**ICS2O FINAL PROJECT Spring 2018 – INDIVIDUAL WORK LOG**

My name: Matthew Ao Partner’s Name: Lucy Zhao

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **TIME SPENT** | **MY CONTRIBUTION TODAY** | **GOAL/PLAN FOR NEXT DAY** |
| May 22 | 1.5 hours | Thought up the idea for the game, established game objectives, style, controls, obstacles/enemies.  Initial proposal was completed today. | Work on Initial Design |
| May 23 | 1.5 hours | Started working on the initial design. | Continue working on Initial Design |
| May 24 | n/a | n/a | n/a |
| May 25 | 1 hour | Brainstormed key variables.  Established the screen dimensions.  Established controls to be used for controlling the player. | Work on Initial Design |
| weekend | 1 hour | Planned out in detail the mechanics of obstacles.  Worked on writing the Pseudocode with partner.  Wrote point counter system. | Finish Initial Design |
| May 28 | 1.5 hours | Completed initial design.  Created a python file for Floppy Birds game.  Coded Pygame window with WIDTH / HEIGHT dimensions, established game loop, set colour constant.  Defined basic code for blitting the penguin. | Code basic penguin blitting onto the screen |
| May 29 | 3 hours | Gathered all penguin images (walking, swimming, etc).  Defined pause game function and coded it into the loop.  Coded “gravity” (falling velocity that accelerates over time if the player is not underwater).  Coded basic jumping control (space bar). | Establish controls for penguin |
| **DATE** | **TIME SPENT** | **MY CONTRIBUTION TODAY** | **GOAL/PLAN FOR NEXT DAY** |
| May 30 | 5 hours | Created a class for the player (penguin) and defined initializing the class, setLocation, keys (controls), movement, blit (displaying the penguin), and execute (running the entire class).  Coded velocity for the penguin and realistic physics; it continues to slide for a certain amount in the same direction even after the key is released, depending on how high its velocity reached after acceleration.  Created the framework for collisions (so far this has only been implemented for when the penguin reaches x = 0 and x = WIDTH).  Created a darker transparent image that covers the screen when the game is paused.  Created a background. | Improve penguin physics, add water |
| May 31 | 4 hours | Added water to the game, and coded realistic physics for the penguin jumping out of the water and diving back in.  Added a defeat function (when the player drops below the screen, it is defeated and the game is paused); as well as a respawn function after the player is defeated.  Added a fullscreen function (press tab to toggle between fullscreen / non fullscreen).  Created stand-in code for new classes: levels, orca, mad mouth, and iceberg. | Finalize penguin mechanics |
| June 1 | 3 hours | Added new penguin animations for when the penguin jumps out and dives back into the water (after the falling velocity reaches a certain value, the penguin orientation is rotated to point downwards).  Added new penguin animations for diving underwater; after diving upwards / downwards reaches a certain velocity, the orientation of the penguin changes to point more upwards / downwards based on its direction.  Edited / added in more thorough commenting. | Condense code, fix bugs, and add sounds |
| weekend | 3 hours | Added new sounds function under penguin and added distinct sounds for different occasions: swooshing of water when the penguin emerges from the water, splashing when the penguin falls back into the water, and bubbling noises every few seconds when the penguin is swimming underwater.  Condensed the code to make it more efficient, edited comments / headings, and fixed bugs. | Work on player collisions with icebergs / platforms |
| June 4 | 2 hours | Created framework for the platforms class.  Created framework for the fish class.  Gathered all platform graphics files and organized them by name into the folder.  Continued commenting and adding in # end ifs and # end defs. | Continue working on player collisions |
| June 5 | 2 hours | Added the first platform to the level, started working on penguin / platform collisions.  Started coding different penguin physics for when penguin is above water as opposed to below water. | Continue working on player collisions |
| June 6 | 3 hours | Updated penguin mechanics underwater to be more realistic.  Continued defining platform collisions under class; furthered learning on object-oriented programming. | Add menus, work on player collisions |

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **TIME SPENT** | **MY CONTRIBUTION TODAY** | **GOAL/PLAN FOR NEXT DAY** |
| June 7 | 2 hours | Created more platforms.  Worked on player / platform collisions.  Added Lucy’s menus to the code, created a button function that took in several arguments, including” x, y, width, height, default colour, colour when hovering over it, the text it displays and its size. | Optimize code and debug |
| June 8 | 3 hours | Replaced all buttons from menus using button functions to condense the code.  Added button clicking sound.  Created different backgrounds for the menus + levels.  Worked on collisions.  Optimized code overall, cleaned up unnecessary code and debugged logic errors. | Modify, add to, and finalize menu screens and buttons |
| weekend | 8 hours | Added Start Screen, created flashing “PRESS ENTER TO START” message on start screen, added sound effect upon clicking Start.  Cleaned up menus and rerouted the pathways (clicking from one menu to another; setting the former to menu condition to False and the new one to True, etc.).  Redid + completed instructions pages in a photo editing software, as well as next / previous page buttons to flip through instructions pages.  Added a quit game button: upon pressing, the game loop stops and pygame.quit() is called.  Added a lock argument to the button function to prevent the button from being clicked when locked; coded the instructions next / previous page buttons to lock if there are no more pages left. | Debug and fix lag issues, finalize pause screen |
| June 11 | 4 hours | Worked on heavy debugging: spent large amounts of time reducing the lag when playing the game.   * Fixed the way images are loaded and converted (.convert\_alpha(), .convert()) * Stopped loading images to the screen in the loop and instead loaded them once in a variable beforehand, then only blit the image in the loop (based on a debugging print to console test, the program ran 12 times faster afterwards)   Added a standing penguin icon to the PLAY button on the main menu which changes to a jumping penguin upon hovering over the button.  Modified Pause Menu based on Lucy’s code; worked out the logic behind the resume, restart and main menu buttons.  For the restart and main menu buttons, several components of the player class had to be rewritten so that its setLocation function could be called to reset all the penguin properties (including its x, y coordinates, x, y velocity, whether or not it’s underwater, whether or not it’s falling, the current player penguin image being displayed, etc.). | Complete platform collisions |
| June 12 | 6 hours | Worked out player collision and implemented it for the first platform; the player can land on the platform after falling and it can jump back up on it from underwater.  The platforms were completed with the use of two classes: platform and platforms, where platform handles collision testing for any single platform, and platforms allows all platform objects to be added to a platforms list and to run through each of them in a loop to test for collisions (later realized a bug in the code only allowed the loop to run once, which was why collisions didn’t work for the other platforms). If any of them returned true for colliding with the player, the platforms class would set the player on top of the current platform, then return the current platform’s instance to the player class. The player class assigns the instance to its current platform argument as a boolean; if True, then the player is on a platform, whereas else, the player is not.  Added condition that reset the instruction page to the first page if the user leaves the instructions menu. | Fix issue with platform detection for other platforms, add fish to the game |
| June 13  PROJECT DUE | 7 hours | Created all platform images in photo editing software and added them all to the screen.  Fixed the error in looping through the Platforms list; now, the program will continuously check collisions for each of them (as long as they are above water).  Modified the player class movement physics to account for platforms. If when on a platform above water, the player would stop falling and be able to jump and walk on the platform. Otherwise, if when on a platform below water, the player would stop sinking, however would still be able to swim down as though the platform were in the background.  Also modified the player’s jumping mechanics to account for whether or not the player jumped from underwater, as opposed to whether the player jumped while on land. If on land, then the jumping motion and height is constant each time, and the player can only jump in small hops (like a real penguin). But if the penguin surfaces from underwater, then the jumping motion and height factors in the player’s y velocity so that the faster the penguin swims up, the higher it leaps after surfacing.  Added Lucy’s fish code to the game, placed it under two classes, individual Fish and Fishies, according to the same logic as with the platforms. | Complete the game functionality; add fish counter and win / lose conditions |
| June 14  EXTENSION | 7 hours | Modified and finalized fish classes. Modified the play level condition to account for the level number and append different number of fish set at different locations to the Fishies class depending on which level it is. Added argument to remove a fish upon player colliding with it (this is eating the fish). Along with this, added functionality of clearing all fish instances from the Fishies list and reading them if the player leaves or restarts any level. Added a fish counter that displays in the top right corner, displaying the number of fish left. If the player “eats” all the fish, the player wins the level  Added win and lose level conditions (menus that appear upon winning or losing). They’re essentially modified versions of the pause level menu; however, instead of having a button for resuming, created a next level button for the win menu and a locked button displaying “YOU LOST” for the lose menu. They still retain the restart and back to menu buttons.  Added a new level lock boolean variables to the level 2 and 3 buttons on the levels menu; if the user has never played those levels, they will remain locked and inaccessible. After playing a level (eg. 2), its button becomes unlocked in the levels menu. This functions even if the user exits a level halfway and returns to the level menu; the button to every level that the user has reached will be unlocked.  Implemented Lucy’s music and sounds to the program.  Added player walking animation for when player is walking on a platform above water.  Added lose conditions: if any of the fish swim past the edge of the screen, the user loses. If the user swims past HEIGHT or WIDTH, the user loses.  Added and modified headers + comments. | N/a |