CSC 416 – Homework 4 (100 points)

Due Date: Monday, October 31st

In this homework, you will develop an app to show a list of cities that have bike shares like Cincinnati's Red Bike (smaller version of this is the NKU bike share). The user will be able to sort this information by city name or cities' distance to a hard-coded location. In this assignment you will learn how to:

- Use JSON data by using an open source library (SwiftyJSON)
- Compute distance using the Haversine formula

Steps to follow for setup :

1. Clone the project https://classroom.github.com/assignment-invitations/9735d1e322e7392c7226fbd48a4b9f45
2. When opening this project make sure to open .xcworkspace!

User Interface should have:

- A TextField
    - Allows to input how many cities will be displayed
    - It should allow only numbers to be inputted
- A TextView
    - It will show the data (= the list of cities) as text
    - It should be uneditable but scrollable
- Three buttons –

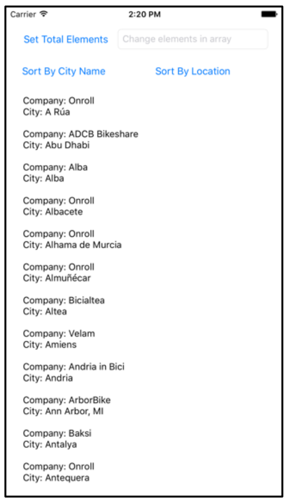
  First Button ("Set Total Elements")
    - Should change the number of cities displayed in the TextView, according to the number provided in the TextField

  Second Button ("Sort by City Name")
    - Should sort cities in the TextView alphabetically by city name

  Third Button ("Sort by Location")
    - Should sort cities in the TextView by distance to the hard-coded location, from the closest to the furthest

You have an example in images of how the app looks like in various stages below:

Launch Screen
(Sorted by city names)



Company: Onroll
City: A Rúa

Company: ADCB Bikeshare
City: Abu Dhabi

Company: Alba
City: Alba

Company: Onroll
City: Albacete

Company: Onroll
City: Alhama de Murcia

Company: Onroll
City: Almuñécar

Company: Bicialtea
City: Altea

Company: Velam
City: Amiens

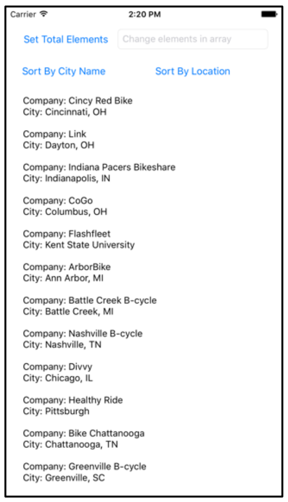
Company: Andria in Bici
City: Andria

Company: ArborBike
City: Ann Arbor, MI

Company: Baksi
City: Antalya

Company: Onroll
City: Antequera

Sort By Location



Company: Cincy Red Bike
City: Cincinnati, OH

Company: Link
City: Dayton, OH

Company: Indiana Pacers Bikeshare
City: Indianapolis, IN

Company: CoGo
City: Columbus, OH

Company: Flashfleet
City: Kent State University

Company: ArborBike
City: Ann Arbor, MI

Company: Battle Creek B-cycle
City: Battle Creek, MI

Company: Nashville B-cycle
City: Nashville, TN

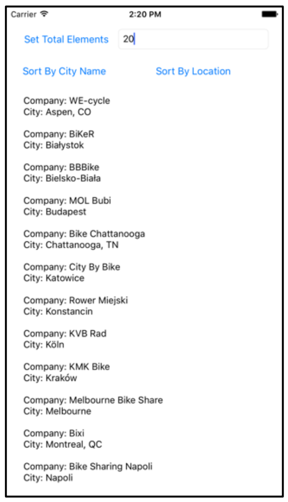
Company: Divvy
City: Chicago, IL

Company: Healthy Ride
City: Pittsburgh

Company: Bike Chattanooga
City: Chattanooga, TN

Company: Greenville B-cycle
City: Greenville, SC

Change Number of Cities
Displayed



Company: WE-cycle
City: Aspen, CO

Company: BiKeR
City: Białystok

Company: BBBike
City: Bielsko-Biała

Company: MOL Bubi
City: Budapest

Company: Bike Chattanooga
City: Chattanooga, TN

Company: City By Bike
City: Katowice

Company: Rower Miejski
City: Konstancin

Company: KVB Rad
City: Köln

Company: KMK Bike
City: Kraków

Company: Melbourne Bike Share
City: Melbourne

Company: Bixi
City: Montreal, QC

Company: Bike Sharing Napoli
City: Napoli

Implementation Details:

- The TextView will be populated with the company name and city name that is gathered from JSON data.
- The list of cities will be sorted by city name (make sure to sort in a closure) and shown at launch.
- The user will be able to change the number of cities displayed in the TextView by specifying the desired number (it could be smaller or bigger) in the TextField and then pressing the "Set Total Elements" button. The cities will be displayed in alphabetical order in this case.
- The user can sort by city names in alphabetical order – using the "Sort by City Name" button. (make sure to sort in a closure)
- The user can also sort by closest city to them -- you can hard code the user location. A hard coded user location is provided in the starter app on Blackboard, but you can change it if you want. When finished sort by distance (make sure to sort in a closure), populate the TextView again.
- JSON data input file is from: https://api.citybik.es/v2/networks/
- You are given in the starter project a library called SwiftyJSON, which allows you to retrieve the JSON data and parse the returning results. The results are returned back as a Dictionary and so each data has keys you will need to use to retrieve values, when parsing it. Save each city into a city struct that has city name, latitude, longitude, company name, and an optional distance to.
  SwiftyJSON Tutorial
  https://www.hackingwithswift.com/read/7/3/parsing-json-data-and-swiftyjson
- To compute distances between two locations you will use the Haversine Formula. A description of how this distance works can be found at https://rosettacode.org/wiki/Haversine_formula. You can also find the function definitions for this distance, implemented in various languages,

Swift included. The Swift function is included below in this document for your convenience – given two locations expressed in latitude & longitude coordinates, the haversine function computes and returns the distance between them on a curved surface. The haversine function is also included in a Utilities class in the starter GitHub project.

```
func haversine(lat1:Double, lon1:Double, lat2:Double, lon2:Double) -> Double {
    let lat1rad = lat1 * M_PI/180
    let lon1rad = lon1 * M_PI/180
    let lat2rad = lat2 * M_PI/180
    let lon2rad = lon2 * M_PI/180

    let dLat = lat2rad - lat1rad
    let dLon = lon2rad - lon1rad
    var a = sin(dLat/2) * sin(dLat/2)
    a = a + sin(dLon/2) * sin(dLon/2) * cos(lat1rad) * cos(lat2rad)
    let c = 2 * asin(sqrt(a))
    let R = 6372.8
    return R * c
  }
```

What to submit: Push the solution to this problem to GitHub. Once that is submitted then please submit your GitHub user name via Assignment 4 in Blackboard.