Subject:	GPM
Course Number:	150
Descriptive Title:	Coding for Creatives
Division:	Library and Learning Resources
Department:	Games and Playable Media
Course Disciplines:	Computer Science, Games and Playable Media
Catalog Description:	This course introduces students to computer programming through the context of digital art and interactive game programming. Students will learn the core concepts of computer science along with the necessary programming skills to create engaging visual and interactive user experiences. Emphasizing creativity and innovation, the course includes laboratory work in which students will develop their projects using an imperative programming language chosen by the instructor (e.g., C++, JavaScript, etc.). Through hands-on practice, students will gain practical experience and bring their creative visions to life. This course is designed for students with no prior programming experience.
Prerequisite:	Proficient in elementary algebra skills
Co-requisite:	
Recommended Preparation:	
Enrollment Limitation:	
Hours Lecture (per week):	2
Hours Laboratory (per week):	3
Outside Study Hours:	4
Total Course Hours:	90
Course Units:	3
Grading Method:	Letter Grade and Pass/No Pass
Credit Status:	Credit, degree applicable
Transfer CSU:	Yes
Effective Date:	
Transfer UC:	Yes
Effective Date:	
General Education ECC:	Area 4B - Language and Rationality: Communication and Analytical Thinking
Term:	
Other:	
CSU GE:	

Term:	
Other:	
IGETC:	
Term:	
Other:	
CalGETC:	
Term:	
Other:	
Student Learning Outcomes:	Upon completion of this course, students will be able to: 1. write proper algorithms. (Properly analyze a problem using top down design and write an algorithm that can be translated into computer code). 2. use the idea of abstraction to reduce the complexity of problems. 3. apply programming concepts to create interactive works of art and games.
Course Objectives:	1. Explain the fundamentals of computer graphics (pixels, screen coordinates, shapes, color, images). 2. Represent data utilizing simple numeric and character data types in a program. 3. Design programming solutions requiring decision-making using appropriate selection statements. 4. Design programming solutions requiring iteration using appropriate loop statements. 5. Design programming solutions requiring functions to manage complexity in programs. 6. Design programming solutions requiring lists (arrays) to manage data. 7. Apply programming concepts such as iteration and functions to generate digital artwork. 8. Instantiate a class object and access methods within that object. 9. Create and modify scripts within an existing game project. 10. Understand and use documentation of an existing Application Programming Interface (API).
Major Topics:	.**Fundamentals of Programming and Programming Languages (2 hours, lecture)** A. Use of the computer and computer languages B. Problem analysis C. Algorithms : meaning and purpose .**Digital Data (6 hours, lecture)** A. Numbers 1. Real 2. Integer B. Text 1. Characters 2. Strings C. Boolean values D. Pixels and color E. Shapes F. Images 1. Representation and display 2. Transparency 3. Translation/Scaling 4. Image Processing G. Audio .**Animation (3 hours, lecture)** A. Frames and The Run Loop B. Animating Objects

C. Sprites

IV. **Control Structures (6 hours, lecture)**

A. Branching with if

B. Nested if statement

C. Branching with if/else

D. Branching with if/else if/else

E. Iteration with while loops

F. Iteration with for loop

IV. **Functions (4 hours, lecture)**

A. Void functions

B. Functions with parameters

C. Functions with return values

D. Using published APIs

IV. **Arrays (5 hours, lecture)**

A. Declaring arrays

B. Getting/setting array data

C. Iterating through arrays

IV. **Object Oriented Concepts (6 hours, lecture)**

A. Classes

B. Member variables

C. Member function

IV. **Interactivity (4 hours, lecture)**

A. Event handling

B. Hit testing

IV. **Fundamentals of Programming and Programming Languages (3 hours, lab)**

A. tStudent generated art using programmatic statements

B. Students follow algorithm to reproduce an image

IV. **Digital Data (9 hours, lab)**

A. Numbers

1. Real
2. Integer

B. Text

1. Characters
2. Strings

C. Boolean values

D. Pixels and color

E. Shapes

F. Images

1. Representation and display
2. Transparency
3. Translation/Scaling
4. Image Processing

G. Audio

IV. **Animation (6 hours, lab)**

D. Frames and The Run Loop

D. Animating Objects

D. Sprites

XII. **Control Structures (9 hours, lab)**

A. Branching with if

B. Nested if statement

C. Branching with if/else

D. Branching with if/else if/else

E. Iteration with while loops

	F. Iteration with for loop XII. **Functions (6 hours, lab)** A. Void functions B. Functions with parameters C. Functions with return values D. Using published APIs XII. **Arrays (6 hours, lab)** A. Declaring arrays B. Getting/setting array data C. Iterating through arrays XII. **Object Oriented Concepts (8 hours, lab)** A. Classes B. Member variables C. Member function XII. **Interactivity (7 hours, lab)** A. Event handling B. Hit testing
Total Lecture Hours:	36
Total Laboratory Hours:	54
Total Hours:	90
A.1. Primary Methods of Evaluation (Part 1 - CCN courses only):	n/a
Primary Method of Evaluation:	2) Problem solving demonstrations (computational or non-computational)
Typical Assignment Using Primary Method of Evaluation:	The student will create a project that meets the following requirements: • A game engine is used to create an environment consisting of at least 3 animated sprites • Variables are used to maintain the state of at least 3 variables. Examples include, player score, health level, sprite position, username, etc. • At least one of the variable values must be displayed as onscreen text • The state of the onscreen text must be updated during the gameplay • The source code for the game must use at least one statement that demonstrates branching. During the demonstration of the game, at least two different branches must be executed.
Critical Thinking Assignment 1:	**Objective:** This assignment aims to enhance students' ability to critically analyze and debug code, fostering skills in identifying defects, understanding their impact, and applying appropriate solutions. It is designed to develop students' critical thinking and problem-solving skills in programming. **Description:** The student will be presented with the source code for an existing program containing some known defects. These defects may include issues like potential array bounds violations, uninitialized variables, and simple logic errors. **Tasks:** 1. Identify and document errors: 1. Review the provided source code and identify all errors present.

	2. For each error, specify: 1. **Line Number:** Indicate the exact line number where the error is located. 2. **Type of Error:** Describe the type of error (e.g., array bounds exceedance, uninitialized variable, logic error, etc.). 3. **Trigger Data Values:** Provide examples of data values that will trigger the error when the program is executed. 4. **User Scenario:** Describe a scenario in which the defect will become apparent to the user during program execution. 2. Apply and Explain Fixes: 1. Implement a viable fix for each identified error. 2. Explain the reasoning behind each fix, detailing why the chosen solution resolves the issue and how it improves the program's reliability or performance.
Critical Thinking Assignment 2:	**Objective:** This assignment aims to develop students' critical thinking skills by designing and implementing a branching narrative that dynamically adapts to user input. **Description:** Students will create an interactive story where the user's choices shape the narrative path, demonstrating their ability to analyze narrative structures and program branching logic. **Tasks:** 1. **Story Design:** Create a narrative where the user encounters at least three decision points, each influencing the story's progression. 2. **User Interaction:** Develop a program that displays individual screens or animated scenarios presenting users with one or more options at each decision point. 3. **Branching Logic:** Implement functionality that tracks user choices and determines the subsequent action or screen based on their input. 4. **Program Flow:** Ensure the program uses branching logic to dynamically control the flow of the story, providing a seamless user experience.
Other Evaluation Methods:	Homework Problems
If Other:	This is designed to be a project based course and the primary method for assessment will be done through computer programming assignments using graphics or game engines.
Instructional Methods:	Lab, Lecture
If other:	
Work Outside of Class:	Required reading, Skill practice, Study, Problem solving activity
If Other:	
Up-To-Date Representative Texts:	William Sherif. Elevating Game Experiences with Unreal Engine 5 Second Edition September 2022 Publisher(s): Packt Publishing ISBN: 9781803239866 Yu Zhang (Author), Mathias Funk. Coding Art: A Guide to Unlocking Your Creativity with the Processing Language and p5.js in Four Simple Steps Second Edition November 2023 Publisher(s): Apress Berkeley, CA ISBN: 9781484297797
Alternative Texts:	

Required Supplementary Readings:	
Other Required Materials:	**UP-TO-DATE REPRESENTATIVE GAME ENGINES** Playcanvas: A free fully-featured 3D game engine with a collaborative online editor using Javascript. Construct 3: An online game engine using Javascript. Unreal Engine: A full-featured, professional level game engine using C++.
Requisite	
Category	
Requisite course:	
Requisite and Matching skill(s): Bold the requisite skill. List the corresponding course objective under each skill(s).	
Requisite Skill:	**Recommended Preparation** / **Category and Justification** **Proficient in elementary algebra skills** / **Computational/Communication Skills**
Requisite Skill and Matching skill(s): Bold the requisite skill(s). if applicable	
Requisite course:	
Requisite and Matching skill(s): Bold the requisite skill. List the corresponding course objective under each skill(s).	
Requisite Skill:	**Recommended Skills** **Utilize variables in expressions, tables and graphs.** -Starting with a linear model in tabular, graphical or symbolic form, translate the model into the other two forms. -Recognize and apply the concepts of variable, expression, equation and function. **Plot points on a Cartesian plane.** -Graph linear equations and systems of linear equations by plotting points or by using intercepts and the slope. -Set up, graph, and solve linear equations, systems of linear equations, and linear inequalities using a variety of techniques.
Requisite Skill and Matching skill(s): Bold the requisite	

skill. List the corresponding course objective under each skill(s). if applicable	
Enrollment Limitations and Category:	
Enrollment Limitations Impact:	
Course Created by:	Mitch Middler
Date:	11/15/2024
Original Board Approval Date:	01/13/2025
Effective Term:	FA 2025