Assignment #4
CS 3060 Programming Languages, Fall 2023
Instructor: S. Roy

Scala #2

Due Date: Hw4a is due on Nov 3, and Hw4b is due on Nov 6 @ 11.59 PM.
Total Points: 60 points

Directions: Using the source provided via Gitlab `https://gitlab.com/sanroy/fa23-cs3060-hw/`, complete the assignment below. The process for completing this assignment should be as follows:

1. You already forked the Repository “sanroy/fa23-cs3060-hw” to a repository “yourId/fa23-cs3060-hw” under your username. If not, do it now.
2. Get a copy of hw4 folder in “sanroy/fa23-cs3060-hw” repository as a hw4 folder in your repository “yourId/fa23-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. Take a screenshot of your git commands (git add, git commit, git status, git pull, git push) on a terminal (which also shows your name and time) and submit the screenshot (a png or pdf file or the like) to gitlab. Check on gitlab.com whether the update time of each file matches with your push time.
6. If you have done yet done so, TA Manoj (username: matlim) and Roy (username: sanroy) as members of your Gitlab repository

Tasks:

1a. **(4 points) Task #1a: Part of Hw4a** Write a function foo which takes a string s and returns the number of unique characters present in string s. One constraint: You are not allowed to use any `var` (i.e., mutable) variable. You need to apply `toSet` function on string s to get the `set` of all characters in s.

1b. **(3 points) Task #1b: Part of Hw4a** Write a function bar which takes a list b of strings as input. For each string s in b, bar computes the number of unique characters present in string s using foo (of Task 1a). Finally, bar returns a new list containing these numbers. As an example, if b is ["xycaabcdb", "cczze", "yxzwx"], then bar should return List(6, 3, 4). One constraint: You are not allowed to use any `var` (i.e., mutable) variable. Also, you have to use the higher-order function `map` and foo that you developed in Task 1a.

Writing README for Task 1 carries 1 point.

2. **(7 points) Task #2: Part of Hw4a** Write a function foo which takes a list b_1 of integers as input. Function foo needs to construct (and return) a list b_2 where b_2 has the same items as b_1, but pair-wise swapped. As an example, if b_1 is List(3,4,15,16,17,18) then b_2 is List(4,3,16,15,18,17); if the list has odd number of items, then the last element should stay in its place e.g., if input b_1 is List(3,4,15,16,17) then output b_2 is List(4,3,16,15,17). One constraint: You are not allowed to use any `var` (i.e., mutable) variable. Hint: design the function foo as a recursive function with multiple cases such as list (e.g., head::tail) patterns. *Writing README for Task 2 carries 1 point.*

3. **(10 points) Task #3: Part of Hw4a** In this task, your code creates a random list of 25 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 shown as Figure 1. Shape is an abstract class which has only a "color" attribute whereas Circle class, and Hexagon class are concrete children of Shape class. Note that you do not know beforehand the order of the shape objects (i.e. Circles, and Hexagons)

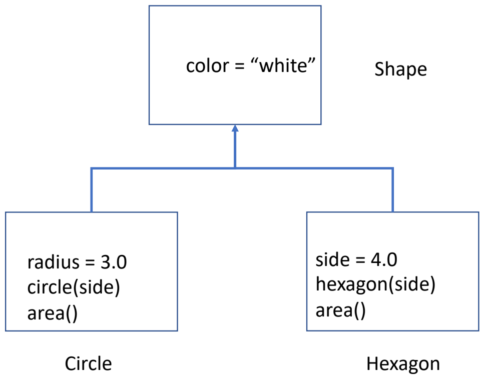

Figure 1: The class hierarchy diagram of the shape types for Task 3

created in the random list, e.g. you do not know beforehand whether the 1st item is Circle or Hexagon. *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 Hexagon object to the list.

4a. **(6 points) Task #4a: Part of Hw4a** Write a function foo which takes a string s and checks if s is a palindrome or not; if s is a palindrome then foo returns true; otherwise, it returns false. One constraint: You are not allowed to use any `var` (i.e., mutable) variable. Hint: to implement function foo, use recursion.

4b. **(4 points) Task #4b: Part of Hw4a** Write a function bar which takes a list b of strings as input. , and bar returns a new list with those strings in b, which are palindromes. As an example, if b is ["malayalam", "cczze", "yxzxy"], then bar should return List("malayalam", "yxzxy"). One constraint: You are not allowed to use any `var` (i.e., mutable) variable. Hint: Use the higher-order function `filter` and foo that you developed in task 4a.

Writing README for Task 4 carries 1 point.

5. **(8 points) Task #5a: Part of Hw4b** Write a Scala program which takes a webpage url (say x) from the user (or as a parameter), and then download webpage x. Count the number of images (i.e. "<img .../>") and scripts (i.e. "<script ...> </script>") present in x. As an example, url x can be "https://www.cnn.com".

   (8 points) Task #5b: Part of Hw4b Say the webpage of url x contains *links* to other webpages which we denote by y, i.e., y is a set of (zero, one, or more) webpages. Your program needs to download all such webpages (i.e. x as well as y). Count the total number of images on x and y. See Figure 2.

   (5 points) Task #5c: Part of Hw4b Also, your program needs to count how many webpages (found in Task 5b) have more than 50 characters.

   (5 points) Task #5d: Part of Hw4b Do Task 5b 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 5b?

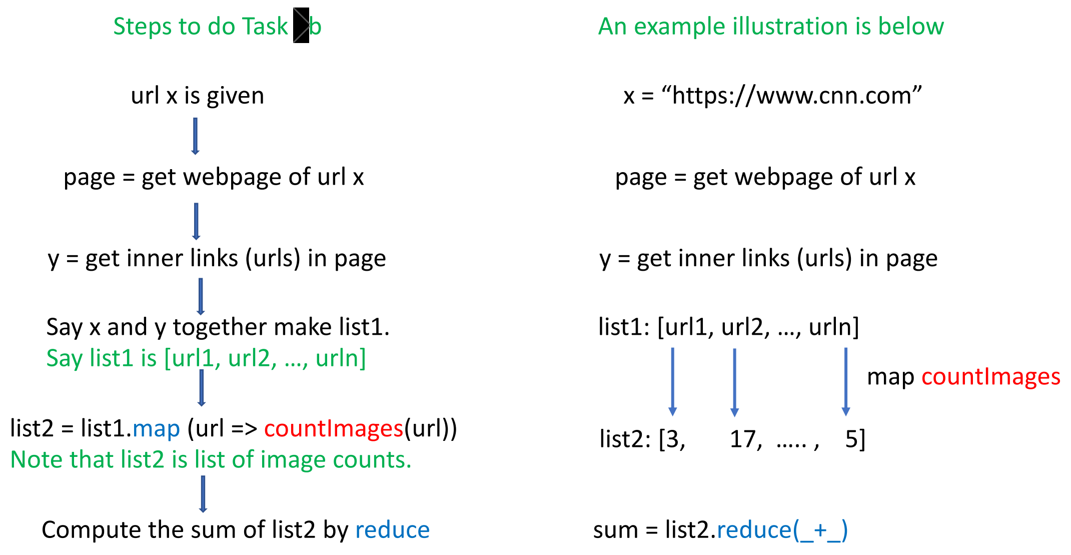
Steps to do Task [redacted]b
url x is given
page = get webpage of url x
y = get inner links (urls) in page
Say x and y together make list1.
Say list1 is [url1, url2, …, urln]
list2 = list1.map (url => countImages(url))
Note that list2 is list of image counts.
Compute the sum of list2 by reduce
An example illustration is below
x = “https://www.cnn.com”
page = get webpage of url x
y = get inner links (urls) in page
list1: [url1, url2, …, urln]
map countImages
list2: [3, 17, ….. , 5]
sum = list2.reduce(_+_)

Figure 2: The steps to do for Task 5b.

Writing README for Task 5 carries 2 points.