CMP-5015A Programming 2

Week 12: Lab Exercises

Learning Outcome:
- being able to serialise an object in Java, to save it and load it
- understand the notion of reflection and be able to use basic build-in methods

About the labs...

Lab sheets. Every week, the lab sheets will propose exercises from easy to difficult ones. You do not have to do them all but make sure you understand well the concepts that have been covered that week. For the most difficult exercises and the ones written optional, treat them as a challenge on which you can work on over several weeks (even as a group) and don't rush to check the solution.

My advice. Try by yourself first, before asking for help or looking at the solution. The best way to become a good programmer is to practise and code, even if sometimes it is a bit of a headache. Teaching assistants and lecturers are here to help (and very happy to), but we will try to make you understand your mistake by yourself, rather than feeding you the solution. Your code is not compiling? Check what the error message says. It is not running as expected? Try to print within the code, to see where it goes wrong. And don't copy-paste solutions... What's the point?

Another advice is to make sure you understand the basics of programming (in whatever language) and have solid foundations before trying to do more advanced fancy things. This is an interesting read:
https://blog.codinghorror.com/why-cant-programmers-program/
So, make sure you can do the "easy stuff" before going for the more difficult ones.

Serialisation

Exercise 1

You can do this exercise on your own or by exchanging file with a friend in the lab (or with a TA, Gavin or Laure!).

1. Create an ArrayList of strings with the message of your choice, serialise it and save it as a .ser object. Send this file to a friend for them to read it and reply with a .ser file.
2. When you receive your friend .ser file, load it into an Object reference and print it out (the entire ArrayList).
3. To read it better, cast the object to Iterable and print each string by calling iterator().

Exercise 2

1. Create a class `Student` with two fields `course` and `name` of type `String` and a field `id` of type int. Make your fields private, provide getters, setters and constructor.
2. Create a class `Module` with field `students` which is an ArrayList of students and an int fields `numStudents` initialised at 0. Make sure Module is serialisable - but without a unique identifier (i.e. no serialVersionUID).
3. Implement a constructor that creates a few object students of your choice, add them to the arrayList students and update numStudents to the number of students you have created.
4. Create a Module object using this constructor. Try saving and loading this. (You might need to do something about the class Student, but what?)
5. Print the module and the students in the module before and after saving/loading. What do you notice? What about the fields in the students?
6. Comment the line where you save the file and add a private field to Module. Compile again the Main. What happens? (this should crash)
7. Add a serialVersionUID to Module. Comment the field you had added. Uncomment the line where you save the file in the Main and compile. Now, comment again the line where you save the file, uncomment your added private field. Compile. What happens? (this time is should not crash)
8. Make one of the field of Module transient and check the property of a transient field.

Reflection

Exercise 3

Download from BlackBoard the HangManGame class (try to figure out how it works if you want) and put it in the src folder of your project. Download also the object1.ser, object2.ser and object3.ser files and put them in the root directory of your project.

1. Use the class Class to determine what the three objects are. Do this one at a time. One will crash. Why?
2. Print out all the public and private fields for the class of object stored in object1.ser.
3. Find out a way of determining whether the field is public, private or protected.
4. Print out all the methods and find out if they are public, private or protected.

More challenging

Exercise 4

- Just like we did in the lecture with Heads or Tails, implement your own version of a game such as Tic-tac-toe or Rock-Paper-Scissor.
- Only using object1.ser and the methods built-in for reflection, can you play a game of HangManGame?