Test Plan – Fishing Game

Introduction:

***Overall objective:*** The test will ensure that the game is free from input errors and all the game logic will function as intended. For example:

* If the player reaches highest score, there should not be more fishes to be caught and projectiles to attack the boat
* If the player loses his lives, there should not be fishes to be caught – simply the game ends even if the highest possible score is not reached
* The player should be able to choose the level

***Reference documents:***

* Programming Assignment Specification: “Fishing Game” (on Moodle)

Description of Test Environment:

* The game will be compiled and tested on a PC with Windows OS and Mac OS. The program used to develop and test the code will be Processing.

Stopping Criteria:

* Throughout the development process, visible errors will be fixed as the arise. Slight errors in the different objects behavior will be overlooked while testing for overall functionality. After playing a “level” of the game, those errors will be tweaked until they are appropriate.
* There will be a short testing period during which there will be attempts to “break” the game essentially, trying to do things that should be not possible in the game.
* The game will be deemed good enough to deliver after it is noted that the character behavior is consistently appropriate each time the game is run for at least 10 test-runs of the game as well as having the behavior of all visual components be consistently appropriate for all tests of the game.

Description of Individual Test Cases:

**Test 1:**

***Test Objective:*** Test whether *‘Splash Screen’* is displayed properly

***Test Description:*** When the game is started by the player, the *‘Splash Screen’* will be displayed giving instructions to the player how to play the game.

***Expected Result:*** On screen there will be a picture of the fisher and text instructions how the player should play the game.

**Test 2:**

***Test Objective:*** Levels work properly

***Test Description:*** If the player presses key 1 the game starts and the level is *‘Easy’.* Else if the player presses key 2, the game starts and the level is *‘Hard’*.

***Expected Result:*** If the player selects the *‘Easy’* level, there is displayed only one projectile (*Shark*). Else if the player selects the *‘Hard’* level, there are displayed 2 projectiles (*Shark* and *Bubble*).

**Test 3:**

***Test Objective:*** Test if all images appear correctly on screen

***Test Description:*** After the player chooses their level, the *‘Splash Screen’* disappear and all images appear on screen:

* *3 2D-Arrays of fishes*
* *Shark (and Bubble)*
* *Hook*
* *Lives*
* *Player’s Boat*

***Expected Result:*** The different objects are distinguishable from each other. All images appear on screen. Depending on the level there may be object from class *‘Bubble’*.

**Test 4:**

***Test Objective:*** Test if all text appears correctly on screen

***Test Description:*** When the game is started – *‘Score’* text; When game is over – text showing end of game

***Expected Result:*** When the game is started- on top left corner there appears the current *‘Score’* of the player; When the game is over:

* If the player achieves the highest score of 108, they win and on screen appears text *“You win! Press key ‘r’ to restart the game”*
* If the player loses all his lives, they lose and on screen appears text *“You lose! Press key ‘r’ to restart the game”*

**Test 5:**

***Test Objective:*** Test if object *‘Boat’* can be moved by player

***Test Description:*** Ensure that *‘Boat’* is visible and moves depending on the pressed arrow key.

***Expected Result:*** When the player presses the left arrow key the *‘Boat’* is moved left and vice versa. The boat doesn’t go off screen in either direction.

**Test 6:**

***Test Objective:*** Test if object *‘Hook’* can be moved by player

***Test Description:*** Ensure that *‘Hook’* is visible and moves depending on the pressed arrow key.

***Expected Result:*** When the player presses the down-arrow key, the *‘Hook’* drops. When the player presses the up-arrow key, the *‘Hook’* goes up. If the *‘Hook’* catches a *‘Fish’* it is restored automatically to its starting y-position. The *‘Hook’* x-position depends on the *‘Boat’* x-position and moves relatively with the *‘Boat’*. The *‘Hook’* cannot go above the *‘Boat’*.

**Test 7:**

***Test Objective:*** Test if *‘Fish’* moves correctly

***Test Description:*** Ensure that all the objects from class *‘Fish’* are visible and move following the preset path.

***Expected Result:*** All the objects from class *‘Fish’* move from the one side of the screen to the other without going off screen. When they reach one of the sides they go up and change their direction.

**Test 8:**

***Test Objective:*** Test whether object from class *‘Projectile’* hits the *‘Boat’*

***Test Description:*** Ensure that all objects from class *‘Projectile’* can hit the *‘Boat’* depending on their distance from it.

***Expected Result:*** If an object from class *‘Projectile’* is on a set distance from the *‘Boat’*, it hits the *‘Boat’* and the player loses a *‘Life’*. The object re-appears at the bottom of the screen at random x-position. If the object from class *‘Projectile’* doesn’t hit the *‘Boat’*, then it reaches the *‘top of the sea’* (a set coordinates), disappears from the screen and then re-appears at the bottom of the screen at random x-position.

**Test 9:**

***Test Objective:*** Test whether objects from class *‘Fish’* can be caught

***Test Description:*** Ensure that all objects from class *‘Fish’* can be caught if the *‘Hook’* is at a set distance.

***Expected Result:*** When an object from class *‘Fish’* is hit by the *‘Hook’*, the object disappears from screen.

**Test 10:**

***Test Objective:*** Player’s *‘Lives’* are displayed correctly

***Test Description:*** Ensure that all *‘Lives’* are displayed correctly on screen.

***Expected Result:*** When the player starts the game, they have 3 *‘Lives’* regardless of the selected *‘Level’*. If the *‘Boat’* is hit by an object from class *‘Projectile’*, a *‘Life’* is lost and it disappears from the screen. When the *‘Boat’* is hit 3 times, all the *‘Lives’* disappear. When the game is re-set the player has 3 *‘Lives’* again.

**Test 11:**

***Test Objective:*** Scoring

***Test Description:*** Ensuring that the *‘Score’* is calculated and displayed correctly on screen

***Expected Result:*** If the player catches an object from class *‘Fish’*, the *‘Score’* changes by increasing with 3, 5 or 10 points depending on the color of the fish that’s been caught. The maximum *‘Score’* is 108 points and if achieved, the game is over. If the game is re-set, the *‘Score’* is set to 0 automatically.

**Test 12:**

***Test Objective:*** Game re-setting

***Test Description:*** Ensure that when the game is over, it can be re-set by the player on pressing key ‘r’.

***Expected Result:*** If the game is over (player loses or wins), the player can press key ‘r’ to re-set the game. When key ‘r’ is pressed, the *‘Start Screen’* appears so the player can choose the level they want to play.