Assignment #6
CS 3060 Programming Languages, Fall 2020
Instructor: S. Roy

Scala #2

Due Date: November 8 @ 11.59 PM.
Total Points: 60 points

Directions: Using the source provided via Gitlab `https://gitlab.com/sanroy/fa20-cs3060-hw/`, complete the assignment below. The process for completing this assignment should be as follows:

1. You already forked the Repository "sanroy/fa20-cs3060-hw" to a repository "yourId/fa20-cs3060-hw" under your username. If not, do it now.
2. Get a copy of hw6 folder in "sanroy/fa20-cs3060-hw" repository as a hw6 folder in your repository "yourId/fa20-cs3060-hw"
3. Complete the assignment, committing changes to git. Each task code should be in a separate file. As an example, task1.scala for Task 1.
4. Push all commits to your Gitlab repository
5. If you have done yet done so, sdd TA (username: prabeshpaudel) as a member of your Gitlab repository

Tasks:

1. **(15 points) Task #1:** In this task, your code creates a random list of 50 shape objects, then traverses the list from start to end, and computes the total area of the shape objects. First you need to implement the class hierarchy diagram of the shape types, which is aavailable in the same directory. Shape is an abstract class which has only a "color" attribute whereas Circle class and Rectangle class are concrete children of Shape class, and they have more attributes and constructors. Note that you do not know beforehand the order of the shape objects (i.e. Circles and Rectangles) created in the random list, e.g. you do not know beforehand whether the 1st item is Circle or Rectangle. *Writing README carries 1 point.*

   Note. When you traverse the list to calculate the total area, you need to call the area() function of each shape object.

   Hint: While building the list of shape objects, generate a random number 0 or 1; if 0, then you may add a Circle object, else add a Rectangle object to the list.

2. **Task #2a: (8 points)** Write a Scala program which takes a webpage url (say x) from the user, and then download webpage x. Count the number of images (i.e. "") and scripts (i.e. "<script ...> </script>") present in x.

   Task #2b: (8 points) Say webpage x contains links to other webpages (can be zero, one, or more) y. Your program needs to download all such pages (i.e. x and y). Count the total num of images on x and y. See Figure 1.

   Task #2c: (7 points) Also, your program needs to count how many pages (found in Task 2b) have more than 2 images.

   Task #2d: (7 points) Do Task 2b again, but now you are using the `par` (i.e. parallel collection). How much time does the concurrency usage save compared to the serial run in Task 2b?

   Writing README for Task 2 carries 2 points.

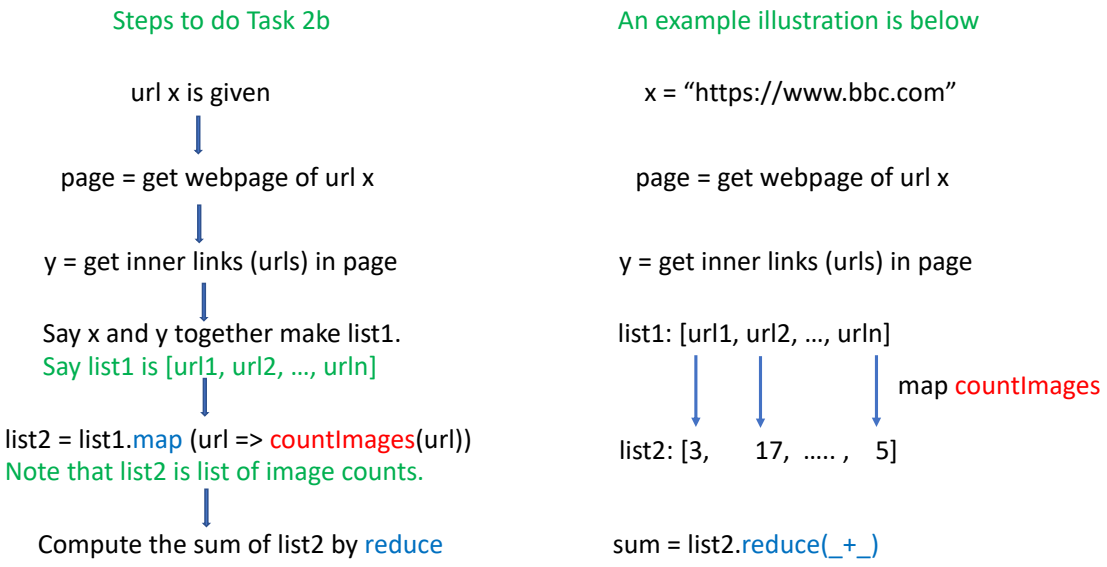

Figure 1: The steps to do for Task 2b.

3. **Task #3a: (10 points)** Write a function foo which takes a list t of strings as input. For each string s in t, foo uses the higher-order function `map` to compute the list of unique characters present in string s. Finally, foo returns a new list containing these lists. As an example, if t is ["abcdb", "bcdd"], then foo should return [['a', 'b', 'c', 'd'], ['b', 'c', 'd']].

   Task #3b: (5 points) Write a function foo which takes a list t of integers as input. Then, foo uses `foldLeft` to compute the sum of cubes of all elements of t. Finally, foo returns the computed sum.

   Writing README for Task 3 carries 1 point.