Prof. Jingke Li (FAB120-06, lij@pdx.edu), Class: TR 12:00-13:15 @ KMC 185; Office Hour: TR 11:00-11:55.

Assignment 1: Getting Started with Scala

(Due Thursday 10/4/18)

This first assignment is to get you started with the programming language Scala, which we'll be using in future assignments. Refer to this week's lecture notes for how to use Scala on the CS Linux system.

Unzip the file `assign1.zip`, and you'll see this handout file, `assign1.pdf`, and two program files: `Hello.scala` and `qsort.c`.

1. Scala

To most of you, Scala probably is a new language. No worries. Scala is an easy language to learn, especially if you've already been programming in an object-oriented language, such as Java or C++. The official website, `www.scala-lang.org`, has some very good introductory materials and tutorials, all under the "Documentation" heading. You may want to start with the following:

- Read a few beginning sections of "Tour of Scala."
- If you are a Java programmer, read "Scala for Java Programmers."
- If you want to install Scala on your own laptop or PC, read "Getting Started."
- The "Cheatsheet" is handy for a quick check on Scala's syntax forms.

2. Running Scala

Scala is available on the CS Linux system (`linuxlab.cs.pdx.edu`). Check that you have it in your environment, and check its version:

```
linux> scalac -version
```

The default version should be 2.11.12. There is a newer version available, 2.12.3. You may switch to use the new version, although for this course it does not make a different. To include the new version in your environment, you need to run `addpkg`, and select both scala and java8.

The Scala package comes with a compiler `scalac` and an interpreter `scala`. To run a program with the compiler, you compile it first (into `.class` files), then run it:

```
linux> scalac Hello.scala
...
linux> scala Hello
Hello, world!
```

Note that even though the compiler's output are legal Java class files, you need to use the command `scala` to run, not the command `java`, since Scala's runtime libraries are needed.

To run a program with the interpreter, you start the interpreter shell, then load the program and invoke its `main` method to execute it:

```
linux> scala
...
scala> :load Hello.scala
...
scala> Hello.main(Array(""))
```

Note that the command for invoking the interpreter is the same as for running a compiled program, `scala`.

Exercises

1. Create (or copy from somewhere) a Scala "Hello, world!" program. Execute it with both the compiler and the interpreter.
2. Practice with the interpreter's REPL (Read-Eval-Print-Loop) shell. Type in some definitions, and see the immediate results.
3. Try to write a new version of the "Hello, world!" program. The new program will read in a name from the command-line, and replace the word "world" by the name in the printout message. Here is a sample run:

```
linux> scalac Hello2.scala
...
linux> scala Hello2 John
Hello, John!
```

3. Quicksort

The file `qsort.c` contains a quicksort program written in C. It reads in a parameter `N`'s value from the command-line, and creates an `N`-element array holding a random permutation of integers, `1 .. N`. It then invokes quicksort on the array, switching to bubblesort when an array section's size is below 10. At the end, it verifies the results and prints out the sorted array. Here is a sample run:

```
linux> gcc -o qsort qsort.c
linux> ./qsort 10
Initial array:
7, 1, 3, 9, 2, 8, 5, 10, 4, 6,
After quickSort:
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
Result verified!
```

Your Tasks

1. Read and understand the program; pay attention to the program structure. Compile and run it, with different `N` values.
2. Write a *corresponding* quicksort program in Scala; call it `Qsort.scala`. Your program must follow the exact program structure of the C version. Specifically,
   - it should have the same set of functions; and each function should have the same set of parameters (execept for `main`, where Scala has only one parameter while C has two)
   - it should read in the parameter `N` from the command-line
   - it should create a randomized permutation of `1 .. N` for the initial array
   - it should print out the result in the same format

The whole program should be wrapped inside an object, `Qsort`.

Some Hints The following are some of the issues you need to handle:

- Syntax for functions and loops
- Command-line input

- Random number generator

Resolve these issues by searching and reading related documents.

4. Submission and Grading

For this assignment, you only need to submit the `Qsort.scala` program. Include your full name in a comment block and place it at the top of the program file. Submit your file through the "Assignment 1" submission fold on the D2L class website (under the "Activities/Assignments" tab). Keep your original file untouched in case there is a need to show its timestamp.

Grading will be based on both your program's correctness and its conformance to the specified requirements. (See "Your Tasks" paragraph above.)