Specification

Learning Objectives

By completing this assignment, students will be able to:

- Implement a well-designed Java class to meet a given specification
- Use arrays to
    - Explore datasets and make informed conclusions
    - Create and manipulate visualizations
- Follow prescribed conventions for code quality, documentation, and readability

💻 Program Behavior

You can see an example interaction with the program below (user input is **bold** and underlined):

This interaction produces a `disneyland_stats.txt` file that looks like this:

And also produces a `world_cup_stats.txt` file that looks like this:

The program begins by printing a short introduction and prompting the user for an input file containing a dataset (see below for the expected dataset format). It also prompts the user for an output file. The program will then process the dataset and generate a text-based histogram. It will also compute and print some summary statistics of the data: the number of values read, the mean (average), and the mode (most common value). The program prints the histogram and summary statistics to the output file specified by the user. It

then repeatedly prompts the user for an input and output file name until the user enters **anything other than "yes" (case-insensitive)** to "Do you want to create another histogram?".

You may assume the user will enter a valid input file name (i.e., an existing file) as well as a valid output file name (any file name ending with `.txt`).

🔢 Dataset Format

Each input file for this program will include a single dataset, with one number per line. The first value in the file will be the total number of values in the dataset. The value will then be followed by each individual data point, one per line. You may assume that all values in each dataset are integers, that all values are non-negative (i.e., greater than or equal to zero), and that the dataset is not empty (contains at least one value).

We have provided several sample datasets for you to work with.

- `seattle_temps.txt` - Average daily temperatures in Seattle for the year 2021, retrieved from the National Oceanic and Atmospheric Administration.
- `disneyland.txt` - Ratings on Trip Advisor for Disneyland California from April 2019, retrieved from Kaggle.
- `world_cup.txt` - Total goals per game from the 2018 FIFA World Cup, retrieved from https://gitlab.com/djh_or/2018-world-cup-stats/blob/master/world_cup_2018_stats.csv.

You may assume that the user will enter a valid file name. If you would like, you can also provide your own data! If you choose to analyze your own dataset, be sure it also meets the above requirements.

📊 Histogram Format

The histogram output by your program should begin at 0 and end at the largest value in your dataset. Each line of the histogram should represent one value and should consist of the value followed by a vertical bar ('`|`'), a space, and then one asterisk ('`*`') for each occurrence of that value in the dataset. All values between 0 and the maximum value in the dataset should be represented-- if a value does not occur in the dataset, your program should still print a line for that value but print zero asterisks on that line.

🚧 Program Structure

Your program should utilize methods, parameters, and returns to add a clear and understandable structure to the code. In particular, your code should include **at least the following 7 methods** with the indicated parameters and return values:

1. A method to print the program's introduction message
    - no parameters
    - no return value
2. A method to read in the data from an input file
    - takes one parameter, a `Scanner`
    - returns an array containing the data read from the parameter `Scanner`
3. A method to find the **index** of the maximum value in an array
    - takes one parameter, an array of integers
    - returns the **index** at which the maximum value of the array occurs
        - If the maximum value occurs multiple times, return the index of the first occurrence
4. A method to count the occurrences of each value in an array
    - takes one parameter, an array of integers

    - returns an array containing the number of times each value occurs in the parameter array
5. A method to compute the mean (average) of the data in an array
    - takes one parameter, an array of integers
    - returns the mean (average) of the data in the parameter array
6. A method to print a text-based histogram
    - takes two parameters: an array of integers, and a `PrintStream` object
    - no return value
7. A method to compute and print the summary statistics
    - takes three parameters: an array of integers containing data, an array of integers containing counts of that data, and a `PrintStream` object
    - no return value

With the exception of methods 3 and 5, all these methods should be called from `main`. Method 3 should be called **twice**, once in method 4 and once in method 7. Method 5 should be called from method 7.

HINT: The two calls to method 3 should pass *different* arrays as parameters: one call will pass the data from the file; the other call will pass the array of counts produced by method 4.

You may include additional methods if you wish, but you ***must*** include these methods as defined. (Naturally, your program will also include a `main` method.)

Along with the required methods above, you **must have a loop in** `main` which allows the program to keep creating histograms until the user chooses to stop.

Finally, your program must **only read the input and output file once.** You may not reset a `Scanner`, and you may not create more than one `Scanner`

connected to the input file. Likewise, you may not create more than one `PrintStream` connected to the output file.