Lab 5 – EcoSort: The Smart Recycling Network

1. **Accept** the **Lab 5** assignment using GitHub Classroom link with your Github account.

2. Clone your GitHub Classroom repository using **IntelliJ IDEA**.

3. Complete **Lab5.java** as described in the comments and the requirements in this file.

4. Test your solutions in the **Lab5.java main function**.

5. Once your code runs correctly in the **main function**, use the provided **JUnit tests** to verify your work.

   - If the tests fail, use the **debugger** to fix your code.
   - **Do not modify the JUnit test files. Any changes will be detected.**

6. Use **Git** to submit your code to your GitHub Classroom repository.

   - Check the **GitHub Actions results** to confirm that your code passes all tests.

7. After completing all the steps above, be prepared to **demo** your code to me.

Overview

In this lab, you'll complete a series of small object-oriented programming tasks inside one file, Lab5.java.

The goal is to practice inheritance, constructor chaining, method overloading and overriding, generics, and the Collections Framework.

Instructions

You will work inside Lab5.java, which already contains starter code and TODO comments for each part.

Follow the directions in the comments to complete the class definitions and methods.

Implementation To-Do

RecyclableItem (Base Class)

- Represents a general recyclable item with a type, weight, and score.
- Implements constructor chaining and a getImpact() method (score * weight).

PlasticItem (Subclass)

- Extends RecyclableItem and adds a biodegradable field.
- Refines getImpact() to add +10 if biodegradable.

MetalItem (Subclass)

- Extends RecyclableItem and adds an alloyType field.
- Refines getImpact() to add +5.

RecyclingStation (Final Class)

- Demonstrates constructor chaining and method overloading.
- Includes calculateProcessingTime(...) methods with different parameter sets.

Bin (Generic-like Class)

- Uses an ArrayList to hold items.
- Includes addItem, removeItem, totalImpact, and a printItems() method using a ListIterator.

Comparators

- WeightComparator: sorts items by weight (ascending).
- ScoreComparator: sorts items by score (descending).

Generic Method

- printGenericList(...): prints all elements in a given list.

Testing

All testing will be done using CollectionsTest.java.
Run the JUnit tests — all should pass when your implementation is correct.