Sarah Telford – a1810750

Testing Report

Overview
This document outlines the steps to start the code and provides a series of test cases to verify the functionality of the graphical user interface (GUI) implementation.

Starting the Code
To start the code, follow these steps:

1. Ensure that you have the necessary prerequisites installed (Java Development Kit, required JAR files, etc.).

2. Compile and run the code using the makefile
   A. Compile:
      ```
      make
      ```
   B. Run:
      ```
      make gui
      ```

Test Cases
Test Case 1: GUI Initialisation
Objective: Ensure the GUI initialises without errors.

Steps:
1. Start the GUI application.
2. Verify that the main window opens without any exceptions or errors.

Expected Result: The GUI should open successfully, and no error messages should be displayed.

Test Case 2: Event Range Check – Junit test
Objective: Verify that the GUI correctly checks whether a GPS event is within the specified range.

Steps:
1. Input a specified latitude and longitude range.
2. Press the apply button to apply input range
3. Check that the events displayed match the event range.

Expected Result: The GUI should only display tracks within the specified range.

Test Case 3: Distance Calculation – Junit test
Objective: Verify the accuracy of distance calculation between two GPS events.

Steps:
1. Input two GPS events with known coordinates.
2. Trigger the distance calculation functionality.

Expected Result: The GUI should accurately calculate the distance between the two events.

Test Case 4: Distance Label Update – Junit test
Objective: Ensure the GUI updates the distance label correctly based on a list of track events.

Steps:
1. Input a list of track events.
2. Trigger the distance label update functionality.

Expected Result: The GUI should update the distance label with the correct total distance.

Test Case 5: Display relevant information.

Objective. Ensure that track numbers, latitude and longitude are displayed in separate cells

Steps:
1. Start the GUI application.
2. Verify that the main window has cells for each stream, with each cell displaying information relevant to the specified event in a user-friendly manner.

Expected Result: The track name, latitude, and longitude for an event are displayed together in a user-friendly manner within the same cell.

Test Case 6: Incoming Events

Objective: ensure that there is a display field that shows each event as it is passed to the GUI, at the time it occurs.

Steps:
1. Start the GUI application.
2. Check for an incoming events cell

Expected Result: Incoming events display area should be visible and updated in real-time as events occur.

Test Case 7: Distance Visible

Objective: Ensure the visibility and clear association of the "distance travelled" attribute with its respective cell.

Steps:
1. Navigate to the relevant section or window of the GUI where distance travelled information is displayed.
2. Verify that the "distance travelled" attribute is prominently visible.
3. Confirm that it is clearly associated with the appropriate cell or entity.

Expected Result: The "distance travelled" attribute should be easily noticeable within the designated section of the GUI, and its association with the corresponding cell or entity should be unmistakable.

Test Case Log

Test	Status	Comments
1	Passed	No errors present on initialisation, start up and runs with no issues
2	Passed	Junit tests passed (see GPSGui_Test for test)
3	Passed	Junit tests passed (see GPSGui_Test for test)
4	Passed	Junit tests passed (see GPSGui_Test for test)
5	Passed	Relevant information visible in separate cells, only one event per cell is visible at a given time
6	Passed	Separate cell visible that rendering the incoming events for all cells
7	Passed	Distance variable is clearly visible, and it is clear what tracker each distance is related to