Las Positas College
3000 Campus Hill Drive
Livermore, CA 94551-7650
(925) 424-1000
(925) 443-0742 (Fax)

Course Outline for CS 46

GAME PROGRAMMING: 2D AND 3D

Effective: Fall 2009

I. CATALOG DESCRIPTION:
CS 46 — GAME PROGRAMMING: 2D AND 3D — 3.00 units

Want to Play? You have played plenty of games. Now it is time to create your own! Design, develop and test small 2D and 3D computer games using game development software tools such as Scratch, Alice, or similar programming development programs. This first programming course will provide the student with an understanding of the principles of game design, genre-specific design issues, storytelling, image manipulation, and development teams. Programming experience is not required to get started. Although this course has a programming focus, other topics briefly covered will include the history of computer/video game technology, game genres and design principles, and the social impact of games.

2.50 Units Lecture 0.50 Units Lab

Grading Methods:
Letter or P/NP

Discipline:

	MIN
Lecture Hours:	45.00
Lab Hours:	27.00
Total Hours:	72.00

II. NUMBER OF TIMES COURSE MAY BE TAKEN FOR CREDIT: 2

III. PREREQUISITE AND/OR ADVISORY SKILLS:

IV. MEASURABLE OBJECTIVES:

Upon completion of this course, the student should be able to:

A. examine the significant events and evolution in the development of the electronic game industry;
B. describe the basic principles of computer game programming;
C. list steps that are involved in computer game design and implementation;
D. design correct and efficient algorithms to satisfy given problem specifications;
E. use MIT’s Scratch (or similar) language to develop 2D computer programs;
F. use Carnegie Mellon’s Alice (or similar) platform to develop 3D computer programs;
G. construct small programs using various elements: such as variables, I/O, conditionals, loops, functions, expressions, and parameters;
H. debug programs with syntax, runtime, and logic errors;
I. use skills gained in the course to design and program an original game, story or animation;

V. CONTENT:

A. Programming Concepts
   1. the notion of an algorithm
   2. top-down design
   3. pseudocode
   4. flowcharts
   5. design of correct, concise, and efficient algorithms
   6. control flow and Boolean logic
   7. effective methods for checking and debugging programs
   8. history of computation and programming languages
   9. types and purposes of programming languages

B. Program Elements
   1. variables and data types
   2. operators, precedence and expressions
   3. input and output
   4. logic and conditional execution
   5. repetition (loops)
   6. functions and parameter passing
   7. recursion
   8. arrays

C. Game design and programming
   1. game genres (role-playing, first person shooter, racing, fighting, puzzle, turn-based, strategy, storytelling)

    2. storyboards and narrative elements
    3. level design
    4. game programming techniques
    5. careers in game programming
- D. Graphics and Animation
    1. mathematical preliminaries: points, distance, vectors,
    2. translation, rotation, polygons
    3. physical preliminaries: motion, collisions, inertia, gravity,
    4. force, mass, light and color
    5. drawing tools
    6. image formats and conversions
    7. bitmaps and color codes
    8. crating 3D models using Blender or Maya
    9. displaying movement
    10. animation tools
- E. Scratch Basics
    1. Introducing Scratch
    2. Getting Comfortable with the Scratch Development Environment
    3. A Review of the Basic Components of Scratch Projects
    4. Making the Kitty Dance – A Quick Scratch Project
- F. Learning How to Write Scratch Programs
    1. Moving Things Around
    2. Sensing Sprite Position and Controlling Environmental
    3. Settings
    4. Storing and Retrieving Data
    5. Doing a Little Math
    6. Conditional and Repetitive logic
    7. Changing the Way Sprites Look and Behave
    8. Spicing things Up with Sounds
    9. Drawing Lines and Shapes The Basics of BASIC
- G. Alice Programming Environment
    1. getting started with ALICE interface
    2. scenarios and storyboards
    3. using prebuild modesl
    4. orientation, resizing and movement instructions
    5. built-in functions and methods
    6. world-level vs object-level properties
- H. Game design and programming
    1. game genres (role-playing, first person shooter, racing, fighting, puzzle, turn-based, strategy, storytelling)
    2. storyboards and narrative elements
    3. level design
    4. game programming techniques
    5. careers in game programming

VI. METHODS OF INSTRUCTION:
- A. **Lecture** -
- B. **Lab** - Hands-on exercises in the laboratory
- C. **Demonstration** -
- D. **Discussion** -

VII. TYPICAL ASSIGNMENTS:

A. Programming assignments—create a bouncing-ball animation. Use collision detection and laws of physics to create a realistic simulation B. Write a recursive minimax algorithm to play tic-tac-toe or another turn-based game C. Script a role-playing game

VIII. EVALUATION:
- A. **Methods**
- B. **Frequency**
    1. Frequency
        - a. One homework assignment each week to cover that week’s discussion. This might be written or a programming assignment.
        - b. chapter quizzes, a midterm, final exam

IX. TYPICAL TEXTS:
1. Joel Adams *Alice in Action: Computing Through Animation.* 1st ed., Delmar/ Cengage Learning, 2007.
2. Pausch, Randy *Learning to Program with Alice.* 2nd ed., Course Technology, 2008.
3. Finney, Kenneth *3D Game Programming All In One.* 2nd ed., Delmar/ Cengage Learning, 2007.
4. Ford, Jerry *Scratch Programming for Teens.*, Delmar/ Cengage Learning, 2009.

X. OTHER MATERIALS REQUIRED OF STUDENTS:
- A. Computer and printer access
- B. Internet access