

Week 10 Practical Class Exercises

Introducing Java

Objectives

The aim of this class is to help you learn to use the Java compiler and run-time system, a bit of DrJava if you're interested and to introduce some common error messages produced by the Java compiler.

You will also write and exercise some simple Java classes.

Exercise 1 (Using the Java System)

To get you started, you should compile and run some of the Java programs we looked at in lectures. Copy the files `RestRate1.java`, `RestRate1P.java`, `RestRate2.java` and `RestRate2P.java` from the *Lectures* page of the course web site (<http://cs.anu.edu.au/Student/comp1100/lects/java/code/>) and in `/dept/dcs/comp1100/public/labs/week10/`.

The Java compiler can be invoked at the command line like this:

```
javac RestRate1P.java
```

If successful, it will produce a file called `RestRate1P.class` which can be run on the JVM (Java Virtual Machine) like this:

```
java RestRate1P
```

(Don't type the `.class` suffix.) Note that `RestRate1` and `RestRate2` are not complete programs so they cannot be run independently. However, they can be loaded into **DrJava** and exercised directly. Just type `drjava` at the console to start it up.

Exercise 2 (Errors)

There is a simple Java program `HelloAge.java` at Week 10 of the *Practicals* page of the course web site and in `/dept/dcs/comp1100/public/labs/week10/`. It contains a number of simple syntactic errors. Attempt to compile `HelloAge.java` to see the error messages produced. Correct the errors (one by one) and repeatedly attempt to compile it until there are no remaining errors. Once it is free of syntactic errors, carefully read the program to understand what it does, before running it with the command `java HelloAge`.

Exercise 3 (Bank Account)

Add a method

```
void addInterest(Double rate)
```

to the `BankAccount` class discussed in lectures and Chapter 3 of the textbook. Calling the method should add interest at the given *percentage rate* to the balance of the account. For example, after the statements

```
BankAccount clemStreamline = new BankAccount(1000.0);  
clemStreamline.addInterest(10.0); // 10% interest
```

the balance in `clemStreamline` should be \$1100.

The original `BankAccount` class can be found through the *Lectures* page of the course web site. All of the code from the *Big Java* textbook is also available through the *Java* page of the course web site. (In this case, you will be interested in Chapter 3.)

Exercise 4 (Savings Account)

Write a class `SavingsAccount` that is very similar to the `BankAccount` class, except that it has an additional field, `interest` representing the percentage interest rate for this account. Supply a constructor that sets both the initial balance and the interest rate. Supply a method `addInterest` (with no explicit parameter) that adds interest to the account.

Write a test program that constructs a savings account with an initial balance of \$1000 and an interest rate of 10%. Then, call the `addInterest` method five times before printing the final balance.

Show your tutor your completed work.