INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Big Data Storage and Processing

TP: Scala

The goal of this TP is to get you familiar with the Scala programming language.

1. Scala installation

Download the latest version of Scala **binaries** from https://www.scala-lang.org (>Download>Other ways to install Scala). Unpack the archive to a directory of your choice (in your `$HOME` for instance) and add the following lines to your `~/.bashrc` file in order to be able to execute Scala from anywhere:

```
export SCALA_HOME = <your SCALA directory>
export PATH = $PATH:$SCALA_HOME
```

Don't forget to make the modifications persistent:

```
$ source ~/.bashrc
```

and to test your Scala installation:

```
$ scala –version
```

2. Using the Scala REPL

The Scala REPL (Read-Eval-Print Loop) is an interactive tool for evaluating expressions at the prompt. Previous results are automatically imported into the scope of the current expression as required. To start the REPL simply type:

```
$ scala
```

Exercise 2.1

Create a list (of integers) and test whether the list is a palindrome.

Exercise 2.2

Print the list “rotated” left (e.g. 1,2,3 -> 2,3,1).

Hint:

- `Lists` have a method head, that returns the first element of the list. E.g:

```
scala> List(42,23,5).head
res0: Int = 42
```

- `Lists` also have a method tail that returns a `List` (of the same type) containing all elements except for the first one. E.g:

```
scala> List(42,23,5).tail
res1: List[Int] = List(23, 5)
```

3. Running standalone applications

You can edit your Scala applications in:

- **IntelliJ** - you will need the Scala SDK. Follow the steps from here: https://docs.scala-lang.org/getting-started-intellij-track/getting-started-with-scala-in-intellij.html
- **Eclipse** - you will need the Scala IDE. You can either:
    - Install it as a plugin from the marketplace (Help >> Eclipse Marketplace) to your existing Eclipse;
    - Install it as a separate, Eclipse for Scala, from: http://scala-ide.org

For the following exercises fill in the code in the `TP.scala` source file.

Exercise 3.1

Write a recursive function `rotate_n`, that takes a `List` of integers and an integer n as parameters, and rotates the list n times.

Exercise 3.2

Assume we have the following `Map`:

```
val fruit_to_color: Map[String ,String] =
Map("banana"->"yellow",
    "blueberry"->"blue",
    "cherry"->"red",
    "lemon"->"yellow",
    "kiwi"->"green")
```

Write a function that returns a `Map` that maps each value in the original map to a list of keys. The function should behave as follow:

```
scala> reverse(fruit_to_color)
res0:
scala.collection.mutable.Map[String,List[String]] =
Map(yellow -> List(lemon, banana),
    green -> List(kiwi),
    red -> List(cherry),
    blue -> List(blueberry))
```

Hint:

- The Scala functional methods for `Collections` might be useful (e.g. `groupBy, mapValues`)