COMP610 — Data Structures and Algorithms

Lab 01

What to do

Do all the tasks, and answer all the questions. For code, make sure you follow the Code Laws. You can work individually or in pairs. Record non-code answers, as you may have to give open-class feedback on them.

Question 1

Download and extract `lab01.zip`, and open it in Eclipse. Read the `IntPair` class, and answer the following questions:

1. Which package is `IntPair` in?
2. The class itself is divided into four sections by comments. What are the names of those sections?
3. What are the fields of the class?
4. When are two `IntPair`s equal to each other, according to the `equals` method?
5. How many factories does the class have? What is the difference between them? Describe *in your own words*, without referring to the code in your description.

Read `IntPairTest` and then run it (right-click the file in the Package Explorer, then go 'Run As', 'JUnit Test') — everything should pass. Then answer the following questions:

1. How many tests are there?
2. Describe, in one sentence, using your own words, what each test is supposed check. Avoid using code descriptions where possible.
3. Are there any methods in `IntPair` that don't have corresponding tests? Why do you think that is?

Question 2

You will need to implement a `StringPair` class. It works identically to the `IntPair`, but its sides are `String`s, rather than `int`s. Ensure that you follow the Code Laws, and use `IntPair` as a basis. Be careful of how you write `equals`, as the elements of a `StringPair` are objects.

When you are finished, run `StringPairTest` — all tests should compile and pass.

Question 3

If we had to write a new `Pair` for every type, we would never get any work done. To avoid that, you will implement a `MonoPair` class. This is the same as the previous two pairs, except that its two sides are of a generic type `T`. Ensure that you follow the Code Laws — be *especially* careful of the factories and `equals`, as they're tricky.

When you are finished, run `MonoPairTest` — all tests should compile and pass. Then answer the following questions:

1. What is a big limitation of `MonoPair`? *Hint:* Consider whether you can define a pair whose left and right elements are of different types.

Question 4

You will need to implement a `Pair` class. It works identically to `MonoPair`, except it has two generic types — `L` for its left element, and `R` for its right element. Ensure that you follow the Code Laws — again, the factories will be trickier than you think (especially `newLeft` and `newRight`).

When you are finished, run `PairTest` — all tests should compile and pass.