COMP2069 – Intro to Graphics Programming

Assignment 4

The Side Scroller

Due class #9 (Thursday July 11, 2013) @ midnight.

Value 15%

The sidescroller.py program **Maximum Mark: 49**

Overview: Use the mailpilot.py program from the lesson 7 code archive as a template. You will create an original side scrolling 2D arcade game (left to right or right to left). The game must have a **start screen** (that includes instructions for the user), the **main game screen**, and a game-end screen. The game must include **enemies** for the player to **avoid**. A **scoring system** must also be included. You must use your own graphic and sound assets (not the assets included with mailpilot.py). The use of classes and object oriented programming is highly encouraged to prepare you for the final project. You may choose a GUI framework such as Pygame, cocos2D, the GameEngine Library (described in Lesson 10) or others to create your interface.

Instructions :

(15 Marks: GUI, 15 Marks: Functionality, 4 Marks: Internal Documentation, 8 Marks: External Documentation, Version Control: 4 Marks)

1. Your application will have the following characteristics **(16 Marks: GUI, 16 Marks Functionality)**
    a. A **Start Screen**, allowing the user to get ready and displaying rules and instructions on how to win the game (2 Marks: GUI, 2 Marks: Functionality)
    b. A **Gameplay screen** where the main game occurs. The background will "scroll" in one direction only – either left or right. (2 Marks: GUI, 2 Marks: Functionality)
    c. A **Game-End screen** – this will display the player's final score and give the player the option to play again (2 Marks: GUI, 2 Marks: Functionality)
    d. Player control of an **Avatar** (a vehicle or character) – the mouse will be the main input device and control all basic movement (2 Marks: GUI, 2 Marks: Functionality).
    e. Computer control (AI) of the **enemies** (e.g. in mailpilot.py the enemies are the clouds. The Artificial Intelligence was very simple – a random distribution and direction of the clouds). The enemies should be abundant enough to challenge the player but not be impossible to beat. (3 Marks: GUI, 3 Marks: Functionality)
    f. Random placement of items to collect and/or obstacles to pass through or over (e.g. in mailpilot.py the player's plane would pass over islands) – this will generate points for the player (2 Marks: GUI, 2 Marks: Functionality)

g. A **Scoring system** – ensure that the player's score is accurately calculated and displayed somewhere on the **Gameplay screen** (1 Mark: GUI, 1 Mark: Functionality).
h. The player must have a **life counter** or **health status** that decreases each time his **avatar** collides with an enemy (1 Mark: GUI, 1 Mark: Functionality)
i. Add sound effects for collisions with enemies and collecting points (1 Marks: GUI, 1 Mark: Functionality).

2. Include **Internal Documentation** for your program **(5 Marks: Internal Documentation):**
   a. Ensure you include a program header for each module of your game that indicates: the Source file name, Author's name, Last Modified by, Date last Modified, Program description, Revision History (2 Marks: Internal Documentation).
   b. Ensure you include a headers for all of your functions and classes (1 Marks: Internal Documentation
   c. Ensure your program uses contextual variable names that help make the program human-readable (1 Marks: Internal Documentation).
   d. Ensure you include inline comments that describe elements of your GUI Design for your scrolling game (1 Marks: Internal Documentation)
3. Include **External Documentation** for your program that includes **(8 Marks: External Documentation)**:
   a. **A company Logo** (0.5 Marks: External Documentation).
   b. **Table of Contents** (0.5 Marks: External Documentation).
   c. **Version History** – ensure you include details for each version of your code (1 Mark: External Documentation).
   d. **Detailed Game Description** – describing how your game works (1 Mark: External Documentation).
   e. **Controls** (0.5 Mark: External Documentation).
   f. **Interface Sketch** – this section should include wireframes of each of your game screens with appropriate labels (1 Mark: External Documentation)
   g. **Screen Descriptions** – Include at least 3 screen shots for your game: 1 for your Start Screen, 1 for your Gameplay Screen and 1 for your Game-End Screen (1 Mark: External Documentation).
   h. **Characters / Vehicles** – Describe the character's Avatar (0.5 Mark: External Documentation).
   i. **Enemies** – Describe the computer controlled enemies and how they function (0.5 Mark: External Documentation).
   j. **Scoring** – Describe how the player can score and how the score is calculated (0.5 Mark: External Documentation).
   k. **Sound Index** – Include an index of all your sound clips (0.5 Mark: External Documentation).
   l. **Art / Multimedia Index** – Include examples of your image assets. Each image should be displayed as a thumbnail (0.5 Mark: External Documentation).
4. Share your files on **Github** to demonstrate Version Control Best Practices **(4 Marks: Version Control).**

a. Your repository must include **your code** and be well structured (2 Marks: Version Control).

b. Your repository must include **commits** that demonstrates the project being updated at different stages of development – each time a major change is implemented (2 Marks: Version Control).

Optional Game Features (i.e. Potential Bonus Marks).

A. Empower the computer controlled enemies to fire bullets.
B. Include a final "boss monster" to avoid.
C. Add power-ups for the player's **avatar** (e.g. extra speed, a shield)
D. Give the player's **avatar** the ability to kill or destroy the computer controlled enemies.
E. Add a cool soundtrack to the game.
F. Animate the player's **avatar** as he controls it (e.g. adds a walk-sequence, tilt it left and right).
G. Allow the game to scroll in any direction in response to the player's controls.

SUBMITTING YOUR WORK

Your submission should include:

1. An external document (MS Word or PDF).
2. A zip archive of your python project files. Please include all versions of your python code.

Please zip all files in to a single project archive.

Program Code & Functionality

Technical Evaluation

Display / User Interface	The Display / User Interface elements meet the program requirements. All text is spelled correctly and appropriate space is allocated for user input. Graphics & Icons are appropriate and match the program's functions.	16
Functionality	The program's deliverables are all met and the program functions as it should. No errors appear as a result of execution. User Input does not crash the program.	16
Internal Documentation & Readability	A program header is present and includes the name of the program, the name of the student, a short revision history and a short description of the program. All procedures and classes include headers that describe their functionality and scope. Inline comments are used to indicate their function when code is new or unclear. Variable names are contextual wherever possible.	5
External Documentation	An external document (MS Word or PDF) has been created that includes a company logo, table of contents, version history, detailed program description, a sketch of the GUI and screenshot (if applicable), and other details outlined in the template provided.	8
Version Control	**GitHub** is used to track App development. A **Commit history** will demonstrate the App being updated at regular points in time that correspond with the milestones of the project.	4
Creative Evaluation		**Mark**
Creativity	The program's GUI / UI is attractive. The programmer has added additional elements outside of the scope of the program that enhance functionality, usability and fun.	0
	Total (/49)	49
	%	100.0%

This assignment is weighted **15%** of your total mark for this course.

Late submissions:

- 10% deducted for each additional day.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:

1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.