20 (Thursday of Week 8)	Midterm exam
November 11 (Friday)	Veterans Day (campus closed)
November 24-25 (Thursday-Friday)	Thanksgiving (campus closed)
Week 16	Final exam (see the outline for the exact date/time)

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:

Please make yourself familiar with CSUS Repeating Courses Policy located at https://www.csus.edu/student-life/class-schedules/registration/repeating-courses.html. This policy specifies that students may not repeat a course for a **third – or subsequent – time** (that is, take a course for a **fourth – or subsequent – time**) without petition. And be aware that our

College is not approving any petitions for exceptions to this repeat policy and hence, does not allow students to take a course for a fourth time. Hence, if you are taking CSC133 for the third time, be aware that you will not be able to take CSC133 for the fourth time at CSUS and you will need to take a similar course elsewhere and transfer it to CSUS.

Course Structure:

This course has two separate 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 do not regularly follow lectures, 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 not following lectures). 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 follow lectures regularly, 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, Fall 2022: available at the "Home" section of Canvas (under the "Lecture Note Slides" module)

Codename One Developer Guide - Revision 3.6: available at the "Home" section of Canvas (under the "Resources" module)

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

The following text materials are **recommended (not required)**:

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

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 **ten days** past the original due date, but **a penalty of 5% per day will be applied to all late work.** Hence, assignments submitted eleven days 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. Late penalty is not waived except under the most extreme (and documented) circumstances, in which case every effort should be made to contact the instructor in advance.

Assignment submissions will be done using Canvas; submission time as recorded on 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 course topics. 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.

The exams will be conducted in-person.

Grading:

The following scale is used in determining the grades:

A	90 -100	C	64 - 67
A-	85 - 89	C-	60 – 63
B+	80 - 84	D+	56 – 59
B	76 - 79	D	53 – 55
B-	72 – 75	D-	50 – 52
C+	68 – 71	F	Below 50

Overall course grades will be computed using the following policy:

Programming Assignments	45%
Midterm Exam	25%
Final Exam	30%

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 BOTH (1) the average of Programming Assignments; and (2) the average of the exams (midterm and final exams).

At the end of the semester, an overall score of each student will be calculated according to the above policy. These scores will then be curved based on overall scores of all students in the class to determine the overall grades. Exams and averages of assignments may also be curved before this final curve is applied. 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 overall score. **Students are required to keep backup (soft) copies of all submitted work until after final grades are posted.**

Computers:

There are several computer options available for doing class assignments. Students may work on any school machine onto which CSC 133 software have been installed (see the "Resources" module of the "Home" section of Canvas for the list of labs that you can use for CSC 133) or on their personal machine provided that they install CSC 133 software.

As a part of CSC 133 software, the students must install CodenameOne (CN1). Information about CN1 can be found at https://www.codenameone.com. CN1 can be 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.** CSC 133 software is tested with Eclipse IDE 2022-03 R (version 4.23.0) available at https://www.eclipse.org/downloads/packages/release/2022-03/r/eclipse-ide-java-developers (on this page **do NOT hit the orange "Download x86_64" button** which downloads latest version of Eclipse, **instead, click on the "x86_64" download link** listed for your operating system to download Eclipse 2022-03 R). 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 **Java SE Development Kit (JDK) version 11** which is freely available at https://www.oracle.com/java/technologies/downloads/#java11. Please note that you need to create an Oracle account to download JDK.

Detailed instructions on how to install the above-mentioned software to a personal machine will be discussed when we cover "Introduction to Mobile App Development and CN1" lecture note slides. Later in the semester, the students will also be required to install JavaFX which will also be discussed in the lectures.

If you choose to work on the assignments on your personal machine (or a **physical** lab machine which is not available remotely), prior to submitting them, you are **required** to build and test them on one of the **remote** lab machines and then submit the files tested on the remote lab machine. Then, in the readme.txt of your submission, you must indicate the remote lab number and the name of the specific machine you have used in that remote lab to test your assignment. Since the assignments will be graded on a remote lab machine, this step will ensure the validity of your submission (e.g., if your personal machine is running Linux, or MacOS, this step will make sure that your assignments also work as expected on a remote lab machine which runs Windows). To build and test your assignment in the remote 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 remote lab workspace.

Communication:

Announcements will be made in class meetings and posted to the “Home” section of Canvas (under the “Announcements, Class Meetings, and Links to Lectures/Office Hours” module). In addition, it is assumed that messages sent to the “Communication Forum” located under the “Discussions” section of Canvas are received and read by you. Using this forum, you can also send questions or discussion items regarding course topics 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.

You can use “Subscribe” functionality of the forum and set your "Notification Preferences" (go to "Account" → "Notifications" → "Discussions" on Canvas) to receive a notification e-mail sent to your SacLink account whenever a new message is posted to the forum. If you do not have SacLink account, create one through https://mysaclink.csus.edu.

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

Recordings and Meetings:

The majority of CSC 133 lectures will be completely pre-recorded and hence, either we will not meet on the days of those lectures or if we will meet, questions about programming assignments or course topics will be discussed (i.e., flipped-classroom methods will be used). If the part of the (or the whole) lecture is not pre-recorded, we will conduct that part of the (or the whole) lecture in a class meeting. For the days that we will conduct a part of the lecture in a meeting, the remaining part of the lecture will be pre-recorded.

Recordings will be posted to the “Home” section of Canvas (under the “Announcements, Class Meetings, and Links to Lectures/Office Hours” module). **Each recording will be available to be streamed starting the day of the lecture (on either Tuesday or Thursday) at 9:00 am**. Please make sure that you watch the lectures in a timely manner (e.g., on the day of the lecture), which is very important for your learning.

Note that **you are not allowed to save a copy of the recordings. Also, you are not allowed to share the links of recordings and office hours with anyone else.**

Most weeks, we will meet only once a week (mostly on Thursdays). Some weeks (like the first week of instruction) we will meet both on Tuesday and Thursday. **All class meetings will be in-person. Our first meetings will be held on the lecture days of the first week of instruction (Aug/30/2022 and Sep/01/2022).** The other days that we will meet will be announced on Canvas. All meetings will begin at the scheduled class start times listed above.

Note that **you are not allowed to make audio or video recordings of the class meetings and office hours.**

Lateness/Attendance/Drops:

It is very important for students to attend all class meetings and to be on time. But attendance will not be taken.

Please inform yourself of the Department, College, and University policies on dropping courses. Policy information can be obtained from Computer Science Department office.

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 "Home" section of Canvas (under the "Ethics" module), for additional information. It is the responsibility of each student to be familiar with, and to comply with, the policies stated in these documents.

In class meetings and office hours, please contribute to an inclusive and respectful culture consistent with the Hornet Honor Code.

Campus Support:

Services to Students with Disabilities (SSWD) offers a wide range of accommodation services that ensure students with disabilities have equal access and opportunity to pursue their educational goals.

Student Health and Counseling Services (SHCS) staff are committed to continuing to provide exceptional service to our campus community. Though many students may be away from campus, most services are offered using secure remote technology.

Please self-diagnose if you are experiencing any COVID-like symptoms or have had exposure to someone who has tested positive for COVID. Contact SHCS at 916-278-6461 to receive guidance and/or medical care. The CDC provides a good source of information regarding COVID-19 and a way to self-check symptoms: https://www.cdc.gov/coronavirus/2019-ncov/index.html.

If you are experiencing challenges with food, housing, financial or other unique circumstances that are impacting your education, help is just a phone call or email away! The CARES office provides case management support for any enrolled student. Email the CARES office at cares@csus.edu to speak with a case manager about the resources available to you. Check out the CARES website.