```
// This program is copyright VUW.
// You are granted permission to use it to construct your answer to a COMP102 assignment.
// You may not distribute it in any other way without permission.

/* Code for COMP102 - 2024T3, Assignment 2
 * Name:
 * Username:
 * ID:
 */

import ecs100.*;
import java.awt.Color;
import java.util.*;

/**
 * The program contains several methods for analysing and displaying the profile of a road, based
on
 * GPS measurements every 10 meters along a section of the road.
 *  There are several things about the profile that a user may be interested in, especially
 *    The average and maximum height
 *    A visual plot of the road profile
 *    A description of whether the road segment is mostly uphill, mostly downhill, mostly flat, or
...
 */
public class RoadProfiler{

    public static final double LEFT = 30;       // where the start of the road segment will be
plotted
    public static final double SEA_LEVEL = 400; // heights will be plotted as a distance above y =
SEA_LEVEL
    public static final double STEP = 10;       // horizontal distance along road between height
measurements


    /**
     *  analyseProfile() reads a sequence of heights from the user and prints out
     *  average and maximum height and plots a profile of the road
     *  by calling appropriate methods
     */
    public void analyseProfile(){
        UI.clearPanes();
        ArrayList<Double> listOfHeights = UI.askNumbers("Enter profile levels, end with 'done':
");
        if (listOfHeights.size() != 0) {
            this.printAverageHeight(listOfHeights);
            UI.printf("Highest point was:  %.2f\n", this.maximumHeight(listOfHeights));
            this.displayProfile(listOfHeights);
        }
        else {
            UI.println("No readings");
        }
    }

    /**
     * Print the average height
     *   Assumes there is at least one number in the list,
     */
    public void printAverageHeight(ArrayList<Double> listOfHeights) {
        /*# YOUR CODE HERE */
        double total = 0;
        double Avg = 0;
        for(double num:listOfHeights){
            total = total + num ;

            Avg = total / listOfHeights.size();
        }
        UI.println("Average height was:  " + Avg+"m");
```

```
    }

    /**
     * Plot a profile of the road segment, assuming that 0m (sealevel) is at y=SEA_LEVEL
     *   - Core: Plot horizontal lines, 10 pixels long for each measurement, and the specified
height above SEA_LEVEL
     *   - Completion: Plot a continuous plot using lines that join up adjacent points
     *   - Challenge:
     *     - Scale the y-axis so that the largest numbers and the smallest just fit on the graph.
     *     - Show the vertical scale on the left side
     *   - Scale the x-axis to fit the number of measurements
     */
    public void displayProfile(ArrayList<Double> listOfHeights) {
        UI.clearGraphics();
        UI.drawLine(LEFT, SEA_LEVEL, LEFT+1000, SEA_LEVEL); // draw the base line to show sea
level
        /*# YOUR CODE HERE */
        double width = 0;
        double y = 0;
        for (double num:listOfHeights){
            UI.drawLine(LEFT+width,SEA_LEVEL - y , LEFT + 10 + width,SEA_LEVEL - num);
            width += 10;
            y=num;

        }

        UI.println("Finished plotting");
    }

    /**
     *  Find and return the maximum level in the list
     *   - There is guaranteed to be at least one level,
     *   - The method will need a variable to keep track of the maximum, which
     *     needs to be initialised to an appropriate value.
     */
    public double maximumHeight(ArrayList<Double> listOfHeights) {
        /*# YOUR CODE HERE */
        double max = 0 ;
        for (double num:listOfHeights){
            if(num > max){max = num;}



        }



        // if you haven't written this method yet, you can just include the line
        return max;
        // to make the class compile.
    }



    /**
     * Set up the Gui with buttons
     */
    public void setupGUI() {
        UI.initialise();
        UI.addButton("Analyse", this::analyseProfile );
        UI.addButton("Quit", UI::quit );
    }

    /**
     * main: construct object and call setup GUI.
     */
    public static void main(String[] args) {
        RoadProfiler rp = new RoadProfiler();
```

```
            rp.setupGUI();
        }

    }
```