Owen Lindsey

CST-239

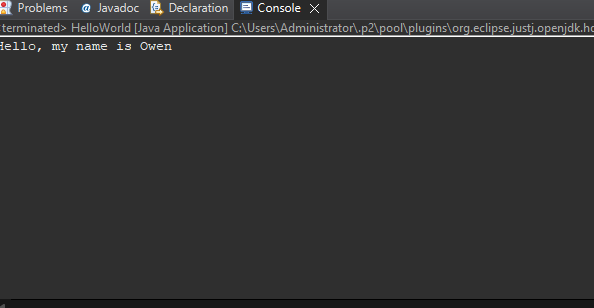
Professor Couch

Activity 1

09/24/2023

**Part 1.)** Deliverables

1. This is a screenshot of the eclipse IDE About B
2. This is a screenshot of the console output when running the helloWorld class.



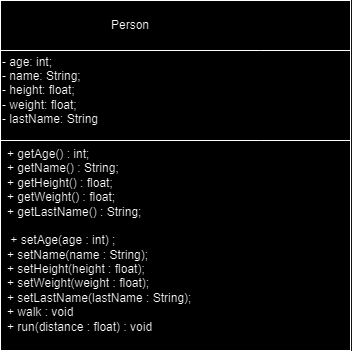
**Part 2.)** Person class / UML / theory of operations

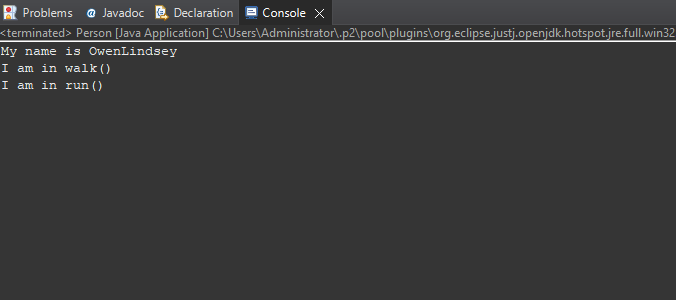
Theory of operations:

As a software development student, this Person class exemplifies fundamental concepts of object-oriented programming (OOP). It introduces defining and encapsulating attributes using private variables, while using public methods (getters and setters) to access and modify them, emphasizing data protection. The non-default constructor ensures immediate object state initialization upon creation, while the super() call hints at inheritance, even if there's no explicit superclass here. Behavior methods, like walk() and run(float distance), highlight that classes encapsulate behavior in addition to state. Lastly, the main() function showcases object instantiation, demonstrating how classes serve as blueprints for creating individual objects.

UML diagram explanation:

The UML diagram represents the Person class's structure. At the top, the class name Person is specified. In the middle section, private attributes such as age, name, height, weight, and lastName are defined with their respective data types. The bottom section displays the class's public methods: a constructor and several getters and setters corresponding to each attribute. Additionally, behavior methods, walk() and run(float distance), are also present. This UML provides a concise, visual summary of the Person class, detailing its attributes and operations, aiding in understanding the class's design and functionality.

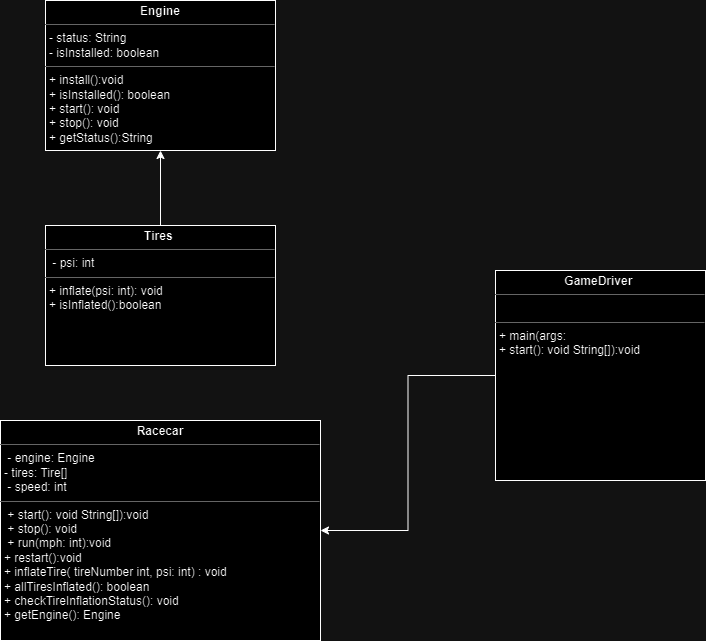


C.) Screenshot of the console output when running the Person class.

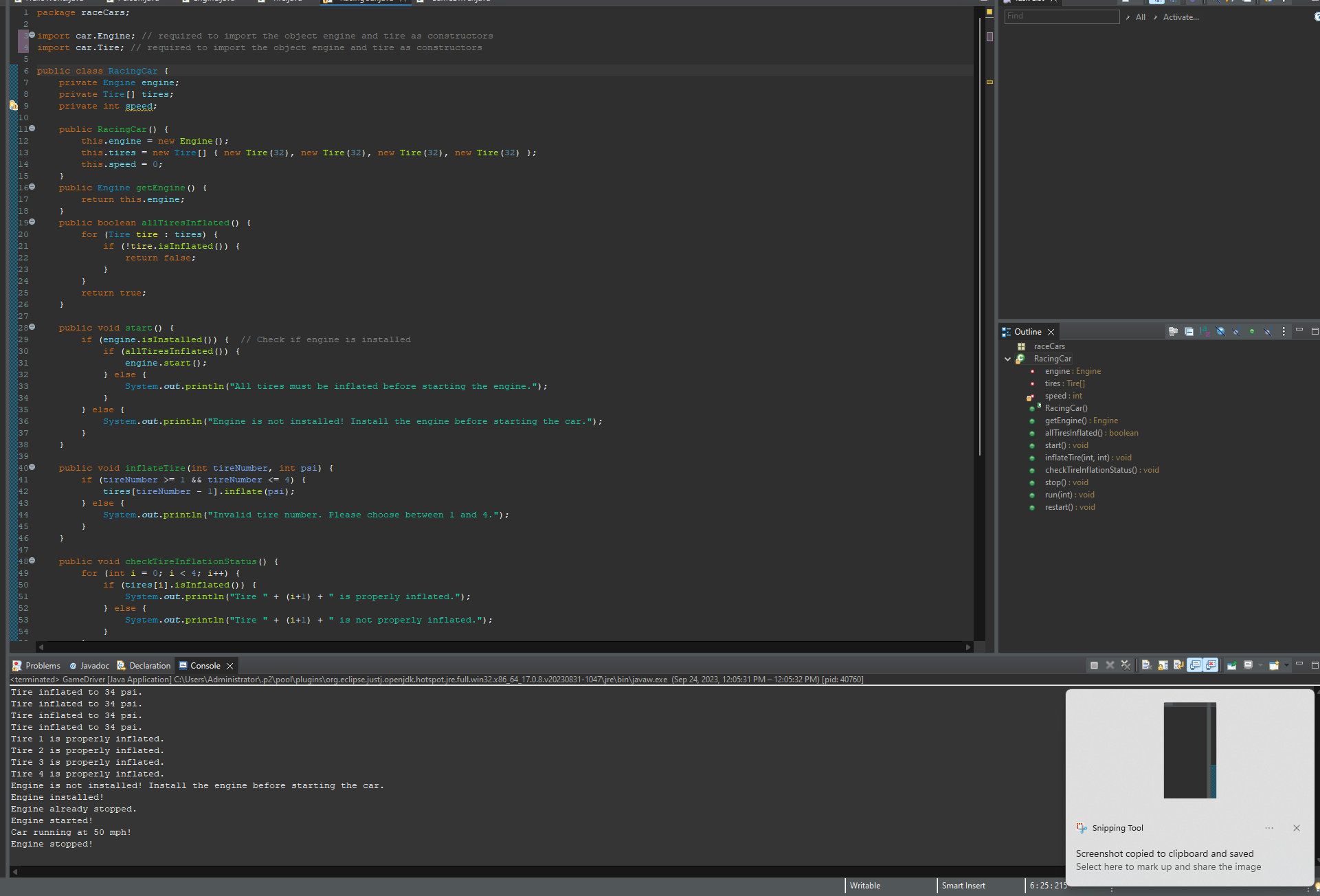
**Part 3.)** RaceCar game application

UML DIAGRAM OF COMPLETED MODEL:

In our race car shopping application, the RaceCar class defines the product attributes, such as name, description, price, and quantity. The InventoryManager class manages these products, allowing for addition and retrieval based on product names. The ShoppingCart class manages user interactions, letting users add or remove items and view their total cost. Each class has a specific function, illustrating a modular design approach. This separation makes the application easier to understand, debug, and expand upon, highlighting the benefits of a clear object-oriented design in software development.



Screenshots of application console:

 These screenshots show that the car must verify if the required rules are being followed. For example, the program will request the race car’s tire pressure which if at the appropriate PSI will then move onto the next check so on and so forth until the car can then finally start driving.