*Department of Computing, Faculty of Science*
*Macquarie University*

# COMP115 Introduction to Computer Science

Session 1, 2014

## Assignment Two: Swimmy Fish Part Two

**Due: 11:00am, Monday May 5, 2014 (Week Eight)**

**Worth: 17%**

### Learning Goals

This assignment contributes to the learning outcomes of COMP115 as follows:

• *Apply problem solving skills to develop algorithms that solve small to medium-sized computational problems*: Introductory experience in developing an algorithm to solve a problem from an informal specification of the problem.

• *Design and code implementations of their algorithms in an imperative programming language*: Practical experience in translating a simple algorithm into an equivalent Processing program that uses drawing, animation and variables.

• *Use standard software engineering practices to document, debug and test their programs*: Experience in developing a clear program from a specification, testing the conformance of the program to the specification, and debugging any problems that are detected.

### Introduction

This assignment asks you to write a program that extends the basic Swimmy Fish implementation that was the topic of Assignment One. You will add more complex behaviour using conditional statements and loops, as well as extend the drawing performed by the program. You should only need to use material up to and including Chapter Six of the textbook.

You can base this assignment on your submission for Assignment One if you wish. Alternatively, you can base it on the Assignment One sample solution.

There is a video on iLearn to show you generally how your assignment should look. As in Assignment One, there are some effects that are just to show you what is happening and should not be part of your code: the circles that appear when the mouse button is clicked, and the text captions that explain what is happening.

We strongly suggest that you divide the assignment up into pieces, tackle each piece separately and don't move on until you get each one working.

### Questions

If you have questions about the programming for this assignment, ask on the Assignment Two Forum on iLearn, consult with your tutor or ask Tony Sloane. No extensions will be granted except for serious and unavoidable disruption. Contact Matt Roberts as soon as possible, preferably before the due date, if you have suffered a disruption that affects your ability to submit this assignment.

### Part A: A More Interesting Fish (20%)

Assignment One drew a simple goldfish. Update your code to draw a more interesting fish. Keep the window size the same as for Assignment One. The assignment video shows you one possibility for fish appearance but we encourage you to be creative and come up with different colour schemes and animations.

You can choose the design as you see fit, but it must contain at least the following elements:

i) it must use at least two different colours and both of those colours should be different from the orange we used in Assignment One, and
ii) the fish must be animated in some way (i.e., it must change shape or colour as it moves across the screen).

### Part B: Placement (10%)

In Assignment One you could click anywhere on the screen to make a new fish appear. Make a change so that clicking only has this effect if the mouse button is clicked while the mouse is in the left half of the window and is not in front of the gap in the barrier (see Part D).

### Part C: Movement (30%)

In Assignment One the fish that are created always move straight across the window to the right. In this assignment that movement is still the default. However, in addition you should allow the user of the program to hit the UP and DOWN arrow keys to move the fish up or down. There will still be no way to move the fish left or right other than the default movement.

In response to an UP or DOWN key being pressed by the user you should:

i) move the fish directly up or down (as appropriate for the key that was pressed) a random number of pixels between 20 (inclusive) and 60 (exclusive), and
ii) 30% of the time instead of moving up when the UP key is pressed and down when the DOWN key is pressed, the fish should move in the opposite direction (e.g., move down when the UP key is pressed).

For information about using the UP and DOWN keys see the Processing Reference documentation for the keyCode system variable.

### Part D: Point Scoring (15%)

In this assignment the fish are not just swimming in an empty ocean. On the right-hand edge of the window you should draw a barrier that has a gap in its middle. (See the video for details.)

The aim of the Swimmy Fish game is to swim fish through the gap without touching the barrier. You should keep track of when a fish swims through the barrier and score one point in this situation. If a fish hits the barrier anywhere you should deduct one point. Display the current score in the top-left hand corner of the window as shown in the video.

For the purposes of detecting when a fish hits the barrier or goes off the window, you should base your check on the rightmost part of the fish to see if it crosses the barrier or window, plus the topmost and bottommost parts of the fish to see if they hit the barrier. The whole fish must pass through the gap in the barrier to score a point.

**Part E: Bubbles (25%)**

The fish in Assignment One didn't seem to be breathing. You should fix that by drawing bubbles coming out of the area where the fish mouth would be and rising in the water. You are free to design your bubbles how you like except that you must conform to the following rules:

i) there must be at least one and no more than four bubbles rising together,
ii) each time the fish breathes out the number of bubbles should be randomly generated, and
iii) you must use a loop statement to code your bubble drawing.

See the assignment video for an example of one possible acceptable bubble behaviour.

**Submission**

Before the due date and time, you must submit your Processing program online via the COMP115 iLearn site. You must submit your program as a single Processing source file called **ass2.pde**.

You can find the pde file inside the folder that stores your Processing sketch. Please rename it (if necessary) and submit just that file. Do not submit the whole sketch.

**Marking**

80% for the correctness of your code. We will check to see that your program correctly implements the specifications given above.

20% for the quality of your code. We will check to make sure that you have used reasonable names and types for your variables and that your code is presented in a manner that makes it understandable (e.g., good formatting, using comments to explain what your code is doing). Use the sample programs shown in lectures or the textbook as guidance.