CSCI 0076A - GAME PROGRAMMING

Catalog Description

Prerequisite: Completion of CSCI 12 with grade of "C" or better
Advisory: Completion with grade of "C" or better or concurrent enrollment in CSCI 13
Hours: 72 (54 lecture, 18 laboratory)
Description: Explore the algorithms, data structure, and techniques used to program computer video games. Emphasis on arcade-style video games (new and classic) written in Java. Topics include 2D animation, sprites, interaction, music, and sound. Underlying issues include graphical user interface programming, multi-threaded applications, realtime programming, use of sophisticated APIs, and societal impacts of computer gaming. (CSU, UC)

Course Student Learning Outcomes

- CSLO #1: Explain and apply basic 2D game concepts including the game engine, playing fields, sprites and events.
- CSLO #2: Explain and apply the game engine's update/render/draw loop and how this loop is utilized to implement a game.
- CSLO #3: Apply the basic 2D game concepts to design and implement a 2D game.

Effective Term

Fall 2020

Course Type

Credit - Degree-applicable

Contact Hours

72

Outside of Class Hours

90

Total Student Learning Hours

162

Course Objectives

Lecture Objectives:
1. Design a storyboard for a computer game.
2. Design and implement graphically-oriented computer programs based on written requirements and specifications.
3. Research and utilize third-party frameworks and Application Programming Interfaces to aid the programmer in developing computer games.
4. Evaluate game designs for user interface issues.
5. Evaluate game designs for gender and age issues.
6. Write a game design document at least five pages long that includes industry-accepted sections on game play, user experience, sample graphics, and flow.
Laboratory Objectives:
1. Design and implement correct algorithms for detecting the collision between two objects on screen.
2. Implement correct algorithms for simulating basic physical phenomenon: falling, bouncing, reflection, and collision.
3. Utilize object-oriented programming techniques (inheritance, extension, and the strategy pattern) to encapsulate game behavior.
4. Design and implement correct algorithms for directing on-screen animations, characters, and sounds in response to user input, such as keypresses and mouse clicks.
5. Design and implement correct algorithms for minimizing screen flicker and tearing.
6. Design and implement correct algorithms that utilize threads to permit concurrent on-screen actions.
7. Design and implement a complete 2D video game incorporating the following components: scoring, increasing difficulty levels, animation, sound effects, winnability (must be winnable), and user interaction.

General Education Information

- Approved College Associate Degree GE Applicability
- CSU GE Applicability (Recommended-requires CSU approval)
- Cal-GETC Applicability (Recommended - Requires External Approval)
- IGETC Applicability (Recommended-requires CSU/UC approval)

Articulation Information

- CSU Transferable
- UC Transferable

Methods of Evaluation

- Essay Examinations
  - Example: In an essay, define the term "double buffering" and describe how this technique can be used to eliminate "tearing" on the screen? Rubric Grading. Answer: Double-buffering allocates two graphical bitmaps in memory. One is designated the "back buffer" and is the one drawn upon by the rendering routines. The other is the "front buffer" and is the one being accessed by the display driver so it appears on the screen. During the video blanking phase, the two buffers are swapped. That is, the front buffer becomes the back buffer and vice-versa. By swapping the buffers during video blanking, tearing is eliminated.
- Objective Examinations
  - Example: Name two different design patterns.
- Problem Solving Examinations
  - Example: 1. Design and implement an algorithm to detect the collision between the two moving squares on the screen. Every time an object collides with another object the size of the object should shrink. Rubric Grading. 2. Write a Java applet which displays a picture. The applet should be 500 pixels wide by 400 pixels high. Your picture should involve at least two types of shapes (lines, rectangles, ovals, etc.) and at least two colors. You should use for-loops to draw multiple copies of the shapes, not just a lot of separate draw statements. For instance, you might draw 6 blue squares running down a diagonal and 5 horizontal green lines across the applet (using two for-loops). Completing the bare minimum program will earn you a B on this assignment. To get a higher grade, you should do something extra to show off. What you do is up to you, but it should demonstrate that you have thought more about your program. For this assignment, possibilities might include changing the sizes of the shapes

across the screen, drawing some fancier pattern, alternating colors, etc. Rubric Grading.

- Projects
  - Example: Design and implement a complete 2D video game incorporating the following components: scoring, increasing difficulty levels, animation, sound effects, winnability (must be winnable), and user interaction. The details of the project should be provided by the instructor. Rubric Grading.

Repeatable

No

Methods of Instruction

- Laboratory
- Lecture/Discussion
- Distance Learning

Lab:

1. Lecture with directed lab assignment. Example: simulated physics. "Real" physics are often too complex to program effectively and actually result in dull games. Simulated physics are easier to implement and are good enough to satisfy our senses. During lecture, the instructor presents equations and formulas for simulating common physical phenomenon, such as falling objects, collisions, and jumping. During directed lab, the students implement the algorithms in their computer programs. Each program is accompanied by a write-up describing which phenomenon are being simulated and how the simulation results in compelling game play. The simulations can be critiqued by the class as a whole. (Laboratory Objective 2)

Lecture:

1. Males and females have differing ideas about what makes games compelling. Age is also a factor. During the instructor lead lecture and discussion, several examples of existing games can be evaluated and critiqued. Student should discuss the topic in groups of there and then share it with the whole class. Prior to class, the students will have read a few case studies of gender and age differences in video games. (Lecture Objective 5)

Distance Learning

1. Instructor provides a video lecture on simulated physics. "Real" physics are often too complex to program effectively and actually result in dull games. Simulated physics are easier to implement and are good enough to satisfy our senses. During lecture, the instructor presents equations and formulas for simulating common physical phenomenon, such as falling objects, collisions, and jumping. As a lab assignment, the students implement the algorithms in their computer programs. Each program is accompanied by a write-up describing which phenomenon are being simulated and how the simulation results in compelling game play. The assignment will be uploaded to LMS for grading. (Lecture Objective 2; Laboratory Objective 2)

Typical Out of Class Assignments
Reading Assignments

1. Read chapters from the assigned college-level text books. Each chapter explains a different technique for programming computer games. Be ready to answer some questions during the lecture. 2. Read online case studies of usability, programming techniques, and game design. Be prepared to discuss in class.

Writing, Problem Solving or Performance

1. Design a simple video game using the techniques which we've discussed in class.Your game should be a two-dimensional, sprite-based, interactive game. The details of characters, layout, input, scoring, etc., are all up to you. You could design a brand-new game or describe a version or variation of some old favorite. 2. You should produce a program which involves both animation and user input. That is, there should be some part of the application which is moving or changing based on an animation loop (such as the ball in Pong) and some part which depends on user input (such as the paddles in Pong). You could have the user input directly control some shape on the screen, or you could have the user input do something like change colors or speeds of shapes in the animation.

Other (Term projects, research papers, portfolios, etc.)
Required Materials

- Killer Game Programming in Java
  - Author: Davison, Andrew
  - Publisher: O'Reilley & Associates
  - Publication Date: 2005
  - Text Edition: 1st
  - Classic Textbook?: No
  - OER Link:
  - OER:
- Beginning Java SE 6 Game Programming
  - Author: Harbour, Jonathan
  - Publisher: Course Technology
  - Publication Date: 2011
  - Text Edition: 3rd
  - Classic Textbook?: No
  - OER Link:
  - OER:

Other materials and-or supplies required of students that contribute to the cost of the course.