Homework 9: Regular Expressions

In this assignment, you'll write several handy regular expressions. The assignment concludes with some more challenging problems that illustrate the kind of computations that are possible with regular expressions.

1 Preliminaries

You should create a directory-tree that looks like this:

```
./regexes
├── build.sbt
├── project
│   └── plugins.sbt
└── src
    ├── main
    │   └── scala. ..... Your solution goes here
    └── test
        └── scala ..... Yours tests go here
```

Your `build.sbt` file must have exactly these lines:

```
resolvers += "PLASMA" at "https://dl.bintray.com/plasma-umass/maven"
libraryDependencies += "edu.umass.cs" %% "compsci220" % "1.2.1"
```

The `project/plugins.sbt` file must have exactly this line:

```
addSbtPlugin("edu.umass.cs" % "cmpsci220" % "3.0.1")
```

2 Programming Tasks

You may assume that your regular expressions will only be used to match complete strings. Therefore, you don't need to use the `^` and `$` metacharacters. A simple way to test if a regular expression `regex` exactly matches a string `str`, is to write `regex.pattern.matcher(str).matches()`.

You should use the template in fig. 35.1 for your solution and fill in the regular expressions as follows:

1. Define the regular expression `notAlphanumeric`, which only matches strings that don't contain any letter or digit.

```
import scala.util.matching.Regex

object Regexes extends hw.regex.RegexLike {
  def notAlphanumeric: Regex = ???
  def time: Regex = ???
  def phone: Regex = ???
  def zip: Regex = ???
  def comment: Regex = ???
  def numberPhrase: Regex = ???
  def roman: Regex = ???
  def date: Regex = ???
  def evenParity: Regex = ???
}
```

Figure 35.1: Template for the regular expressions.

```
class TrivialTestSuite extends org.scalatest.FunSuite {

  test("The Regexes object must be defined") {
    val regexes: hw.regex.RegexLike = Regexes
  }

}
```

Figure 35.2: Your solution must pass this test suite with no modifications.

2. Define the regular expression `time`, which only matches times written as five-character strings `HH:MM`, where the hours range from 00–23 and the minutes from 00–59.

3. Define the regular expression `phone`, which only matches phone numbers in the format `(XXX) XXX-XXXX`, where the letter `X` is a placeholder for a digit.

4. Define the regular expression `zip`, which matches either five-digit or nine-digit ZIP codes. i.e., strings in the format `XXXXX` or `XXXXX-XXXX`.

5. Define the regular expression `comment`, which only matches strings that start with `/*` and end with `*/`.

6. Define the regular expression `numberPhrase`, which only matches the strings `twenty`, `twenty-one`, `twenty-three`, ⋯, `ninety-nine` (don't forget the hyphen).

7. Define the regular expression `roman`, which only matches strings that represent the numbers 0—39 in roman numerals. i.e., the string may only contain the characters `I`, `V`, and `X`, with the usual constraints on roman numerals.

8. Define the regular expression `date`, which only matches dates written as ten-character strings `YYYY-MM-DD`. For example, `2016-04-01` represents April 1 2016, whereas `2016-04-31` is not a valid date. You should account for leap years by assume that Feb 29th is valid if the year is evenly divisible by four. E.g., `2016-02-29` is a valid date but `2017-02-29` is invalid.

9. The *parity* of a number is the sum of the digits of the number. For example, the parity of 203 is $2+0+3=5$ (odd) and the parity of 307 is $3+0+7=10$ (even). Define the regular expression `evenParity` that only matches numbers with even parity. You may find it convenient to accept numbers with leading zeroes.

3 Check Your Work

Figure 35.2 is a trivial test suite that simply ensures that you've defined the right object.

4 Hand In

From the `sbt` console, run the command `submit`. The command will create a file called `submission.tar.gz` in your assignment directory. Submit this file using Moodle.

For example, if the command runs successfully, you will see output similar to this:

```
Created submission.tar.gz. Upload this file to Moodle.
[success] Total time: 0 s, completed Jan 17, 2016 12:55:55 PM
```

Note: The command will not allow you to submit code that does not compile. If your code doesn't compile, you will receive no credit for the assignment.