School of Computer Science
University of Birmingham

Worksheet 1

MSc/ICY Software Workshop 1

Assessed Exercise: 5% of the continuous assessment mark.

Submission: Thursday 24 October 2013 12:00 noon

5% late submission penalty per 24 hours or parts thereof. No submission after 24 hours.
Tests and comments are mandatory. Submit your `.java` files via Canvas as a single zip file.

Exercise 1: (Basic, 30%) Define a class `ComputerAccount` and a constructor to create it. A computer account should be built from the three strings `realName`, `userName`, and `password`. It should implement methods `printRealName`, `printUserName`, `printPassword`, and `changePassword`. The first three methods take no arguments. The last method takes one argument of type `String`, the new password, and sets the password to the new password; its return value is `void`.
Note that you have always to comment and test your programs appropriately, not just for this exercise on this worksheet. We will not write this to the exercises in future, but still if you fail to do so marks will be subtracted.

Exercise 2: (Basic, 20%) Define a class `Student` as well as a constructor and getter methods. A `Student` should be represented by his/her name, his/her Student ID, and the Degree Programme, each of which is a String.
Also write a method `toString` method that returns a `String` in the following format: `"[John Smith, ID: 1111111, ICY]"`.

Exercise 3: (Medium, 20%) In exercise 2 of Worksheet 0, you wrote a program that converts masses given in the imperial system into the metric system. Write a `Java`-program that can deal with weights given in pounds. Make use of the conversion:

$$1 \text{ pound} = 0.45359237 \text{ kilograms}$$

Define a `Weight` class, and write a constructor `Weight(double p)` to generate a weight given in pounds. Implement methods `getPounds` and `getKilograms` which return the weight in pounds and kilograms, respectively.

Exercise 4: (Advanced, 20%) In Exercise 3 of Worksheet 0 we looked at addition and multiplication of two fractions. In the current exercise fractions should be represented by a `Java` class `Fraction`. You have to define this class, which in addition to the constructor and the accessor methods `getNumerator` and `getDenominator` has methods `toString` (to print a rational number) as well as `getSum`, and `getProduct`, which generate the sum and the product of fractions and return the corresponding object of Class `Fraction`. For instance, if we generate fractions `Fraction f1 = new Fraction(1,2);` and `Fraction f2 = new Fraction(3,7);` then

- `f1.toString()` should return the string `"1/2"`;
- `System.out.print(f2.getProduct(f1));` should print `3/14` (which corresponds to the product of `f2` and `f1`); and
- `System.out.print(f2.getSum(f1));` should print `13/14` (which corresponds to the sum of `f2` and `f1`).

Exercise 5: (Advanced, 10%) Create a class `FixedBankAccount` with constructor `FixedBankAccount(float initialCapital, int d)` where the `initialCapital` is left for `d` days on the account and then the total capital is returned. The annual interest rate is `interestRate` of type `float` (e.g., `interestRate = 0.01;`). The daily interest is a 365th of this. (We do not consider leap years.) The interest is added to the capital after each full year. Write a method which computes the total capital after `d` days have elapsed. (Make use only of programming constructs introduced in the lecture, that is, not of loops or conditionals. Test your program for `interestRate = 0.01`, `initialCapital = 1000.00`, and `d = 1000`.)