CS 2334: Programming Structures and Abstraction

Project 4

Graphics with JavaFX

Due Date: Dec 10, 2020

Fall 2020

100/100 pts

1. Objective:

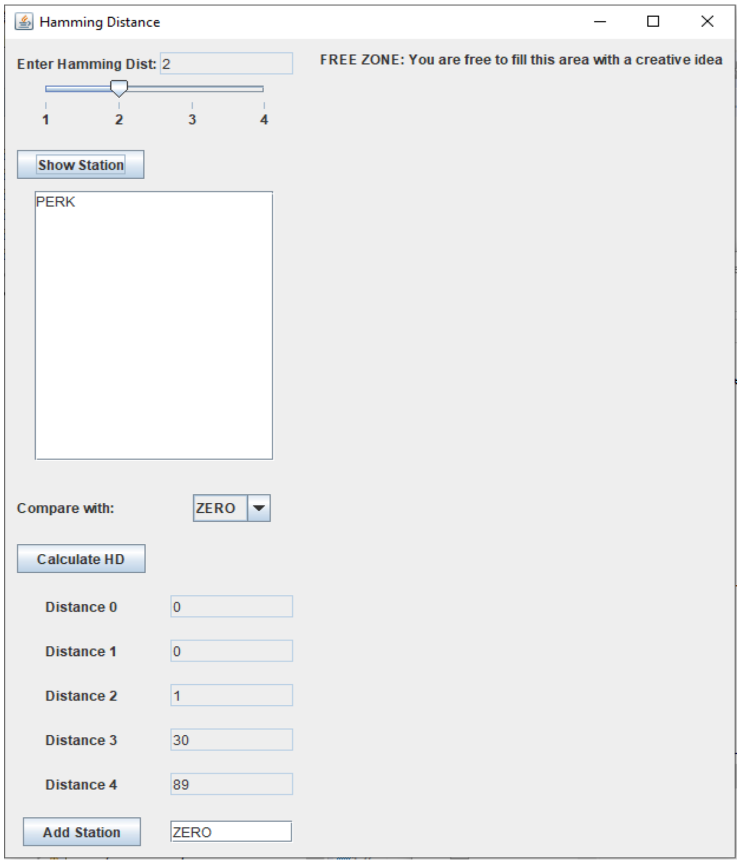
The objective of this programming project is to implement a java program where graphics content will be implemented using JavaFX. After completing the project, students will have an intermediate understanding of Graphics using JavaFX.

2. Project Specification:

2.1. Overall Program Behavior:

From Project 1 to Project 3, you are familiar with Hamming Distance, Mesonet, and reading a file where a station ID is a 4-letter code. For Project 4, we are giving a sample output of what you will generate. You are free to write class, method, etc. to generate the desired output. We will provide Mesonet.txt to read data. Zybook Section 17 will be helpful to implement this project. You can use the source code we provided, or you are free to start on your own, or you can start by modifying one of the templates from Zybook Section 17.

The screenshot of one of the final outputs (read the description along with running the JAR file):


Hamming Distance
FREE ZONE: You are free to fill this area with a creative idea
1
2
3
4
Show Station
Calculate HD
Add Station

Enter Hamming Dist: 2

PERK

Compare with: ZERO

Distance 0: 0

Distance 1: 0

Distance 2: 1

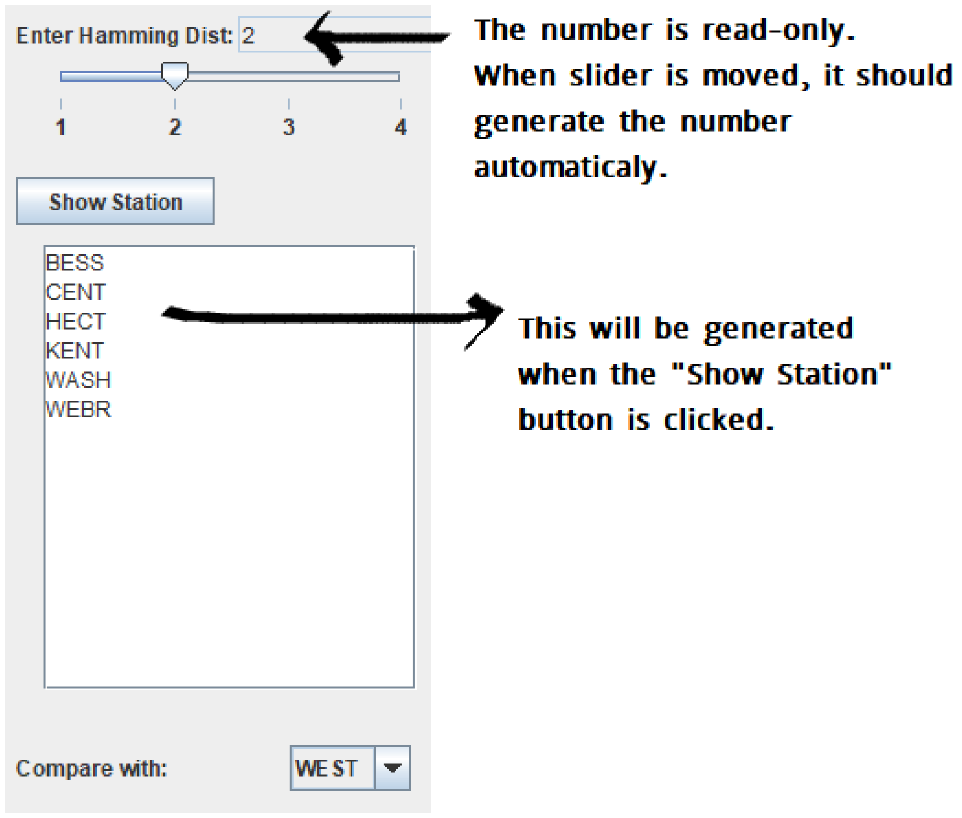
Distance 3: 30

Distance 4: 89

Add Station: ZERO

2.2 Description:

Part 1: up to 25 points (excluding the dropdown box = 10 points).



BESS
CENT
HECT
KENT
WASH
WEBR

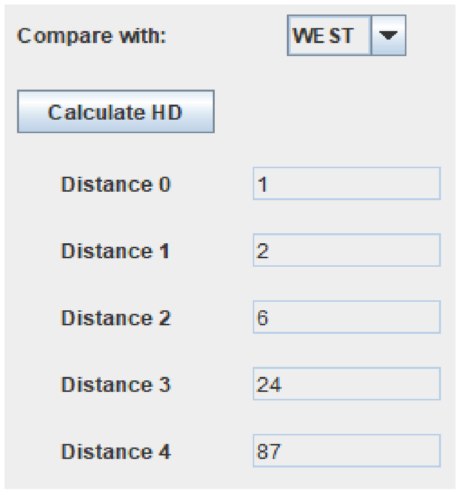
The number is read-only. When slider is moved, it should generate the number automaticaly.

This will be generated when the "Show Station" button is clicked.

You can see a Slider, a Button, a text area (listView), a dropdown box (comboBox), and some labels (Text).

In this example, if you press the button “Show Station” it will compare the Hamming Distance for WEST (it is selected in the dropdown) with all the stations where Hamming distance is 2 (this value is selected by the slider). The list of stations with Hamming distance 2 (respect to WEST) is being shown in the text area (listView).

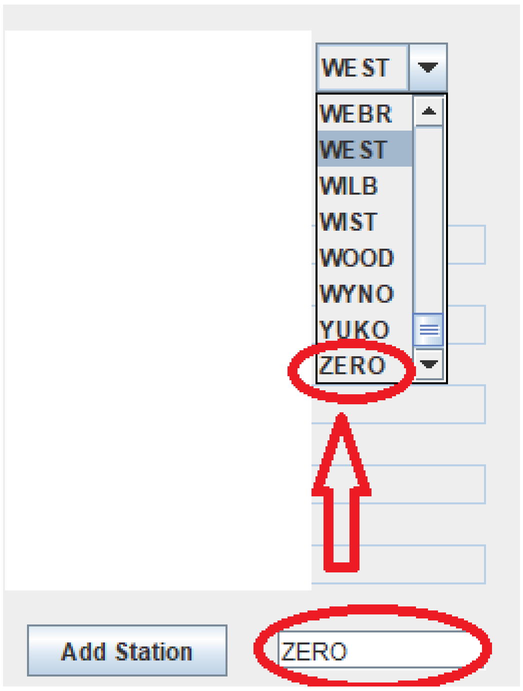
Part 2: up to 15 points (excluding the dropdown box = 10 points).



Distance 0	1
Distance 1	2
Distance 2	6
Distance 3	24
Distance 4	87

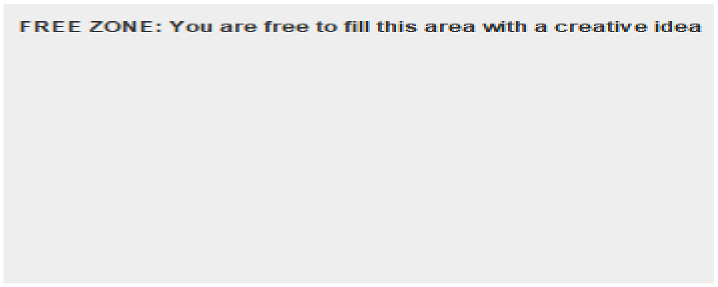
Here is the same dropdown box with another button, “Calculate HD”. The result is the distances between “WEST” and all the stations.

Part 3: up to 10 points.



“Add Station” will add a 4-letter code written in the text box to the list of the Dropdown box. In the example, “ZERO” is added to the List. Generally, in the dropdown box, you can add a station multiple times creating a duplication and the new station is added at the end of the list only. But you should prevent the duplication and will show the sorted list in the drop-down list including the newly added station to get a full credit.

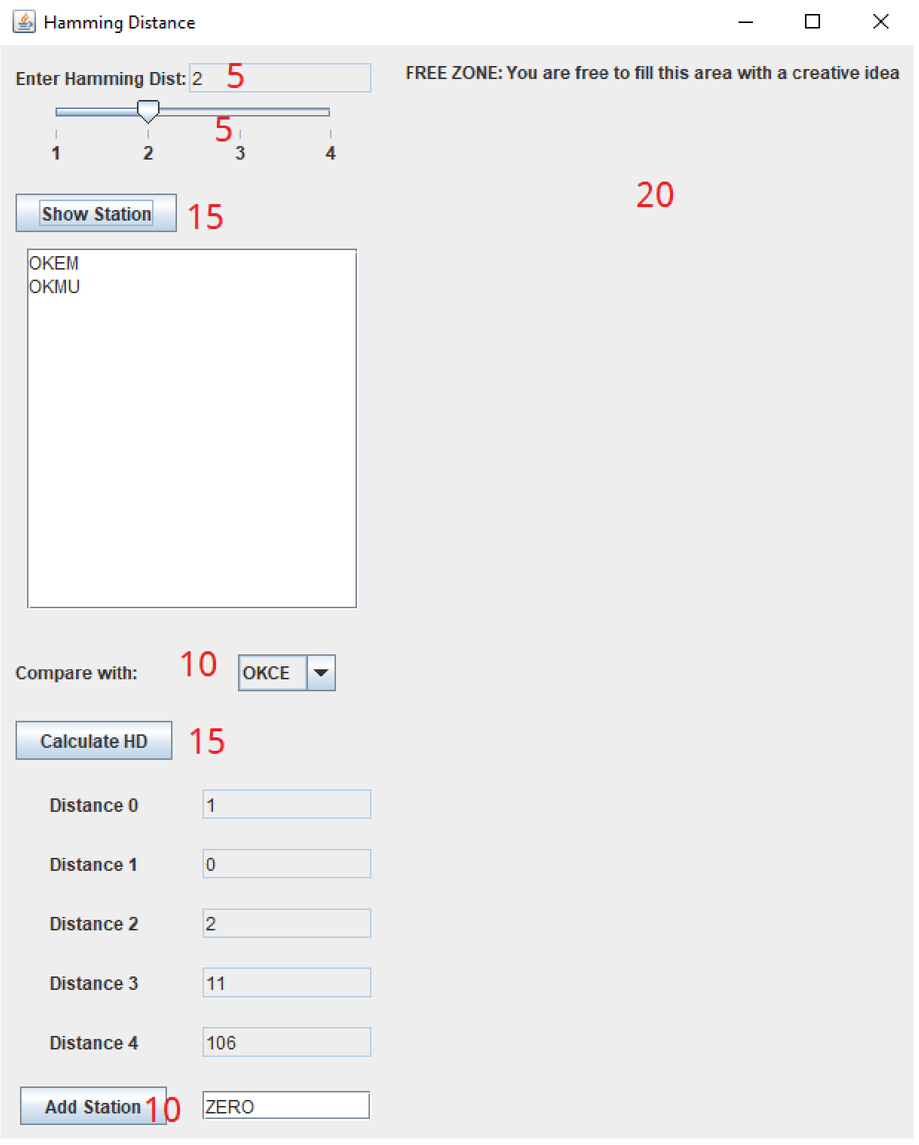
Part 4: up to 20 points.


FREE ZONE: You are free to fill this area with a creative idea

This area is open to you. You have to implement a creative idea using recursion and JavaFX. For example, you can implement Fibonacci sequence/GCD/Word scramble, etc. using slider/button/Text/listView/comboBox etc.

The point is, you are free to choose the topic, but the topic should be on recursion. Then implementation will be using JavaFX. Implementing one topic is sufficient for full credit.

3. Point Distribution:



Enter Hamming Dist: 2

OKEM
OKMU

Compare with: OKCE

Distance 0 1

Distance 1 0

Distance 2 2

Distance 3 11

Distance 4 106

ZERO

The points depend on the correctness of the output.

Documentation: 20 points (Javadoc on Github). As a part of the documentation, you will draw the UML of your program as well as describe your problem-solving approach with a detailed description of variables and methods. Standard coding style, variable names etc. should be maintained.

In total, (5 + 5 + 15 + 10 + 15 + 10 + 20 = 80) + 20 (Javadoc) = **100** points.

4. Submission Instructions: (Due on Dec 10, 2020, during lab session)

No submission on Zylab/Canvas is required. During the lab session, you will be graded based on your output, coding standard, and Documentation. Javadoc should be submitted to Github. Github commits along with a meaningful comment will be considered as part of the documentation (for grading).

Plagiarism will not be tolerated under any circumstances. Participating students will be penalized depending on the degree of plagiarism. It includes "No-code-sharing" among the students. It can lead to academic misconduct reporting to the Integrity Council if an identical code is found among the students.

5. Late Penalty:

There is no late submission for this project. There is no evaluation of this project after the lab session since this is the last week. If you have any difficulty to be present during the lab session, you can contact an instructor and demonstrate your project on or before Dec 10.

Note: You must use JavaFX to implement this project.

Note: In case you want to see the sample output for this project, I have uploaded a JAR file. You can directly run it; however, use the command prompt to run if you face a problem to run directly. For Linux, browse to your folder then run using the command: java -jar filename.jar

Note: The sample output is the latest; however, if you see any difference in the pdf and the JAR file, you can follow either one. The sample is given as a layout to follow, no need to waste time developing an exact duplicate of the sample in terms of design, i.e., if you place all the items with proper results that would be sufficient. Grading will be mainly on your accuracy of output.

Good Luck!!!