# CMPS 12A – Fall 2017
# Programming Assignment 3: Data Analysis
# Due: Friday November 3, 2017

## Overview
This assignment is designed to create a program that reads in a set of data, do some simple analyses on the data, and print these analyses to the user. The dataset we will be using is yearly rainfall (in inches) in San Francisco since 1849. The analyses that will be calculated are average rainfall, highest rainfall year with rainfall, and lowest rainfall year with rainfall. While the analyses are seemingly simple, reading in the data from a file, assigning the data to an array, and passing/returning an array to methods will be the primary challenges of this program. The dataset we are using comes from Climate of San Francisco (website: http://ggweather.com/sf/monthly.html)

## Details
The main method will first call the populateArray method to read in the data from the rainfall.txt file and store it in an array, type double. This data file can be found on the course webpage in the Examples/Program3 directory, you will need to include this file in the same folder in which you are writing your java program. After the data is read into an array, you should continually prompt the user for an option to choose for data analysis, and each option should call the appropriate method. The options you can use are “highest” (highest rain year), “lowest” (lowest rain year), “average” (average rainfall since 1849), and “quit”. Entering quit should stop the program. If the user doesn’t enter the right command, tell the user so and re-display options.

Since this program will be reading files, include `import.java.io.*;` at the top of your java file. Name your java file ReadFile.java.

You will need to implement 4 methods:

1. **populateArray()** which will accept no parameters, but will return an array of type double. The array will contain all values listed in rainfall.txt, data-type double. Each line will be its own value stored in the array, which is the amount of rain that fell each year (the first line in the file is amount of rainfall during year 1849 followed by each year till 2016). To make things easier for you (so you don’t have to loop through the file just to count how many years there are), there are 168 years in this dataset.

   To make this method work, you will need to first create a File object of the rainfall.txt file, read that file into a Scanner object, loop through the entire Scanner object, assigning each double to an array entry. This process (populating the array) will need take place inside a try-catch block; a demonstration of which can be found in the file ReadFile.java (in the

examples folder on the course webpage). This method will return the array of doubles that was just populated.

2. **findHighestYearIndex()**, which will accept an array of type double (which is the rain dataset) as a parameter, and return an integer. The returned integer is the index in the array, the value of which has the highest amount of rainfall for that year. Bare in mind, this will be an integer between the range of 0 and 167.

3. **findLowestYearIndex()**, which will accept an array of type double (which is the rain dataset) as a parameter, and return an integer. The returned integer is the index of the array, the value of which has the lowest amount of rainfall for that year. As above, this will be an integer between the range of 0 and 167.

4. **findAverage()**, which will accept an array of type double (which is the rain dataset) as a parameter, and return a double. The returning double is the average of all yearly rain totals in the dataset.

Sample Output (user input in bold):

Enter the option "average", "highest", "lowest":
**average**
Average rainfall since 1849: 21.86 inches

Enter the option "average", "highest", "lowest":
**lowest**
Lowest rainfall since 1849: 7.42 inches which occured during year 1850

Enter the option "average", "highest", "lowest":
**highest**
Highest rainfall since 1849: 49.27 inches which occured during year 1861

Enter the option "average", "highest", "lowest":
**mispelling**
Invalid input

Enter the option "average", "highest", "lowest":
**quit**
Program done.

## What to turn in

Submit a file called ReadFile.java on Canvas. No need to submit .class file or rainfall.txt file.

## Grading Rubric

| | |
|---|---|
| Program compiles and runs | 2pts |
| 2 pts for each method | 8pts |

1 point will be deducted for improper indentation and/or lack of comments.