Scala Exercises

Atix Labs | http://www.atixlabs.com

Abstract

In this document you will find a series of exercises in order to asses your skills. *Functional Programming, Scala, Cryptocurrencies, Smart Contracts*

Before Starting

The following items will be taken in consideration:

- Delivering something it's a good start.
- We don't expect from you to know all the answers beforehand. Researching is a great skill.
- If you cannot solve an item, don't worry, we can always learn new things later. It's a plus if you share with us any partial result or idea in order to understand you thinking process.
- It's important not to miss the deadline.
- You can ask questions.

Functional Programming

Please, answer the following exercises using Scala [1]

1. "In cryptography and computer science, a hash tree or Merkle tree is a tree in which every non-leaf node is labelled with the hash of the labels or values (in case of leaves) of its child nodes. Hash trees allow efficient and secure verification of the contents of large data structures." [2]

   Given the following code, implement a Merkle root hash calculation function:

```
import scala.annotation.tailrec
import scala.collection.immutable.Seq
import java.security.MessageDigest

object FirstExercise {

  case class Node(someData: Seq[Byte])

  def hash(data: Seq[Byte]): Seq[Byte] =
   MessageDigest
    .getInstance("MD5")
    .digest(data.toArray[Byte]).to[Seq]

  def merkleRootHash(node: Node*): Seq[Byte] = ???

  def main(args: Array[String]): Unit = {
    println(merkleRootHash(
      Node(Seq(1, 1)),
      Node(Seq(0, 8)),
      Node(Seq(7, 10)),
      Node(Seq(2, 3)),
      Node(Seq(1)))
    )

    // Add more cases if needed
  }
}
```

[1]Haskell answers will be accepted too
[2]https://en.wikipedia.org/wiki/Merkle_tree

2. Implement, using foldRight, the following Seq[Int] operations:
   - `def sum(): Int`: calculates the sum of all the elements
   - `def elem(i: Int): Option[Int]`: returns the element for a given i position
   - `def filter(items: Seq[Int]): Seq[Int]`: returns a sequence without the given elements
   - `def map[A](fn: Int => A): Seq[A]`

3. Score a bowling game

Bowling is game where players roll a heavy ball to knock down pins arranged in a triangle. Write code to keep track of the score of a game of bowling.

The game consists of 10 frames. A frame is composed of one or two ball throws with 10 pins standing at frame initialization. There are three cases for the tabulation of a frame.

An open frame is where a score of less than 10 is recorded for the frame. In this case the score for the frame is the number of pins knocked down.

A spare is where all ten pins are knocked down after the second throw. The total value of a spare is 10 plus the number of pins knocked down in their next throw.

A strike is where all ten pins are knocked down after the first throw. The total value of a strike is 10 plus the number of pins knocked down in their next two throws. If a strike is immediately followed by a second strike, then we can not total the value of first strike until they throw the ball one more time.

Here is a three frame example:

Frame 1	Frame 2	Frame 3
X (strike)	5/ (spare)	9 0 (open frame)

Frame 1 is $(10 + 5 + 5) = 20$

Frame 2 is $(5 + 5 + 9) = 19$

Frame 3 is $(9 + 0) = 9$

This means the current running total is 48.

The tenth frame in the game is a special case. If someone throws a strike or a spare then they get a fill ball. Fill balls exist to calculate the total of the 10th frame. Scoring a strike or spare on the fill ball does not give the player more fill balls. The total value of the 10th frame is the total number of pins knocked down.

For a tenth frame of X1/ (strike and a spare), the total value is 20.

For a tenth frame of XXX (three strikes), the total value is 30.

Requirements: Write code to keep track of the score of a game of bowling. It should support two operations:

```
// Is called each time the player rolls a ball. The argument
// is the number of pins knocked down.
def roll(pins : Int) = ???
// Is called only at the very end of the game. It returns
// the total score for that game.
def score() : Int = ???
```

Don't forget to add as many tests and examples as you consider

Cryptocurrencies

1. Double-spending is the result of successfully spending some money more than once. This is an important problem any cryptocurrency should solve. How does Bitcoin prevent this kind of attacks?

2. What's a blockchain fork? What kinds of fork exist?

Smart Contracts (optional, Don't worry if you don't know the answer)

1. Let's imagine you need to implement a token based system using an Ethereum compatible smart contract (for example[3]). As a business requirement you want to prevent charging users when they use your system. How would you implement that? *Note: This question doesn't require coding.*

[3]https://github.com/OpenZeppelin/zeppelin-solidity/blob/master/contracts/token/BasicToken.sol