Assignment

Pancake Sorting Problem

A customer has a pancake business. The problem in his words:

> The chef in our place is sloppy, and when he prepares a stack of pancakes they come out all different sizes. Therefore, when I deliver them to a customer, on the way to the table I rearrange them (so that the smallest winds up on top, and so on, down to the largest at the bottom) by grabbing several from the top and flipping them over, repeating this (varying the number I flip) as many times as necessary. (https://doi.org/10.1016/0012-365X(79)90068-2)

The customer is fed up with rearranging them, he wants to automate the process of sorting pancakes (Pancake Sorting Machine) and therefore wants to model it in code. His user story:

> As a pancake business owner I want to be able to model the sorting of pancakes on a plate by flipping a (sub)stack of pancakes with a pancake spatula so that I can automate this process in a later stadium.

More context: The process is as follows, there is a chef who bakes pancakes and puts the pancakes on a plate. The plate always contains minimum 1 pancake and maximum 25 pancakes. All the pancakes have different diameters, so no pancake have the same size. Afterwards the plate goes to the assistant cook who sorts the pancakes by size: the smallest pancake is on top and the largest pancake is at the bottom. The assistant cook is only allowed to sort the pancakes by flipping the stack of pancakes using a spatula (any other method comes at the expense of taste...).

Your task is to create a Java program that models this.

- Create this program via TDD.
- Before each commit, review your code with the Code Review Checklist.
- Use Git with atomic commits that have comprehensive commit messages: https://dev.to/wordssaysalot/art-of-writing-a-good-commit-message-56o7 (body is not necessary).
- Analyze your code with Sonarqube.
- Refactor your code after each review.
- Let your code be reviewed at least once by a classmate via a merge request.
- More information on this problem: https://en.wikipedia.org/wiki/Pancake_sorting

Submission

- Code via git repository.
- A README.md (use the provided template!) in the root of your git repository.
- Submit an **exported pdf** of your README.md in DLO. You can export a README.md as follows: https://www.jetbrains.com/help/idea/markdown.html#export-markdown-file-to-another-format

Rubric

	Below Expectation	Meets Expectation	Above Expectation
Git log	More than 3 offences of git commit message conventions.	Max 3 offences of the git commit message conventions.	All commit messages adhere to git commit message conventions: Consistent casing. The imperative mood is used. No periods at at of message. Atomic commits. Clear messages. Correct use of structural elements: docs:, feat:, test:, refactor: , style: etc..
Sonarqube	Misses items of Meets expectations.	Reliability, Security,Maintainability and Duplications Ratings Screenshots are present.	Reliability, Security,Maintainability, Coverage and Duplications Ratings Screenshots are present and are discussed.
Test Driven Development	Misses items of Meets expectations.	Code snippets of the test classes are provided. A rationale is present for why the unittests are of good quality. The rationale is based on valid arguments.	Code snippets of the test classest are provided. A rationale is present for why the unittests are of good quality. The rationale is based on valid arguments. Short discussion how SOLID design principles have been applied or could have been applied.
Code Reviews	Misses items of Meets	Proofs to the code reviews are provided. Clear code reviews	Proofs to the code reviews are provided. Clear code reviews were given. Code reviews use the terms of the article

	expectations.	were given.	"What to look for in a code review" and clearly reference each section.

Above expectations items:

- *Solid design principles* are not discussed during the workshop so requires extra effort by the student to study these.
- How to show *code coverage* in Sonarqube is not discussed during the workshop so requires extra effort by the student to study these.
- The article *"What to look for in a code review"* (https://google.github.io/eng-practices/review/reviewer/looking-for.html) is briefly discussed in class but requires extra effort by the student to get acquainted with the article.

Resources

Code Reviews:

- Google: Engineering Practices: https://google.github.io/eng-practices/
- Modern Code Review: A Case Study at Google: https://research.google/pubs/pub47025/
- Google: What to look for in a code review: https://google.github.io/eng-practices/review/reviewer/looking-for.html
- Code Review Process in Gitlab: Create merge request https://www.youtube.com/watch?v=Ddd3dbl4-2w
- Code Review Process in Gitlab: Add comments on a merge request: https://www.youtube.com/watch?v=TviJH6oRboo

Maintainable software:

- SOLID principles to make Object Oriented designs more maintainable: https://learning-oreilly-com.rps.hva.nl/library/view/adaptive-code-agile/9781509302598/part03.html
- Building Maintainable Software, Java Edition: Ten Guidelines for Future-Proof CodeFebruary 2016 https://learning-oreilly-com.rps.hva.nl/library/view/building-maintainable-software/9781491955987/ch01.html

Git conventions:

- Art of writing a good commit message: https://dev.to/wordssaysalot/art-of-writing-a-good-commit-message-56o7
- How to write a Git commit message: https://cbea.ms/git-commit/
- Convention on top of commit messages: https://www.conventionalcommits.org/en/v1.0.0/#summary

Unit Testing:

- FIRST: https://learning.oreilly.com/library/view/pragmatic-unit-testing/9781680500769/f_0044.html
- Right BICEP: https://learning.oreilly.com/library/view/pragmatic-unit-testing/9781680500769/f_0051.html
- Getting Started with Mockito: https://www.baeldung.com/mockito-annotations
- To Mock or not to Mock: http://resolver.tudelft.nl/uuid:a0b02521-ad00-4f96-9d96-

17a85dc23f99

Static Code Analysis:

- Sonarqube installation: https://docs.sonarqube.org/latest/setup/get-started-2-minutes/
- Sonarlint IDE plugin: https://www.sonarlint.org/
- Code Coverage: https://www.baeldung.com/sonarqube-jacoco-code-coverage

Problems with sorting pancakes:

- The original paper on the pancake problem by Bill Gates (you might have heard of him) & Christos Papadimitriou (you might have heard of him): https://doi.org/10.1016/0012-365X(79)90068-2