Evolution Network Visualizer

We are providing you with a CS15NetworkVisualizer class to help you with Part III of Evolution — optimization. The visualizer has two methods: show and plot. The reason for giving you this class is to show you how visualizing your data can help you make optimization decisions with your Neural Networks.

Importing

Use the following import statement to use the visualizer:

```java
import cs015.fnl.EvolutionSupport.CS15NetworkVisualizer;
```

Note that if you copied cs015.jar locally to your computer prior to November 16, you will need to copy it again to be able to access the visualizer.

Constructor

```java
public CS15NetworkVisualizer(int numInputNodes, int numHiddenLayerNodes)
```

The constructor for the NetworkVisualizer. Creates a new stage and scene and initializes the display with nodes and empty weights.

Show

```java
public void show(double[][] syn0, double[][] syn1)
```

This method takes in two 2D arrays of doubles representing weights of a bird's NeuralNetwork. The length of the arrays should be consistent with the number of input nodes and hidden layer nodes specified in the constructor; that is, syn0 should be of size [numHiddenLayerNodes][numInputNodes], and syn1 should be [1][numHiddenLayerNodes]. This method displays the connected layers of its NeuralNetwork:
The stroke of a line represents the magnitude of a weight, where thin lines represent weights closer to 0 and thick lines closer to ±1 (You should be passing in normalized weights). The color of a line represents the sign of the weight, where green lines represent positive weights and red lines represent negative weights.

Plot

```java
public void plot(ArrayList<Integer> generations,
                  ArrayList<Integer> fitnessHistory, String title)
```

This method takes in an ArrayList of ints containing generation numbers to be used for the x axis, an ArrayList of doubles containing fitness scores of each generation, and a String to title the plot. It plots a line of generations vs fitness. The length of generations must be equal to the length of fitnessHistory. For example: