Functional Programming with Scala

Homework 5

Instructor: Dr. Ionut Cardei

Chapter 8. Property-based testing
Chapter 10. Monoids

Read the textbook chapter and the **lecture notes posted online** for material and examples not covered in the textbook.

Make sure you follow exactly all requirements listed in this document to get full credit.

Academic Honesty

For every assignment, including this, you have to work alone. Your submission must be your original work, that was not shared with anybody else. It is not enough to rename identifiers and move code around. Your solution for non-trivial answers (e.g. one liners) must be substantially different from any other.
Note that it is easy for the grader to find code or answers taken from online sources or to find solutions that are too similar.
Also, do not copy text (like examples, definitions) from notes/textbook or other sources, unless explicitly required to do so.
When in doubt, please contact the instructor at icardei@fau.edu.

Coding Requirements

For this assignment we can use Intellij IDEA to write the code. Emacs also accepted.

Don't forget to read the **Important Notes** at the end of this file.

This homework has 3 problems (one for extra credit).
Total score: 100 points + extra credit.

Homework Formatting and Delivery

1. Start a new Word document called h5.docx. At the top write a title and your name.
2. Write the text answers for the problems in the given order (e.g. 1, 2, 3,...). Include the Scala code from your source files where applicable. Do not reorder solutions in the document.
3. Although syntax highlighting is not required, it is very appreciated by the grader, as it makes your code more readable, hence more likely to prevent grading mistakes. Paste directly from IDEA or check out http://hilite.me.
4. Make sure the code builds and runs correctly.

5. Convert the Word file to PDF.
6. Upload the PDF file with the answers and the Scala files on Canvas.

Follow strictly all requirements to get 100% credit. E.g. if a problem says: “use function foldRight” then your solution must use function foldRight to get full credit.

Problem 1.

Write the solution in a file called p1.scala, in an object called p1.
a) Write a function *isSorted* that checks that an *IndexedSeq* object is in sorted order using a function argument *less: (A,A) => Boolean*.

b) Consider the *quicksort* function you wrote for Homework 4. Define a set of properties for the quicksort function, including checking for ascending order using the *isSorted* function from part a). Address all cases: empty vectors, vectors of one element, and vectors of more than one element. Also, compare results with the standard library *sorted* method.
Write a program that tests those properties using the Prop/Gen classes from the Gen.scala file.
The solution is graded also on the correct definition of properties.

(Hint: Gen.listOf generates lists and quicksort works with IndexedSeq. One could define a Gen and a sized generator for Vector or just convert back and forth between List and Vector.)

Problem 2.

Write the solution in a file called p2.scala, in an object called p2. Click for file Monoid.scala.
a) Write a function with this signature:

```
def getMinMax[A](v: IndexedSeq[A])(less: (A,A) => Boolean): MinMax[A]
```

getMinMax returns the minimum and the maximum values of an object **IndexedSeq**[**A**] wrapped in a **MinMax**[**A**] object defined like this:

```
case class MinMax[A](min:Option[A], max:Option[A])
```

If the sequence is empty, then the result is **MinMax**[**A**](**None**,**None**).
If the sequence has one element *x*, then the result is **MinMax**[**A**](**Some**(x),**Some**(x)).
Example: getMinMax for Vector(1, -3, 0, 5, 2, 4) using less = _ < _ returns
MinMax[**A**](**Some**(-3),**Some**(5)).

To get any credit your solution must do the following:

- define a **monoid** for the **MinMax**[**A**] type, using the less**:** (**A**,**A**) **=> Boolean**) function given as argument to getMinMax.
- Use the foldMapV function described in the tetbook/notes to compute the **MinMax**[**A**] object with the minimum and maximum values of the *v* argument.

Write a *main* method that demonstrates how the getMinMax function is used.

Hint: get inspiration from function `ordered(ints: IndexedSeq[Int]): Boolean` from the textbook/notes.

b) Prove formally that the **MinMax[A]** monoid follows the two monoid laws.

Problem 3. Extra Credit 10 points

A company selling used phones on eBay keeps its inventory information in a CSV file that looks like this:

Product ID	Name	Unit cost $	Sell price $	Stock	Sold
2324	Apple iPhone 6s	189	210	5	2
9842	Motorola G6	146	155	10	7
4471	Samsung Galaxy S7	190	199	3	5

The Canvas homework page includes a link to this file.
Write a program with **the IO monad code from the textbook** (following the principles from Chapter 13) that reads from the terminal the name of a file with this format and then computes and displays the total cost, revenue, and profit from items sold across the inventory.
Feel free to interpret what these terms mean and to select the output format.
Make sure you comment your code.

Grading:

Problems 1 and 2 are worth 50 points each.

The points for programming problems are split into:

- code correctness: 85%
- programming style: 15%.

IMPORTANT NOTES:

- A submission that does not follow the instructions 100% (i.e. perfectly) will not get full credit.
- Submit the **h5.pdf** PDF file and the required .scala source files.
- Only submissions uploaded before the deadline will be graded.
- You have unlimited attempts to upload this assignment, but only the last one will be graded.