Programming Assignment: Percolation

Passed · 90/100 points

|  |  |
| --- | --- |
| **Deadline** | Pass this assignment by December 4, 11:59 PM PST |

1. [**Instructions**](https://www.coursera.org/learn/introduction-to-algorithms/programming/Lhp5z/percolation)
2. [My submission](https://www.coursera.org/learn/introduction-to-algorithms/programming/Lhp5z/percolation/submission)
3. [Discussions](https://www.coursera.org/learn/introduction-to-algorithms/programming/Lhp5z/percolation/discussions)

**Specification**

Here is the programming assignment [specification](http://coursera.cs.princeton.edu/algs4/assignments/percolation.html) that describes the assignment requirements.

Be sure that your code conforms to the prescribed APIs: each program must be in the "default" package (i.e., no **package** statements) and include only the public methods and constructors specified (extra **private** methods are fine). Note that **algs4.jar** uses a "named" package, so you must use an **import** statement to access a class in **algs4.jar**.

**Checklist**

The [checklist](http://coursera.cs.princeton.edu/algs4/checklists/percolation.html) contains frequently asked questions and hints. If you're not sure where to start, see the section at the end of the checklist.

**Testing**

The file [percolation-testing.zip](http://coursera.cs.princeton.edu/algs4/testing/percolation-testing.zip) contains sample data files and programs that you can use to test Percolation.java.

**Web Submission**

Submit a zip file named **percolation.zip** that contains only the two source files **Percolation.java** and **PercolationStats.java**. Here are three approaches to creating the zip file:

Mac OS X.

* Select the required files in the Finder.
* Right-click and select *Compress 2 Items*.
* Rename the resulting file to **percolation.zip**.

Windows.

* Select the required files in Windows Explorer.
* Right-click and select *Send to -> Compressed (zipped) folder*.
* Rename the resulting file to **percolation** (the .zip extension is automatic).

Command line (Linux or Mac OS X).

* Change to the directory containing the required .java files.
* Execute the command: **zip percolation.zip Percolation.java PercolationStats.java**

**Assessment Report**

Here is some information to help you interpret the assessment report. See the [Assessment Guide](https://www.coursera.org/learn/java-data-structures-algorithms-1/resources/R2mre) for more details.

* *Compilation*: we compile your .java files using a Java 8 compiler. Any error or warning messages are displayed and usually signify a major defect in your code. If your program does not compile, no further tests are performed.
* API: we check that your code exactly matches the prescribed API (no extra methods and no missing methods). If it does not, no further tests are performed.
* *Style*: we run [checkstyle](http://checkstyle.sourceforge.net/" \t "_blank) to automatically checks the style of your Java programs. Here is a list of available [Checkstyle checks](http://checkstyle.sourceforge.net/checks.html" \t "_blank), which you can use to help decode any warning messages.
* *Bugs*: we run [findbugs](http://findbugs.sourceforge.net/" \t "_blank) to check for common bug patterns in Java programs. A warning message strongly suggests a bug in your code but occasionally there are false positives. Here is a summary of [bug descriptions](http://findbugs.sourceforge.net/bugDescriptions.html), which you can use to help decode warning messages.
* *Correctness*: we perform a battery of unit tests to check that your code meets the specifications.
* *Memory*: we determine the amount of memory according to the 64-bit memory cost model from lecture.
* *Timing*: we measure the running time and count the number of elementary operations.