A01 Percolation Part 2

Start Assignment

Due Saturday by 11:59pm **Points** 50 **Submitting** a text entry box or a file upload
Available Jan 15 at 12am - Feb 5 at 11:59pm 22 days

Fundamentals

Assignment: Percolation Part 2

Learning Objectives

- Select the appropriate data structure(s) to solve a problem given a set of programming specifications.
- Implement and use a Monte Carlo simulation to estimate the percolation threshold

Overview

In this second part of the assignment, you will ensure that class Percolation is complete and implement the Monte Carlo simulation.

Instruction

Continue the project you started in part 1 of this assignment. Implement the class **PercolationStats,** and perform the necessary testing to ensure that all requirements of the assignment are met.

Assignment Instructions:

The assignment instructions **A01 Percolation.pdf** (https://slcc.instructure.com/courses/773686/files/126485212/download?download_frd=1) are based on an assignment of Princeton's algorithm course. They come with additional resources listed below.

JUnit tests for the class PercolationStats are available on CodePost. You can submit as often as you like up until the deadline. Before you submit via Canvas, create a screenshot as described before.

Please note, that the JUnit tests on CodePost are only a subset of the tests that are used for grading. They are intended to facilitate your testing but they don't cover all cases.
It is the responsibility of both team partners to complete the testing and to ensure that all assignment requirements are met.

Additional Resources:

Important: Whenever there is a difference or inconsistency between the assignment instructions (pdf) and one of the additional resources, the instructions need to be followed.

Why are there occasional inconsistencies between the instructions and the resources?
It helps you prepare for the workforce. It is often possible to find resources that include helpful information, but those resources are rarely targeted to a specific task a company asked you to perform. It is an important skill to identify information that is relevant and to discern information that does not apply to a given task.

Checklist:
Here is a **checklist** that includes frequently asked questions, test clients, and possible progress steps.

Videos:

Here is a **video from Professor Sedgewick** (https://cuvids.io/app/video/101/watch) where he introduces percolation as an example

Here is another video that you might find helpful. The beginning shows the course site from Princeton. Even though your instructions are based on theirs there are some important differences. Accessing information directly from the Princeton course site might provide you with information that does not match the requirements of this course.

video (https://www.youtube.com/watch?v=o60oHXesOuA)

(https://www.youtube.com/watch?v=o60oHXesOuA)

To keep the scope more manageable I removed the requirement to analyze the runtime and memory usage.

Submission

One team member embeds the screenshot from CodePost **and** attaches the required java files (no zip, no JAR). **How to embed images in Canvas** (https://community.canvaslms.com/t5/Student-Guide/How-do-I-embed-images-from-Canvas-into-the-Rich-Content-Editor/ta-p/356)

Both team members submit the name of the partner and the discussed pebble distribution. If the pebble distribution is not 50/50, include a description that explains the difference.

A01 - Part 2		
Criteria	**Ratings**	**Pts**
JUnit tests for class PercolationStats Full points if all tests pass. -4 points for each failed test. 1 failed test (26), 2 failed tests (22), 3 failed tests(18), etc.		30 pts
Functional and Performance Requirements Additional tests are executed to test whether the functional and performance requirements are met. Points are deducted based on the severity of the issue(s).		15 pts
Style | Best Practices 1 point is deducted for each deviation of the Style Guide. E.g. -1 point for indentation issues, -1 point for variable names that are not descriptive, -1 point for doc comments that miss the 'summary description', etc. There are also point deductions if best practices are ignored like avoiding code duplication.		5 pts
		Total Points: 50