Horse Race Simulator – Report

Part I: Encapsulation and Horse Class Design

The Horse class uses private fields with public getter and setter methods. This provides encapsulation, which ensures that internal states such as confidence and position can only be accessed or modified through controlled methods. This prevents accidental changes and allows safe updates to a horse's state.

Part I: Unit Testing

I tested the Horse class using the HorseTest.java file. The test methods check:

- If a horse moves forward correctly
- If the confidence updates after movement
- If the fall() method sets the horse as fallen and stops further movement

Below is a sample output from the test: (Paste a screenshot from IntelliJ terminal or output window showing HorseTest results)

Part I: Improvements to the Race Class

I implemented random speeds for horses and improved the race logic so that horses fall at regular intervals (every 5 seconds), with a realistic probability (e.g. 10%). I also ensured that races can end properly if all horses fall.

Part II: GUI Design

The GUI allows users to customize the race by choosing:

- Number of horses
- Horse symbols
- Weather and track type

The GUI is implemented using Java Swing. The race is visualized using RaceTrackGUI, which shows horses moving across the track in real time, with falling horses marked in red.

Reflection

I learned how to implement object-oriented principles in Java and how to visualize real-time simulations using Swing. I also improved my Git skills by creating branches and pushing my project to GitHub.