CS320 Programming Languages
README

We use the programming language `Scala` to implement interpreters of various programming languages in this course. All the programming exercises and projects run on `sbt`, which is the interactive build tool for `Scala`.

1 Installation

First, install `Scala` and `sbt` by following the instruction in the site:

`https://www.scala-lang.org/download/`

You can use `IntelliJ` or other IDEs but we recommend you to use terminal to build and run the exercise project. How to use `sbt` in termainal is described in Section 3.

2 Structure

The exercise project consists of the following directories and files:

```
cs320
├── build.sbt
├── project
├── README.pdf
└── src/main/scala/cs320
    ├── Main.scala
    ├── Homework.scala
    └── ex01
        ├── Exercise01.scala
        ├── ex01.pdf
        └── package.scala
```

The `build.sbt` and `project` files are related to the `sbt` configuration. The `src/main/scala/cs320` directory contains all the source codes of the exercise. The `Main.scala` file is the top-level source code to evaluate tests of each exercise. The `Homework.scala` file provides common utility functions that will be used in exercises. The `ex01` directory contains all the information of exercise #1.

From now, we will give you a programming exercise as the directory `exXX`, such as, `ex01` or `ex05`. Then, you just copy and paste it into the `src/main/scala/cs320` directory. Each programming exercise consists of three files: `exXX.pdf` explains what you should do, `ExerciseXX.scala` describes the specification, `package.scala` is the main file that you should implement.

3 `sbt` Console

You can build and run your implementation using `sbt` console system. On the top of the project directory(i.e. `cs320`), just type the `sbt console`:

```
$ sbt "console"
[info] Loading global plugins from .../.sbt/0.13/plugins
[info] Loading project definition from .../project/cs320-exercise/cs320/project
[info] Set current project to cs320 (in build file:.../cs320/)
[info] Starting scala interpreter...
[info]
Welcome to Scala 2.12.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_152).
Type in expressions for evaluation. Or try :help.

scala>
```

If you want to load Exercise #1, then just type `import cs320.ex01._`. Then, you can use any functions or variables defined in `ex01/package.scala`. If you want to test some functionalities, we recommend you write some test cases in the `tests` function using `test` or `testExc` functions. The `test` function checks whether given two arguments are same and `testExc` function checks whether the first argument throws an error with a message containing the second argument string. If it fails, it prints out which tests are failed and their positions based on line numbers. We recommend you write enough test cases before implementation. It helps you consider the corner cases and organize your thoughts.

```
scala> import cs320.ex01._
import cs320.ex01._

scala> test(1,1)
PASS [<console>:15]

scala> test(1,2)
FAIL [<console>:15]: 1 is not equal to 2

scala> testExc(error("abcd"), "a")
PASS [<console>:15]

scala> testExc(error("abcd"), "e")
FAIL [<console>:15]: "abcd" does not contain "e"

scala> tests
FAIL [package.scala:26]: Not yet implemented
FAIL [package.scala:27]: Not yet implemented
FAIL [package.scala:28]: Not yet implemented
FAIL [package.scala:29]: Not yet implemented
FAIL [package.scala:30]: Not yet implemented
FAIL [package.scala:31]: Not yet implemented
FAIL [package.scala:32]: Not yet implemented
FAIL [package.scala:35]: Not yet implemented
FAIL [package.scala:36]: Not yet implemented
FAIL [package.scala:38]: Not yet implemented
FAIL [package.scala:39]: Not yet implemented
FAIL [package.scala:42]: Not yet implemented
FAIL [package.scala:43]: Not yet implemented
FAIL [package.scala:44]: Not yet implemented
FAIL [package.scala:47]: Not yet implemented
FAIL [package.scala:48]: Not yet implemented

scala>
```

You can also directly execute tests for a certain exercise using `run` command.

```
$ sbt "run ex01"
[info] Loading global plugins from .../.sbt/0.13/plugins
[info] Loading project definition from .../project/cs320/cs320/project
[info] Set current project to cs320 (in build file:.../project/cs320/cs320/)
[info] Running cs320.Main ex01
FAIL [package.scala:26]: Not yet implemented
FAIL [package.scala:27]: Not yet implemented
FAIL [package.scala:28]: Not yet implemented
FAIL [package.scala:29]: Not yet implemented
FAIL [package.scala:30]: Not yet implemented
FAIL [package.scala:31]: Not yet implemented
FAIL [package.scala:32]: Not yet implemented
FAIL [package.scala:35]: Not yet implemented
FAIL [package.scala:36]: Not yet implemented
FAIL [package.scala:38]: Not yet implemented
FAIL [package.scala:39]: Not yet implemented
FAIL [package.scala:42]: Not yet implemented
FAIL [package.scala:43]: Not yet implemented
FAIL [package.scala:44]: Not yet implemented
FAIL [package.scala:47]: Not yet implemented
FAIL [package.scala:48]: Not yet implemented
[success] Total time: 2 s, completed Sep 3, 2019 1:31:13 PM
```

Moreover, the triggered execution is helpful for continuous compilations. You can make a console run when certain codes change by prefixing the command with `~`. Monitoring is terminated when `enter` is pressed.

```
$ sbt "~console"
...
$ sbt "~run ex01"
...
```

4 CAUTION

There are several rules you should follow:

- DO NOT use mutable variables; `var x = ...`
- DO NOT use mutable data structures; `scala.collection.mutable.Map`
- DO NOT use loops; `while(...){...}` or `for(...){...}`