Programming Assignment 1: Percolation | percolation.zip

Submission	
Submission time	Sat-26-Oct 03:16:12
Raw Score	75.50 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  14/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 75.50% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 10:16 Percolation.java
-rw-r--r-- 1 2.6K Oct 26 10:16 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 10:16 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************
```

Submission

```
% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================



% checkstyle *.java
*-----------------------------------------------------------
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================


*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
```

Submission

```
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed
```

Submission

```
Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
```

Submission

```
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
     isFull(5, 1) returns wrong value [after 17 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
  *  N = 10
```

Submission

```
     isFull(10, 1) returns wrong value [after 69 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 365 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 2) returns wrong value [after 1555 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     isFull(3, 1) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  N = 5
  *  N = 7
     isFull(7, 1) returns wrong value [after 32 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(8, 2) returns wrong value [after 61 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 20
  *  N = 50
     isFull(41, 1) returns wrong value [after 1543 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED
```

Submission

```
Total: 9/13 tests passed!
================================================================

Testing methods in PercolationStats
*-----------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.602 (passed)
        stddev():  0.019 (passed)

        Overall result: passed

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.669 (FAILED, outside of range)
        stddev():  0.071 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.587 (passed)
        confidenceHi():  0.607 (passed)
==> passed

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5857135008255461
  *  confidenceHi() = 0.6900908395876016
==> FAILED
```

Submission

```
Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
  *  N =  23, T =  0
  *  N = -42, T =  0
==> passed

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
==> passed

Total: 5/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed
```

Submission

```
Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97400 (380.5x)
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.
```

Submission

```
Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                   250
                1
=> passed       32     0.00          828                  3092
                1
=> passed      128     0.03        11554                 48006
                1
=> passed      512     0.10       186371                785726
                1
=> passed     1024     0.29       730968               3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.


Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
```

Submission	

```
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 03:15:37
Raw Score	75.50 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  14/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed
```

Submission

```
Raw score: 75.50% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 10:15 Percolation.java
-rw-r--r-- 1 2.6K Oct 26 10:15 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 10:15 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================



% checkstyle *.java
*-----------------------------------------------------------
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
```

Submission

```
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed
```

Submission

```
Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
```

Submission

```
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed
```

Submission

```
Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
     isFull(7, 1) returns wrong value [after 31 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(10, 1) returns wrong value [after 82 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 244 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 1) returns wrong value [after 1916 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
     isFull(5, 5) returns wrong value [after 14 total calls to op
en()]
     - student   = true
```

Submission

```
     - reference = false
  *  N = 7
     isFull(7, 7) returns wrong value [after 20 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(10, 1) returns wrong value [after 79 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(13, 13) returns wrong value [after 261 total calls to
 open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(41, 2) returns wrong value [after 1515 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED


Total: 9/13 tests passed!
================================================================


Testing methods in PercolationStats
*-----------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.602 (passed)
        stddev():  0.019 (passed)

        Overall result: passed
```

Submission

```
Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    0.656 (passed)
        stddev():  0.070 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.588 (passed)
        confidenceHi():  0.609 (passed)
==> passed

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5867731242576526
  *  confidenceHi() = 0.6928921653193472
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
  *  N =  23, T =  0
  *  N = -42, T =  0
==> passed

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
==> passed

Total: 5/7 tests passed!
```

Submission

```
================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97400 (380.5x)
```

Submission

```
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*------------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                     250
                1
=> passed       32     0.00          828                    3092
```

Submission

```
                1
=> passed      128     0.02        11554                 48006
                1
=> passed      512     0.11       186371                785726
                1
=> passed     1024     0.30       730968               3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission

Submission

Submission time	Sat-26-Oct 03:13:46
Raw Score	75.50 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  14/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 75.50% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 10:13 Percolation.java
-rw-r--r-- 1 2.6K Oct 26 10:13 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 10:13 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
```

Submission

```
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
```

Submission

```
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed
```

Submission

```
Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed
```

Submission

```
Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
     isFull(5, 1) returns wrong value [after 15 total calls to op
en()]
      - student   = true
      - reference = false
  *  N = 7
```

Submission

```
     isFull(7, 1) returns wrong value [after 33 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
  *  N = 20
     isFull(20, 2) returns wrong value [after 310 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 2) returns wrong value [after 1705 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
     isFull(5, 5) returns wrong value [after 16 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
  *  N = 10
  *  N = 20
     isFull(20, 17) returns wrong value [after 273 total calls to
 open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(48, 9) returns wrong value [after 1590 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED


Total: 9/13 tests passed!
================================================================
```

Submission

```
Testing methods in PercolationStats
*-----------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.603 (passed)
        stddev():  0.021 (passed)

        Overall result: passed

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.654 (passed)
        stddev():  0.070 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.590 (passed)
        confidenceHi():  0.612 (passed)
==> passed

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5891444077640876
  *  confidenceHi() = 0.6924945785113203
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
```

Submission

```
  *  N = -23, T = 42
  *  N =  23, T =  0
  *  N = -42, T =  0
==> passed

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
==> passed

Total: 5/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)
```

Submission

```
Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97400 (380.5x)
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.
```

Submission

```
For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                2 * connected()
                 N   seconds       union()             + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                    250
                1
=> passed       32     0.00          828                   3092
                1
=> passed      128     0.02        11554                  48006
                1
=> passed      512     0.11       186371                 785726
                1
=> passed     1024     0.30       730968                3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.


Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().
```

Submission

```
                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4                0               1
         1
=> passed      128        4                0               1
         1
=> passed      512        4                0               1
         1
=> passed     1024        4                0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 03:13:30
Raw Score	75.50 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  14/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 75.50% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Submission

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 10:13 Percolation.java
-rw-r--r-- 1 2.6K Oct 26 10:13 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 10:13 studentSubmission.zip


******************************************************************
*************
*  compiling
******************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
=================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
=================================================================



% checkstyle *.java
*-----------------------------------------------------------
=================================================================


% findbugs *.class
*-----------------------------------------------------------
=================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:
```

Submission

```
PercolationStats:

================================================================



******************************************************************
*************
*  executing
******************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
```

Submission

```
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
```

Submission

```
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
```

Submission

```
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
     isFull(7, 1) returns wrong value [after 38 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(10, 1) returns wrong value [after 87 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 326 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 1) returns wrong value [after 2189 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
  *  N = 7
     isFull(6, 6) returns wrong value [after 25 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
```

Submission

```
  *  N = 20
     isFull(19, 16) returns wrong value [after 278 total calls to
 open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(38, 49) returns wrong value [after 1487 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED


Total: 9/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.604 (passed)
        stddev():  0.020 (passed)

        Overall result: passed

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.660 (FAILED, outside of range)
        stddev():  0.070 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats
```

Submission

```
Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.588 (passed)
        confidenceHi():  0.609 (passed)
==> passed

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5790816683961664
  *  confidenceHi() = 0.68402407198536
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
  *  N =  23, T =  0
  *  N = -42, T =  0
==> passed

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
==> passed

Total: 5/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*------------------------------------------------------------
Running 4 total tests.
```

Submission

```
Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*------------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97400 (380.5x)
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================
```

Submission

```
*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                   2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                     250
                1
=> passed       32     0.00          828                    3092
                1
=> passed      128     0.02        11554                   48006
                1
=> passed      512     0.12       186371                  785726
                1
=> passed     1024     0.25       730968                 3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
```

Submission

```
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 03:01:55
Raw Score	72.25 / 100.00

Submission

Feedback

See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  13/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 72.25% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 10:02 Percolation.java
-rw-r--r-- 1 2.5K Oct 26 10:02 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 10:02 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
=================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
=================================================================
```

Submission

```
% checkstyle *.java
*-----------------------------------------------------------
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================


*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
```

Submission

```
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
```

Submission

```
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
```

Submission

```
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
     isFull(10, 1) returns wrong value [after 54 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 359 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 1) returns wrong value [after 2288 total calls to
 open()]
     - student   = true
     - reference = false
```

Submission

```
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
     isFull(8, 10) returns wrong value [after 42 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(18, 7) returns wrong value [after 210 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(39, 47) returns wrong value [after 1506 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED


Total: 9/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.608 (passed)
        stddev():  0.021 (passed)

        Overall result: passed
```

Submission

```
Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    0.653 (passed)
        stddev():  0.070 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.589 (passed)
        confidenceHi():  0.612 (passed)
==> passed

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5838202
  *  confidenceHi() = 0.6894515133184068
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
```

Submission

```
==> passed

Total: 4/7 tests passed!

=================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

=================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)
```

Submission

```
                 T        bytes
--------------------------------------------
=> FAILED       16        97400 (380.5x)
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*------------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
```

Submission

```
----------------------------
=> passed        8     0.00          86                    250
                1
=> passed       32     0.00         828                   3092
                1
=> passed      128     0.02       11554                  48006
                1
=> passed      512     0.10      186371                 785726
                1
=> passed     1024     0.30      730968                3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.


Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
```

Submission

```
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 02:59:21
Raw Score	72.25 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  13/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 72.25% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:59 Percolation.java
-rw-r--r-- 1 2.5K Oct 26 09:59 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:59 studentSubmission.zip
```

Submission

```
*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
```

Submission

```
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
```

Submission

```
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
```

Submission

```
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
     isFull(5, 1) returns wrong value [after 20 total calls to op
```

Submission

```
en()]
     - student   = true
     - reference = false
  *  N = 7
  *  N = 10
     isFull(10, 3) returns wrong value [after 65 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 335 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 1) returns wrong value [after 2275 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
     isFull(5, 1) returns wrong value [after 19 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
  *  N = 10
     isFull(9, 8) returns wrong value [after 47 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(8, 20) returns wrong value [after 216 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(39, 6) returns wrong value [after 1358 total calls to
 open()]
```

Submission

```
     - student   = true
     - reference = false
==> FAILED


Total: 9/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.607 (passed)
        stddev():  0.020 (passed)

        Overall result: passed

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.653 (passed)
        stddev():  0.069 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
PercolationStats reports:
        confidenceLo():    0.590 (passed)
        confidenceHi():  0.610 (passed)
==> passed
```

Submission

```
Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.5845893476937963
  *  confidenceHi() = 0.6887509088948529
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
==> passed

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  order = stddev(), mean()
==> passed

Total: 4/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*------------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
```

Submission

```
-------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
-------------------------------------------
=> FAILED       16        97400 (380.5x)
=> FAILED       32        97464 (253.8x)
=> FAILED       64        97592 (152.5x)
=> FAILED      128        97848  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97336.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
```

Submission

```
*  timing
*****************************************************************
*************

Timing Percolation
*------------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                     250
                1
=> passed       32     0.00          828                    3092
                1
=> passed      128     0.02        11554                   48006
                1
=> passed      512     0.12       186371                  785726
                1
=> passed     1024     0.26       730968                 3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
```

Submission

```
indicates that it uses 9.6x too many calls.


Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 02:49:15
Raw Score	59.25 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
```

Submission

```
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  9/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 59.25% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:49 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:49 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:49 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
================================================================
```

Submission

```
% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
```

Submission

```
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
```

Submission

```
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
```

Submission

```
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
     isFull(7, 1) returns wrong value [after 32 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(10, 2) returns wrong value [after 71 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 228 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
```

Submission

```
     isFull(3, 4) returns wrong value [after 13 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
     isFull(7, 6) returns wrong value [after 32 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(7, 9) returns wrong value [after 61 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(19, 20) returns wrong value [after 293 total calls to
 open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(47, 12) returns wrong value [after 1529 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED



Total: 9/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    6066.918 (FAILED, outside of range)
        stddev():  195.207 (FAILED, outside of range)
```

Submission

```
        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    26161.627 (FAILED, outside of range)
        stddev():  2796.918 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 5844.820924506238
  *  confidenceHi() = 6076.954737239822
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22073.472162918228
  *  confidenceHi() = 27965.370968061077
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 59.848425
  *  2mean = 58.61
  *  1mean = 237.565419
  *  2mean = 232.76
```

Submission

```
  *  1mean = 59.814062875
  *  2mean = 58.31
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 531.854204875; stddev = 32.638551940072425
  *  order = stddev(), mean()
  *  mean = 536.8542042500001; stddev = 33.50322749954518
==> FAILED

Total: 0/7 tests passed!

================================================================

****************************************************************
*************
*  memory usage
****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================
```

Submission

```
Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97392 (380.4x)
=> FAILED       32        97456 (253.8x)
=> FAILED       64        97584 (152.5x)
=> FAILED      128        97840  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97328.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
```

Submission

```
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                     2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00          86                    250
                1
=> passed       32     0.00         828                   3092
                1
=> passed      128     0.02       11554                  48006
                1
=> passed      512     0.11      186371                 785726
                1
=> passed     1024     0.25      730968                3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
```

Submission

```
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 02:47:10
Raw Score	59.25 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  9/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 59.25% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

Submission

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:47 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:47 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:47 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================



% checkstyle *.java
*-----------------------------------------------------------
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================
```

Submission

```
*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> passed

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
```

Submission

```
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
```

Submission

```
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
```

Submission

```
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
     isFull(20, 2) returns wrong value [after 286 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 2) returns wrong value [after 1884 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
     isFull(10, 7) returns wrong value [after 70 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(17, 7) returns wrong value [after 233 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(35, 7) returns wrong value [after 1500 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED
```

Submission

```
Total: 9/13 tests passed!
================================================================

Testing methods in PercolationStats
*-----------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    6095.245 (FAILED, outside of range)
        stddev():  184.363 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    26144.961 (FAILED, outside of range)
        stddev():  2789.449 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 5898.468913860101
  *  confidenceHi() = 6131.285749001362
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22104.255033290985
  *  confidenceHi() = 28004.246657702053
```

Submission

```
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 60.656504999999996
  *  2mean = 57.93
  *  1mean = 239.61592199999998
  *  2mean = 237.53
  *  1mean = 59.76883675
  *  2mean = 59.06
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 537.08536; stddev = 32.922804747780226
  *  order = stddev(), mean()
  *  mean = 534.9848577500001; stddev = 36.337070794219606
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.
```

Submission

```
Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97392 (380.4x)
=> FAILED       32        97456 (253.8x)
=> FAILED       64        97584 (152.5x)
=> FAILED      128        97840  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97328.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================
```

Submission

```
*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                     250
                1
=> passed       32     0.00          828                    3092
                1
=> passed      128     0.02        11554                   48006
                1
=> passed      512     0.12       186371                  785726
                1
=> passed     1024     0.29       730968                 3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
```

Submission

```
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 02:38:58
Raw Score	56.00 / 100.00

Submission

Feedback

See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  8/20 tests passed
Memory:       4/8 tests passed
Timing:       9/9 tests passed

Raw score: 56.00% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:38 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:38 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:38 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*------------------------------------------------------------
================================================================

% javac PercolationStats.java
*------------------------------------------------------------
================================================================
```

Submission

```
% checkstyle *.java
*-----------------------------------------------------------
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
     - IndexOutOfBoundsException NOT thrown for isOpen()
```

Submission

```
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
```

Submission

```
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed
```

Submission

```
Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
     isFull(5, 2) returns wrong value [after 17 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
  *  N = 10
     isFull(10, 1) returns wrong value [after 84 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(20, 1) returns wrong value [after 341 total calls to
```

Submission

```
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 2) returns wrong value [after 2221 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
     isFull(5, 5) returns wrong value [after 16 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
     isFull(5, 2) returns wrong value [after 29 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(5, 1) returns wrong value [after 69 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(17, 13) returns wrong value [after 226 total calls to
 open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(40, 4) returns wrong value [after 1453 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED


Total: 8/13 tests passed!
================================================================
```

Submission

```
Testing methods in PercolationStats
*-----------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    6064.755 (FAILED, outside of range)
        stddev():  199.935 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    26217.849 (FAILED, outside of range)
        stddev():  2792.823 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 5876.748227225469
  *  confidenceHi() = 6109.321681103487
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22069.74205610429
  *  confidenceHi() = 27939.56406647413
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
```

Submission

```
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 59.757515999999995
  *  2mean = 61.58
  *  1mean = 241.34319299999999
  *  2mean = 235.79
  *  1mean = 59.140696125
  *  2mean = 58.91
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 536.115510875; stddev = 33.37725989977465
  *  order = stddev(), mean()
  *  mean = 532.788878125; stddev = 33.41039198776254
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
```

Submission

```
-------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
-------------------------------------------
=> FAILED       16        97392 (380.4x)
=> FAILED       32        97456 (253.8x)
=> FAILED       64        97584 (152.5x)
=> FAILED      128        97840  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97328.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
```

Submission

```
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.



For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                   250
                1
=> passed       32     0.00          828                  3092
                1
=> passed      128     0.02        11554                 48006
                1
=> passed      512     0.12       186371                785726
                1
=> passed     1024     0.27       730968               3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
```

Submission

```
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32         4               0               1
         1
=> passed      128         4               0               1
         1
=> passed      512         4               0               1
         1
=> passed     1024         4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 02:32:46
Raw Score	56.00 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
```

Submission

```
Style:         PASSED
Findbugs:      No potential bugs found.
API:           PASSED

Correctness:   8/20 tests passed
Memory:        4/8 tests passed
Timing:        9/9 tests passed

Raw score: 56.00% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:32 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:32 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:32 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*------------------------------------------------------------
=================================================================

% javac PercolationStats.java
*------------------------------------------------------------
=================================================================




% checkstyle *.java
*------------------------------------------------------------
=================================================================
```

Submission

```
% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************


Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for isOpen()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
     - IndexOutOfBoundsException NOT thrown for isOpen()
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
```

Submission

```
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
  *  filename = input8.txt
  *  filename = input8-no.txt
  *  filename = input10-no.txt
  *  filename = greeting57.txt
  *  filename = heart25.txt
==> passed

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
  *  N = 5
  *  N = 10
  *  N = 10
  *  N = 20
  *  N = 20
  *  N = 50
  *  N = 50
==> passed

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
  *  filename = input1-no.txt
  *  filename = input2.txt
  *  filename = input2-no.txt
==> passed

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     isFull(18, 1) returns wrong value [after 231 total calls to
open()]
     - student   = true
     - reference = false
  *  filename = input10.txt
     isFull(9, 1) returns wrong value [after 56 total calls to op
en()]
     - student   = true
     - reference = false
  *  filename = input50.txt
```

Submission

```
     isFull(22, 28) returns wrong value [after 1412 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     isFull(3, 1) returns wrong value [after 4 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input4.txt
     isFull(4, 4) returns wrong value [after 7 total calls to ope
n()]
     - student   = true
     - reference = false
  *  filename = input7.txt
     isFull(6, 1) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
  *  filename = snake101.txt
==> passed

Test 8: Opens every site
  *  filename = input5.txt
==> passed

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
==> passed

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
  *  filename = input8.txt;  order =     isFull(), percolates(),
```

Submission

```
    isOpen()
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
==> passed

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
  *  N = 5
  *  N = 7
  *  N = 10
  *  N = 20
  *  N = 50
==> passed

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
  *  N = 5
  *  N = 7
     isFull(7, 1) returns wrong value [after 28 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
  *  N = 20
     isFull(20, 1) returns wrong value [after 339 total calls to
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(50, 1) returns wrong value [after 1667 total calls to
 open()]
     - student   = true
     - reference = false
==> FAILED
```

Submission

```
Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
  *  N = 5
     isFull(5, 4) returns wrong value [after 12 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 7
     isFull(6, 1) returns wrong value [after 23 total calls to op
en()]
     - student   = true
     - reference = false
  *  N = 10
     isFull(10, 3) returns wrong value [after 50 total calls to o
pen()]
     - student   = true
     - reference = false
  *  N = 20
     isFull(13, 9) returns wrong value [after 246 total calls to 
open()]
     - student   = true
     - reference = false
  *  N = 50
     isFull(41, 13) returns wrong value [after 1519 total calls t
o open()]
     - student   = true
     - reference = false
==> FAILED


Total: 8/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------
```

Submission

```
PercolationStats reports:
        mean():    6047.122 (FAILED, outside of range)
        stddev():  202.725 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    26356.848 (FAILED, outside of range)
        stddev():  2836.426 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 5869.679304881756
  *  confidenceHi() = 6092.424796492741
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22601.24232979259
  *  confidenceHi() = 28616.15430406328
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
```

Submission

```
  *  1mean = 59.545395
  *  2mean = 59.48
  *  1mean = 239.99976000000004
  *  2mean = 237.13
  *  1mean = 58.24119875
  *  2mean = 59.565
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 537.301440375; stddev = 32.80296125782147
  *  order = stddev(), mean()
  *  mean = 534.7336015; stddev = 31.747648718685976
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)
```

Submission

```
Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97392 (380.4x)
=> FAILED       32        97456 (253.8x)
=> FAILED       64        97584 (152.5x)
=> FAILED      128        97840  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97328.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
```

Submission

```
              find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
=> passed        8     0.00           86                    250
                1
=> passed       32     0.00          828                   3092
                1
=> passed      128     0.02        11554                  48006
                1
=> passed      512     0.11       186371                 785726
                1
=> passed     1024     0.22       730968                3100964
                1
==> 5/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().
```

Submission

```
                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
=> passed       32        4               0               1
         1
=> passed      128        4               0               1
         1
=> passed      512        4               0               1
         1
=> passed     1024        4               0               1
         1
==> 4/4 tests passed

Total: 9/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 02:21:38
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
```

Submission

```
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:21 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:21 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:21 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================



% checkstyle *.java
*-----------------------------------------------------------
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:
```

Submission

```
PercolationStats:

================================================================



****************************************************************
*************
*  executing
****************************************************************
*************


Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test2(TestPercolation.java:201)
          TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
    *   N = 3
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:235)
          TestPercolation.main(TestPercolation.java:541)

    *   N = 5
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:236)
          TestPercolation.main(TestPercolation.java:541)

    *   N = 10
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:237)
          TestPercolation.main(TestPercolation.java:541)

    *   N = 10
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
```

Submission

```
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:238)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 20
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:239)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 20
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:240)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 50
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:241)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 50
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:242)
      TestPercolation.main(TestPercolation.java:541)

==> FAILED


Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
```

Submission

```
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:253)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:254)
     TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:263)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input10.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:264)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input50.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:265)
     TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
```

Submission

```
      TestPercolation.test6(TestPercolation.java:276)
      TestPercolation.main(TestPercolation.java:544)

   *  filename = input7.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test6(TestPercolation.java:277)
      TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
   *  filename = snake13.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test7(TestPercolation.java:287)
      TestPercolation.main(TestPercolation.java:545)

   *  filename = snake101.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test7(TestPercolation.java:288)
      TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
   *  filename = input5.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isFull(Percolation.java:49)
      TestPercolation.checkIsFull(TestPercolation.java:22)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test8(TestPercolation.java:296)
```

Submission

```
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
```

Submission

```
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
```

Submission

```
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
```

Submission

```
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold
```

Submission

```
Creating new PercolationStats(100, 50)
------------------------------------------------

PercolationStats reports:
        mean():    6063.408 (FAILED, outside of range)
        stddev():  198.290 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    26144.294 (FAILED, outside of range)
        stddev():  2770.098 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 5860.494716862262
  *  confidenceHi() = 6092.318657323769
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22287.866684051463
  *  confidenceHi() = 28207.28489808384
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED
```

Submission

```
Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 60.474686999999996
  *  2mean = 58.36
  *  1mean = 238.50481200000002
  *  2mean = 237.59
  *  1mean = 59.457279
  *  2mean = 58.87
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 534.366767375; stddev = 32.54823511493913
  *  order = stddev(), mean()
  *  mean = 536.1758123750001; stddev = 31.958938425449247
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*------------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed
```

Submission

```
Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16        97392 (380.4x)
=> FAILED       32        97456 (253.8x)
=> FAILED       64        97584 (152.5x)
=> FAILED      128        97840  (84.9x)
==> 0/4 tests passed


Estimated student memory = 4.00 T + 97328.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.
```

Submission

```
Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity           21                     30
               1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity           68   (0.5x)            30
(0.2x)         1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity          263   (0.1x)           230
(0.1x)         1
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity         1029   (0.0x)         1554
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3276
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission

Submission

Submission time	Sat-26-Oct 02:06:56
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 09:07 Percolation.java
-rw-r--r-- 1 2.4K Oct 26 09:07 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 09:07 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
```

Submission

```
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



******************************************************************
*************
*  executing
******************************************************************
*************


Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
```

Submission

```
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
```

Submission

```
     TestPercolation.test3(TestPercolation.java:235)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:236)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:237)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:238)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:239)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:240)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:241)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:242)
     TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
```

Submission

```
          TestPercolation.main(TestPercolation.java:542)

       *  filename = input2.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test4(TestPercolation.java:253)
          TestPercolation.main(TestPercolation.java:542)

       *  filename = input2-no.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test4(TestPercolation.java:254)
          TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
       *  filename = input20.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:263)
          TestPercolation.main(TestPercolation.java:543)

       *  filename = input10.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:264)
          TestPercolation.main(TestPercolation.java:543)

       *  filename = input50.txt
          java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:265)
          TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:276)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input7.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:277)
     TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
```

Submission

```
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
```

Submission

```
37)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 50
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.test1(TestPercolationStats.java:17)
     TestPercolationStats.main(TestPercolationStats.java:223)


Creating new PercolationStats(200, 10)
------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 10
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.test1(TestPercolationStats.java:17)
     TestPercolationStats.main(TestPercolationStats.java:224)
```

Submission

```
Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
-----------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 50
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.test1c(TestPercolationStats.java:69)
     TestPercolationStats.main(TestPercolationStats.java:232)

==> FAILED

Creating new PercolationStats(200, 10)
-----------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 10
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.test1c(TestPercolationStats.java:69)
     TestPercolationStats.main(TestPercolationStats.java:233)

==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException: 100
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
     TestPercolationStats.test3(TestPercolationStats.java:169)
     TestPercolationStats.main(TestPercolationStats.java:236)

     java.lang.ArrayIndexOutOfBoundsException: 100
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
```

Submission

```
     TestPercolationStats.test3(TestPercolationStats.java:170)
     TestPercolationStats.main(TestPercolationStats.java:236)

     java.lang.ArrayIndexOutOfBoundsException: 200
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
     TestPercolationStats.test3(TestPercolationStats.java:171)
     TestPercolationStats.main(TestPercolationStats.java:236)

==> FAILED

Test 4: Call the methods of PercolationStats in either order.
     java.lang.ArrayIndexOutOfBoundsException: 200
     PercolationStats.<init>(PercolationStats.java:24)
     TestPercolationStats.test4(TestPercolationStats.java:182)
     TestPercolationStats.main(TestPercolationStats.java:237)

==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
```

Submission

```
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsExcepti
on: 16
        at PercolationStats.<init>(PercolationStats.java:24)
        at MemoryOfPercolationStats.main(MemoryOfPercolationStats
.java:81)
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------

Total: 0/4 tests passed:Test aborted. Ran out of time or crashed
before completion.
================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.
```

Submission

```
Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity          21                    30
               1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity          68   (0.5x)           30
(0.2x)         1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity         263   (0.1x)          230
(0.1x)         1
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity         1029   (0.0x)         1554
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3276
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission

Submission

Submission time	Sat-26-Oct 01:56:10
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 08:56 Percolation.java
-rw-r--r-- 1 2.5K Oct 26 08:56 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 08:56 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
```

Submission

```
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



******************************************************************
*************
*  executing
******************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
```

Submission

```
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test2(TestPercolation.java:198)
         TestPercolation.main(TestPercolation.java:540)

    *    filename = input10-no.txt
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test2(TestPercolation.java:199)
         TestPercolation.main(TestPercolation.java:540)

    *    filename = greeting57.txt
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test2(TestPercolation.java:200)
         TestPercolation.main(TestPercolation.java:540)

    *    filename = heart25.txt
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test2(TestPercolation.java:201)
         TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
    *    N = 3
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.random(TestPercolation.java:214)
```

Submission

```
    TestPercolation.test3(TestPercolation.java:235)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 5
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:236)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 10
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:237)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 10
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:238)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 20
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:239)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 20
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.random(TestPercolation.java:214)
         TestPercolation.test3(TestPercolation.java:240)
         TestPercolation.main(TestPercolation.java:541)

      *  N = 50
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.random(TestPercolation.java:214)
         TestPercolation.test3(TestPercolation.java:241)
         TestPercolation.main(TestPercolation.java:541)

      *  N = 50
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.random(TestPercolation.java:214)
         TestPercolation.test3(TestPercolation.java:242)
         TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
      *  filename = input1.txt
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test4(TestPercolation.java:251)
         TestPercolation.main(TestPercolation.java:542)

      *  filename = input1-no.txt
         java.lang.ArrayIndexOutOfBoundsException
         Percolation.isFull(Percolation.java:49)
         TestPercolation.checkIsFull(TestPercolation.java:22)
         TestPercolation.check(TestPercolation.java:75)
         TestPercolation.file(TestPercolation.java:138)
         TestPercolation.test4(TestPercolation.java:252)
```

Submission

```
          TestPercolation.main(TestPercolation.java:542)

       *  filename = input2.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test4(TestPercolation.java:253)
          TestPercolation.main(TestPercolation.java:542)

       *  filename = input2-no.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test4(TestPercolation.java:254)
          TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
       *  filename = input20.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:263)
          TestPercolation.main(TestPercolation.java:543)

       *  filename = input10.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:264)
          TestPercolation.main(TestPercolation.java:543)

       *  filename = input50.txt
          java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:265)
          TestPercolation.main(TestPercolation.java:543)

     ==> FAILED

     Test 6: Check for backwash with predetermined sites that havemult
     iple percolating paths
       *  filename = input3.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:275)
          TestPercolation.main(TestPercolation.java:544)

       *  filename = input4.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:276)
          TestPercolation.main(TestPercolation.java:544)

       *  filename = input7.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:277)
          TestPercolation.main(TestPercolation.java:544)

     ==> FAILED

     Test 7: Predetermined sites with very long percolating path
       *  filename = snake13.txt
          java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     isFull(1, 2) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
```

Submission

```
  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)
```

Submission

```
==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
    *  N = 50
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.randomCalls(TestPercolation.java:496)
       TestPercolation.test13(TestPercolation.java:522)
       TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
=================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
-------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 50
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.test1(TestPercolationStats.java:17)
     TestPercolationStats.main(TestPercolationStats.java:223)


Creating new PercolationStats(200, 10)
-------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 10
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.test1(TestPercolationStats.java:17)
     TestPercolationStats.main(TestPercolationStats.java:224)




Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
-------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 50
```

Submission

```
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.test1c(TestPercolationStats.java:69)
     TestPercolationStats.main(TestPercolationStats.java:232)

==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
     java.lang.ArrayIndexOutOfBoundsException: 10
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.test1c(TestPercolationStats.java:69)
     TestPercolationStats.main(TestPercolationStats.java:233)

==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException: 100
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
     TestPercolationStats.test3(TestPercolationStats.java:169)
     TestPercolationStats.main(TestPercolationStats.java:236)

     java.lang.ArrayIndexOutOfBoundsException: 100
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
     TestPercolationStats.test3(TestPercolationStats.java:170)
     TestPercolationStats.main(TestPercolationStats.java:236)

     java.lang.ArrayIndexOutOfBoundsException: 200
     PercolationStats.<init>(PercolationStats.java:20)
```

Submission

```
     TestPercolationStats.twoPercolationStats(TestPercolationStat
s.java:132)
     TestPercolationStats.test3(TestPercolationStats.java:171)
     TestPercolationStats.main(TestPercolationStats.java:236)

==> FAILED

Test 4: Call the methods of PercolationStats in either order.
     java.lang.ArrayIndexOutOfBoundsException: 200
     PercolationStats.<init>(PercolationStats.java:20)
     TestPercolationStats.test4(TestPercolationStats.java:182)
     TestPercolationStats.main(TestPercolationStats.java:237)

==> FAILED

Total: 0/7 tests passed!

=================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)
```

Submission

```
Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsExcepti
on: 16
        at PercolationStats.<init>(PercolationStats.java:20)
        at MemoryOfPercolationStats.main(MemoryOfPercolationStats
.java:81)
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------

Total: 0/4 tests passed:Test aborted. Ran out of time or crashed
before completion.
================================================================



******************************************************************
*************
*  timing
******************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.
```

Submission

```
For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                N   seconds       union()             + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity           21                     30
               1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity           68   (0.5x)            30
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity          263   (0.1x)           230
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
```

Submission

```
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity         1029   (0.0x)         1554
(0.0x)         1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3276
(0.0x)         1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
```

Submission

```
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 01:51:48
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        FAILED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 08:51 Percolation.java
-rw-r--r-- 1 2.5K Oct 26 08:51 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 08:51 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
=================================================================
```

Submission

```
% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
PercolationStats.java:18:17: Assignment of parameter 'T' is not a
llowed.
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************


Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
```

Submission

```
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)
```

Submission

```
  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:235)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:236)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 10
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:237)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 10
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:238)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 20
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:239)
    TestPercolation.main(TestPercolation.java:541)

 *  N = 20
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.random(TestPercolation.java:214)
    TestPercolation.test3(TestPercolation.java:240)
    TestPercolation.main(TestPercolation.java:541)
```

Submission

```
  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:241)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:242)
     TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:253)
    TestPercolation.main(TestPercolation.java:542)

  * filename = input2-no.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:254)
    TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
  * filename = input20.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:263)
    TestPercolation.main(TestPercolation.java:543)

  * filename = input10.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:264)
    TestPercolation.main(TestPercolation.java:543)

  * filename = input50.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:265)
```

Submission

```
          TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
    *  filename = input3.txt
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.check(TestPercolation.java:75)
       TestPercolation.file(TestPercolation.java:138)
       TestPercolation.test6(TestPercolation.java:275)
       TestPercolation.main(TestPercolation.java:544)

    *  filename = input4.txt
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.check(TestPercolation.java:75)
       TestPercolation.file(TestPercolation.java:138)
       TestPercolation.test6(TestPercolation.java:276)
       TestPercolation.main(TestPercolation.java:544)

    *  filename = input7.txt
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.check(TestPercolation.java:75)
       TestPercolation.file(TestPercolation.java:138)
       TestPercolation.test6(TestPercolation.java:277)
       TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
    *  filename = snake13.txt
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.check(TestPercolation.java:75)
       TestPercolation.file(TestPercolation.java:138)
       TestPercolation.test7(TestPercolation.java:287)
```

Submission

```
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
```

Submission

```
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)
```

Submission

```
  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
          TestPercolation.test12(TestPercolation.java:459)
          TestPercolation.main(TestPercolation.java:551)

      *   N = 7
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:45)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
          TestPercolation.test12(TestPercolation.java:460)
          TestPercolation.main(TestPercolation.java:551)

      *   N = 10
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:45)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
          TestPercolation.test12(TestPercolation.java:461)
          TestPercolation.main(TestPercolation.java:551)

      *   N = 20
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
          TestPercolation.test12(TestPercolation.java:462)
          TestPercolation.main(TestPercolation.java:551)

      *   N = 50
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
          TestPercolation.test12(TestPercolation.java:463)
          TestPercolation.main(TestPercolation.java:551)

==> FAILED
```

Submission

```
Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
    *  N = 50
       java.lang.ArrayIndexOutOfBoundsException
       Percolation.isFull(Percolation.java:49)
       TestPercolation.checkIsFull(TestPercolation.java:22)
       TestPercolation.randomCalls(TestPercolation.java:496)
       TestPercolation.test13(TestPercolation.java:522)
       TestPercolation.main(TestPercolation.java:552)

==> FAILED



Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    6093.143 (FAILED, outside of range)
        stddev():  216.861 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    26236.960 (FAILED, outside of range)
        stddev():  2778.814 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
```

Submission

```
------------------------------------------------
  *  confidenceLo() = 5899.30794059368
  *  confidenceHi() = 6135.888900788782
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 22203.12292610215
  *  confidenceHi() = 28129.78038256141
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 60.30297
  *  2mean = 58.68
  *  1mean = 236.817945
  *  2mean = 233.92
  *  1mean = 59.6381835
  *  2mean = 59.765
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 532.1858631250001; stddev = 34.210549958530024
  *  order = stddev(), mean()
  *  mean = 532.939631875; stddev = 32.724425864637176
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
```

Submission

```
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================




Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
```

Submission

```
==> 0/4 tests passed



Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> passed        8 Infinity          21                    30
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity          68   (0.5x)           30
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity         263   (0.1x)          230
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity        1029   (0.0x)         1554
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity        2054   (0.0x)         3276
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
```

Submission

```
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
```

Submission

```
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 01:05:08
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
```

Submission

```
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 08:05 Percolation.java
-rw-r--r-- 1 2.7K Oct 26 08:05 PercolationStats.java
-rw-r--r-- 1 1.8K Oct 26 08:05 studentSubmission.zip


******************************************************************
*************
*  compiling
******************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
=================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
=================================================================



% checkstyle *.java
*-----------------------------------------------------------
=================================================================


% findbugs *.class
*-----------------------------------------------------------
=================================================================
```

Submission

```
Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************


Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
```

Submission

```
*  filename = input6.txt
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.file(TestPercolation.java:138)
   TestPercolation.test2(TestPercolation.java:196)
   TestPercolation.main(TestPercolation.java:540)

*  filename = input8.txt
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.file(TestPercolation.java:138)
   TestPercolation.test2(TestPercolation.java:197)
   TestPercolation.main(TestPercolation.java:540)

*  filename = input8-no.txt
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.file(TestPercolation.java:138)
   TestPercolation.test2(TestPercolation.java:198)
   TestPercolation.main(TestPercolation.java:540)

*  filename = input10-no.txt
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.file(TestPercolation.java:138)
   TestPercolation.test2(TestPercolation.java:199)
   TestPercolation.main(TestPercolation.java:540)

*  filename = greeting57.txt
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.file(TestPercolation.java:138)
   TestPercolation.test2(TestPercolation.java:200)
```

Submission

```
          TestPercolation.main(TestPercolation.java:540)

      *   filename = heart25.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test2(TestPercolation.java:201)
          TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
      *   N = 3
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:235)
          TestPercolation.main(TestPercolation.java:541)

      *   N = 5
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:236)
          TestPercolation.main(TestPercolation.java:541)

      *   N = 10
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isFull(Percolation.java:49)
          TestPercolation.checkIsFull(TestPercolation.java:22)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.random(TestPercolation.java:214)
          TestPercolation.test3(TestPercolation.java:237)
          TestPercolation.main(TestPercolation.java:541)

      *   N = 10
```

Submission

```
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:238)
        TestPercolation.main(TestPercolation.java:541)

    *   N = 20
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:239)
        TestPercolation.main(TestPercolation.java:541)

    *   N = 20
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:240)
        TestPercolation.main(TestPercolation.java:541)

    *   N = 50
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:241)
        TestPercolation.main(TestPercolation.java:541)

    *   N = 50
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:242)
        TestPercolation.main(TestPercolation.java:541)
```

Submission

```
==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:253)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:254)
     TestPercolation.main(TestPercolation.java:542)

==> FAILED
```

Submission

```
Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:263)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input10.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:264)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input50.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:265)
     TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:276)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input7.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:277)
     TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
```

Submission

```
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED


Test 11: Call all methods in random order until just before syste
```

Submission

```
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)
```

Submission

```
  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
```

Submission

```
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
     *  N = 7
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.randomCalls(TestPercolation.java:496)
        TestPercolation.test13(TestPercolation.java:519)
        TestPercolation.main(TestPercolation.java:552)

     *  N = 10
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.randomCalls(TestPercolation.java:496)
        TestPercolation.test13(TestPercolation.java:520)
        TestPercolation.main(TestPercolation.java:552)

     *  N = 20
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isOpen(Percolation.java:45)
        TestPercolation.checkIsOpen(TestPercolation.java:43)
        TestPercolation.randomCalls(TestPercolation.java:490)
        TestPercolation.test13(TestPercolation.java:521)
        TestPercolation.main(TestPercolation.java:552)

     *  N = 50
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isOpen(Percolation.java:45)
        TestPercolation.checkIsOpen(TestPercolation.java:43)
        TestPercolation.randomCalls(TestPercolation.java:490)
        TestPercolation.test13(TestPercolation.java:522)
        TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.
```

Submission

```
Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
```

Submission

```
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 0.0; stddev = 0.0
  *  order = stddev(), mean()
  *  mean = 0.0; stddev = 0.0
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
```

Submission

```
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
==> 0/4 tests passed


Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************


Timing Percolation
```

Submission

```
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity           21                     30
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity           68   (0.5x)            30
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> FAILED      128 Infinity         263   (0.1x)         230
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity        1029   (0.0x)        1554
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity        2054   (0.0x)        3276
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
```

Submission

```
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed
```

Submission

```
Total: 5/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 01:04:21
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        PASSED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 2.0K Oct 26 08:04 Percolation.java
-rw-r--r-- 1 2.7K Oct 26 08:04 PercolationStats.java
-rw-r--r-- 1 1.8K Oct 26 08:04 studentSubmission.zip


******************************************************************
```

Submission

```
*************
*  compiling
*****************************************************************
*************


% javac Percolation.java
*-----------------------------------------------------------
=================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
=================================================================



% checkstyle *.java
*-----------------------------------------------------------
=================================================================


% findbugs *.class
*-----------------------------------------------------------
=================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

=================================================================


*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
```

Submission

```
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
  *  N = 10, (i, j) = (6, 11)
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)
```

Submission

```
  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
```

Submission

```
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:235)
   TestPercolation.main(TestPercolation.java:541)

*  N = 5
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:236)
   TestPercolation.main(TestPercolation.java:541)

*  N = 10
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:237)
   TestPercolation.main(TestPercolation.java:541)

*  N = 10
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:238)
   TestPercolation.main(TestPercolation.java:541)

*  N = 20
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isFull(Percolation.java:49)
   TestPercolation.checkIsFull(TestPercolation.java:22)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:239)
   TestPercolation.main(TestPercolation.java:541)

*  N = 20
```

Submission

```
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:240)
        TestPercolation.main(TestPercolation.java:541)

   *  N = 50
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:241)
        TestPercolation.main(TestPercolation.java:541)

   *  N = 50
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.random(TestPercolation.java:214)
        TestPercolation.test3(TestPercolation.java:242)
        TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
   *  filename = input1.txt
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
        TestPercolation.check(TestPercolation.java:75)
        TestPercolation.file(TestPercolation.java:138)
        TestPercolation.test4(TestPercolation.java:251)
        TestPercolation.main(TestPercolation.java:542)

   *  filename = input1-no.txt
        java.lang.ArrayIndexOutOfBoundsException
        Percolation.isFull(Percolation.java:49)
        TestPercolation.checkIsFull(TestPercolation.java:22)
```

Submission

```
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:252)
    TestPercolation.main(TestPercolation.java:542)

  * filename = input2.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:253)
    TestPercolation.main(TestPercolation.java:542)

  * filename = input2-no.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:254)
    TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
  * filename = input20.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:263)
    TestPercolation.main(TestPercolation.java:543)

  * filename = input10.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isFull(Percolation.java:49)
    TestPercolation.checkIsFull(TestPercolation.java:22)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:264)
    TestPercolation.main(TestPercolation.java:543)
```

Submission

```
  *  filename = input50.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:265)
     TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:276)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input7.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:277)
     TestPercolation.main(TestPercolation.java:544)

==> FAILED
```

Submission

```
Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)
```

Submission

```
  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:364)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:21)
```

Submission

```
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
37)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TestPercolation.checkIsFull(TestPercolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:496)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
```

Submission

```
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 0.0; stddev = 0.0
  *  order = stddev(), mean()
```

Submission

```
  *  mean = 0.0; stddev = 0.0
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
```

Submission

```
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
==> 0/4 tests passed


Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                               2 * connected()
                 N   seconds       union()              + find()
       constructor
```

Submission

```
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity          21                     30
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity          68   (0.5x)           30
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity         263   (0.1x)          230
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity        1029   (0.0x)         1554
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isFull(Percolation.java:49)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
```

Submission

```
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3276
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
```

Submission

```
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:45)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 00:31:23
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

Submission

```
Compilation:  PASSED
Style:        FAILED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 1.6K Oct 26 07:31 Percolation.java
-rw-r--r-- 1 2.2K Oct 26 07:31 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 07:31 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
```

Submission

```
Percolation.java:2:1: File contains tab characters (this is the f
irst instance).
PercolationStats.java:5:1: File contains tab characters (this is
the first instance).
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
```

Submission

```
  *  N = 10, (i, j) = (6, 12)
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 11)
     - IndexOutOfBoundsException NOT thrown for isFull()
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
```

Submission

```
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:235)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
```

Submission

```
     TestPercolation.test3(TestPercolation.java:236)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:237)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:238)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:239)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:240)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:241)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:242)
     TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:253)
```

Submission

```
     TestPercolation.main(TestPercolation.java:542)

   *  filename = input2-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:254)
     TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
   *  filename = input20.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:263)
     TestPercolation.main(TestPercolation.java:543)

   *  filename = input10.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:264)
     TestPercolation.main(TestPercolation.java:543)

   *  filename = input50.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:265)
     TestPercolation.main(TestPercolation.java:543)

==> FAILED
```

Submission

```
Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:276)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input7.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:277)
     TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test7(TestPercolation.java:288)
          TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test8(TestPercolation.java:296)
          TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.twoPercolations(TestPercolation.java:310)
          TestPercolation.test9(TestPercolation.java:336)
          TestPercolation.main(TestPercolation.java:548)

          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.twoPercolations(TestPercolation.java:310)
          TestPercolation.test9(TestPercolation.java:337)
          TestPercolation.main(TestPercolation.java:548)

          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
```

Submission

```
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:360)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)
```

Submission

```
  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
```

Submission

```
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 0.0; stddev = 0.0
  *  order = stddev(), mean()
  *  mean = 0.0; stddev = 0.0
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
```

Submission

```
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*------------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
==> 0/4 tests passed


Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!
```

Submission

```
================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity           21                     32
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
```

Submission

```
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity           68   (0.5x)           32
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity          263   (0.1x)          232
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity         1029   (0.0x)         1556
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3278
(0.0x)          1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.
```

Submission

```
Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission	
Submission time	Sat-26-Oct 00:09:18
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
Style:        FAILED
Findbugs:     No potential bugs found.
API:          PASSED

Correctness:  0/20 tests passed
Memory:       4/8 tests passed
Timing:       5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

Submission

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 1.6K Oct 26 07:09 Percolation.java
-rw-r--r-- 1 2.3K Oct 26 07:09 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 07:09 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
Percolation.java:2:1: File contains tab characters (this is the f
irst instance).
Percolation.java:3:38: Name 'perco_uf' must match pattern '^[a-z]
[a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:4:29: Name 'open_status' must match pattern '^[a
-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:12:21: Name 'bottom_line' must match pattern '^[
a-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:13:20: 'for' is not followed by whitespace.
Percolation.java:21:19: 'if' is not followed by whitespace.
Percolation.java:25:19: 'if' is not followed by whitespace.
Percolation.java:25:38: Expression can be simplified.
Percolation.java:29:19: 'if' is not followed by whitespace.
Percolation.java:32:19: 'if' is not followed by whitespace.
Percolation.java:35:19: 'if' is not followed by whitespace.
```

Submission

```
Percolation.java:38:19: 'if' is not followed by whitespace.
Percolation.java:43:19: 'if' is not followed by whitespace.
Percolation.java:52:19: 'if' is not followed by whitespace.
PercolationStats.java:6:1: File contains tab characters (this is
the first instance).
PercolationStats.java:13:20: 'for' is not followed by whitespace.
PercolationStats.java:20:20: 'for' is not followed by whitespace.
PercolationStats.java:27:20: 'for' is not followed by whitespace.
PercolationStats.java:41:21: Name 'count_percolate' must match pa
ttern '^[a-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
PercolationStats.java:42:19: 'if' is not followed by whitespace.
PercolationStats.java:42:37: '{' is not preceded with whitespace.
PercolationStats.java:56:14: 'while' is not followed by whitespac
e.
PercolationStats.java:73: Use x*x instead of Math.pow(x, 2)
PercolationStats.java:74: Use x*x instead of Math.pow(x, 2)
PercolationStats.java:75: Use x*x instead of Math.pow(x, 2)
================================================================



% findbugs *.class
*-----------------------------------------------------------
================================================================



Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:

================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*-----------------------------------------------------------
Running 13 total tests.
```

Submission

```
Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 11)
     - IndexOutOfBoundsException NOT thrown for isFull()
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:196)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input8.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:197)
     TestPercolation.main(TestPercolation.java:540)
```

Submission

```
  *  filename = input8-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:198)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = input10-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:199)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = greeting57.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:200)
     TestPercolation.main(TestPercolation.java:540)

  *  filename = heart25.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
   Percolation.isOpen(Percolation.java:46)
   TestPercolation.checkIsOpen(TestPercolation.java:43)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:235)
   TestPercolation.main(TestPercolation.java:541)

*  N = 5
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isOpen(Percolation.java:46)
   TestPercolation.checkIsOpen(TestPercolation.java:43)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:236)
   TestPercolation.main(TestPercolation.java:541)

*  N = 10
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isOpen(Percolation.java:46)
   TestPercolation.checkIsOpen(TestPercolation.java:43)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:237)
   TestPercolation.main(TestPercolation.java:541)

*  N = 10
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isOpen(Percolation.java:46)
   TestPercolation.checkIsOpen(TestPercolation.java:43)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:238)
   TestPercolation.main(TestPercolation.java:541)

*  N = 20
   java.lang.ArrayIndexOutOfBoundsException
   Percolation.isOpen(Percolation.java:46)
   TestPercolation.checkIsOpen(TestPercolation.java:43)
   TestPercolation.check(TestPercolation.java:75)
   TestPercolation.random(TestPercolation.java:214)
   TestPercolation.test3(TestPercolation.java:239)
   TestPercolation.main(TestPercolation.java:541)
```

Submission

```
  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:240)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:241)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:242)
     TestPercolation.main(TestPercolation.java:541)

==> FAILED

Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
```

Submission

```
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:252)
    TestPercolation.main(TestPercolation.java:542)

  * filename = input2.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:253)
    TestPercolation.main(TestPercolation.java:542)

  * filename = input2-no.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test4(TestPercolation.java:254)
    TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
  * filename = input20.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:263)
    TestPercolation.main(TestPercolation.java:543)

  * filename = input10.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test5(TestPercolation.java:264)
```

Submission

```
          TestPercolation.main(TestPercolation.java:543)

     *   filename = input50.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test5(TestPercolation.java:265)
          TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
     *   filename = input3.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:275)
          TestPercolation.main(TestPercolation.java:544)

     *   filename = input4.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:276)
          TestPercolation.main(TestPercolation.java:544)

     *   filename = input7.txt
          java.lang.ArrayIndexOutOfBoundsException
          Percolation.isOpen(Percolation.java:46)
          TestPercolation.checkIsOpen(TestPercolation.java:43)
          TestPercolation.check(TestPercolation.java:75)
          TestPercolation.file(TestPercolation.java:138)
          TestPercolation.test6(TestPercolation.java:277)
          TestPercolation.main(TestPercolation.java:544)

==> FAILED
```

Submission

```
Test 7: Predetermined sites with very long percolating path
  *  filename = snake13.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:287)
     TestPercolation.main(TestPercolation.java:545)

  *  filename = snake101.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test7(TestPercolation.java:288)
     TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
  *  filename = input5.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test8(TestPercolation.java:296)
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED


Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
```

Submission

```
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:399)
```

Submission

```
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:462)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 50
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)
```

Submission

```
        Overall result: FAILED



Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 0.0; stddev = 0.0
  *  order = stddev(), mean()
  *  mean = 0.0; stddev = 0.0
==> FAILED
```

Submission

```
Total: 0/7 tests passed!

================================================================

****************************************************************
*************
*  memory usage
****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================



Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)
```

Submission

```
                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
==> 0/4 tests passed


Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================




*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*------------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                                 2 * connected()
                 N   seconds       union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity          21                     32
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity          68   (0.5x)            32
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity         263   (0.1x)           232
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      512 Infinity        1029   (0.0x)          1556
(0.0x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> FAILED     1024 Infinity        2054   (0.0x)         3278   
(0.0x)         1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed       32        0                0                0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission

Submission time	Sat-26-Oct 00:08:20
Raw Score	18.89 / 100.00
Feedback	See the Assessment Guide for information on how to read this report.

Assessment Summary

```
Compilation:  PASSED
```

Submission

```
Style:         FAILED
Findbugs:      No potential bugs found.
API:           PASSED

Correctness:   0/20 tests passed
Memory:        4/8 tests passed
Timing:        5/9 tests passed

Raw score: 18.89% [Correctness: 65%, Memory: 10%, Timing: 25%, St
yle: 0%]
```

Assessment Details

```
The following files were submitted:
----------------------------------
total 12K
-rw-r--r-- 1 1.6K Oct 26 07:08 Percolation.java
-rw-r--r-- 1 2.3K Oct 26 07:08 PercolationStats.java
-rw-r--r-- 1 1.7K Oct 26 07:08 studentSubmission.zip


*****************************************************************
*************
*  compiling
*****************************************************************
*************



% javac Percolation.java
*-----------------------------------------------------------
================================================================

% javac PercolationStats.java
*-----------------------------------------------------------
================================================================




% checkstyle *.java
*-----------------------------------------------------------
Percolation.java:2:1: File contains tab characters (this is the f
irst instance).
```

Submission

```
Percolation.java:3:38: Name 'perco_uf' must match pattern '^[a-z]
[a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:4:29: Name 'open_status' must match pattern '^[a
-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:12:21: Name 'bottom_line' must match pattern '^[
a-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
Percolation.java:13:20: 'for' is not followed by whitespace.
Percolation.java:21:19: 'if' is not followed by whitespace.
Percolation.java:25:19: 'if' is not followed by whitespace.
Percolation.java:25:38: Expression can be simplified.
Percolation.java:29:19: 'if' is not followed by whitespace.
Percolation.java:32:19: 'if' is not followed by whitespace.
Percolation.java:35:19: 'if' is not followed by whitespace.
Percolation.java:38:19: 'if' is not followed by whitespace.
Percolation.java:43:19: 'if' is not followed by whitespace.
Percolation.java:52:19: 'if' is not followed by whitespace.
PercolationStats.java:6:1: File contains tab characters (this is
the first instance).
PercolationStats.java:13:20: 'for' is not followed by whitespace.
PercolationStats.java:20:20: 'for' is not followed by whitespace.
PercolationStats.java:27:20: 'for' is not followed by whitespace.
PercolationStats.java:41:21: Name 'count_percolate' must match pa
ttern '^[a-z][a-zA-Z0-9]*$|^[A-Z][A-Z_0-9]*$'.
PercolationStats.java:42:19: 'if' is not followed by whitespace.
PercolationStats.java:42:37: '{' is not preceded with whitespace.
PercolationStats.java:56:14: 'while' is not followed by whitespac
e.
PercolationStats.java:73: Use x*x instead of Math.pow(x, 2)
PercolationStats.java:74: Use x*x instead of Math.pow(x, 2)
PercolationStats.java:75: Use x*x instead of Math.pow(x, 2)
================================================================


% findbugs *.class
*-----------------------------------------------------------
================================================================


Testing the APIs of your programs.
*-----------------------------------------------------------
Percolation:

PercolationStats:
```

Submission

```
=================================================================



*****************************************************************
*************
*  executing
*****************************************************************
*************

Testing methods in Percolation
*------------------------------------------------------------
Running 13 total tests.

Test 1: Check whether exception is called if (i, j) are out of bo
unds
  *  N = 10, (i, j) = (0, 6)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (12, 6)
  *  N = 10, (i, j) = (11, 6)
  *  N = 10, (i, j) = (6, 0)
     - IndexOutOfBoundsException NOT thrown for open()
     - IndexOutOfBoundsException NOT thrown for isOpen()
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 12)
     - IndexOutOfBoundsException NOT thrown for isFull()
  *  N = 10, (i, j) = (6, 11)
     - IndexOutOfBoundsException NOT thrown for isFull()
==> FAILED

Tests 2 through 8 create a Percolation object using your code, th
en repeatedly
open sites using open(i, j). After each call to open, we check th
at isFull(),
isOpen(), and percolates() return the corrrect results.

Test 2: Open predetermined list of sites using files
  *  filename = input6.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
```

Submission

```
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test2(TestPercolation.java:196)
    TestPercolation.main(TestPercolation.java:540)

 *  filename = input8.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test2(TestPercolation.java:197)
    TestPercolation.main(TestPercolation.java:540)

 *  filename = input8-no.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test2(TestPercolation.java:198)
    TestPercolation.main(TestPercolation.java:540)

 *  filename = input10-no.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test2(TestPercolation.java:199)
    TestPercolation.main(TestPercolation.java:540)

 *  filename = greeting57.txt
    java.lang.ArrayIndexOutOfBoundsException
    Percolation.isOpen(Percolation.java:46)
    TestPercolation.checkIsOpen(TestPercolation.java:43)
    TestPercolation.check(TestPercolation.java:75)
    TestPercolation.file(TestPercolation.java:138)
    TestPercolation.test2(TestPercolation.java:200)
    TestPercolation.main(TestPercolation.java:540)

 *  filename = heart25.txt
    java.lang.ArrayIndexOutOfBoundsException
```

Submission

```
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test2(TestPercolation.java:201)
     TestPercolation.main(TestPercolation.java:540)

==> FAILED

Test 3: Open random sites until system percolates (then test is t
erminated)
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:235)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:236)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.random(TestPercolation.java:214)
     TestPercolation.test3(TestPercolation.java:237)
     TestPercolation.main(TestPercolation.java:541)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
```

Submission

```
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:238)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 20
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:239)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 20
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:240)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 50
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:241)
      TestPercolation.main(TestPercolation.java:541)

   *  N = 50
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.random(TestPercolation.java:214)
      TestPercolation.test3(TestPercolation.java:242)
      TestPercolation.main(TestPercolation.java:541)

==> FAILED


Test 4: Opens predetermined sites, but where N = 1 and N = 2 (cor
```

Submission

```
ner case test)
  *  filename = input1.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:251)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input1-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:252)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:253)
     TestPercolation.main(TestPercolation.java:542)

  *  filename = input2-no.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test4(TestPercolation.java:254)
     TestPercolation.main(TestPercolation.java:542)

==> FAILED

Test 5: Check for backwash with predetermined sites
  *  filename = input20.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
```

Submission

```
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:263)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input10.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:264)
     TestPercolation.main(TestPercolation.java:543)

  *  filename = input50.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test5(TestPercolation.java:265)
     TestPercolation.main(TestPercolation.java:543)

==> FAILED

Test 6: Check for backwash with predetermined sites that havemult
iple percolating paths
  *  filename = input3.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
     TestPercolation.test6(TestPercolation.java:275)
     TestPercolation.main(TestPercolation.java:544)

  *  filename = input4.txt
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.file(TestPercolation.java:138)
```

Submission

```
      TestPercolation.test6(TestPercolation.java:276)
      TestPercolation.main(TestPercolation.java:544)

   *  filename = input7.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test6(TestPercolation.java:277)
      TestPercolation.main(TestPercolation.java:544)

==> FAILED

Test 7: Predetermined sites with very long percolating path
   *  filename = snake13.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test7(TestPercolation.java:287)
      TestPercolation.main(TestPercolation.java:545)

   *  filename = snake101.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test7(TestPercolation.java:288)
      TestPercolation.main(TestPercolation.java:545)

==> FAILED

Test 8: Opens every site
   *  filename = input5.txt
      java.lang.ArrayIndexOutOfBoundsException
      Percolation.isOpen(Percolation.java:46)
      TestPercolation.checkIsOpen(TestPercolation.java:43)
      TestPercolation.check(TestPercolation.java:75)
      TestPercolation.file(TestPercolation.java:138)
      TestPercolation.test8(TestPercolation.java:296)
```

Submission

```
     TestPercolation.main(TestPercolation.java:546)

==> FAILED

Test 9: Create multiple Percolation objects at the same time
        (to make sure you didn't store data in static variables)
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:336)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:337)
     TestPercolation.main(TestPercolation.java:548)

     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.check(TestPercolation.java:75)
     TestPercolation.twoPercolations(TestPercolation.java:310)
     TestPercolation.test9(TestPercolation.java:338)
     TestPercolation.main(TestPercolation.java:548)

==> FAILED

Test 10: Open predetermined list of sites using file
         but change the order in which methods are called
  *  filename = input8.txt;  order =     isFull(),     isOpen(),
percolates()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isFull(), percolates(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
```

Submission

```
()]
     - student   = false
     - reference = true
  *  filename = input8.txt;  order =     isOpen(),     isFull(),
percolates()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:178)
     TestPercolation.test10(TestPercolation.java:385)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order =     isOpen(), percolates(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:179)
     TestPercolation.test10(TestPercolation.java:386)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isOpen(),
    isFull()
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.file(TestPercolation.java:180)
     TestPercolation.test10(TestPercolation.java:387)
     TestPercolation.main(TestPercolation.java:549)

  *  filename = input8.txt;  order = percolates(),     isFull(),
    isOpen()
     isFull(1, 3) returns wrong value [after 1 total call to open
()]
     - student   = false
     - reference = true
==> FAILED

Test 11: Call all methods in random order until just before syste
m percolates
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
```

Submission

```
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:398)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:399)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:400)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:401)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:402)
     TestPercolation.main(TestPercolation.java:550)

  *  N = 50
```

Submission

```
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsUntilPercolation(TestPercolation.
java:363)
     TestPercolation.test11(TestPercolation.java:403)
     TestPercolation.main(TestPercolation.java:550)

==> FAILED

Test 12: Call all methods in random order with inputs not prone t
o backwash
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:458)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:459)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
     TestPercolation.test12(TestPercolation.java:460)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
25)
```

Submission

```
     TestPercolation.test12(TestPercolation.java:461)
     TestPercolation.main(TestPercolation.java:551)

  *  N = 20
     isFull(1, 2) returns wrong value [after 3 total calls to ope
n()]
     - student   = false
     - reference = true
  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCallsNoBackwash(TestPercolation.java:4
31)
     TestPercolation.test12(TestPercolation.java:463)
     TestPercolation.main(TestPercolation.java:551)

==> FAILED

Test 13: Call all methods in random order until all sites are ope
n
  *  N = 3
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:517)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 5
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:518)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 7
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:519)
     TestPercolation.main(TestPercolation.java:552)
```

Submission

```
  *  N = 10
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.open(Percolation.java:22)
     TestPercolation.randomCalls(TestPercolation.java:484)
     TestPercolation.test13(TestPercolation.java:520)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 20
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:521)
     TestPercolation.main(TestPercolation.java:552)

  *  N = 50
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TestPercolation.checkIsOpen(TestPercolation.java:43)
     TestPercolation.randomCalls(TestPercolation.java:490)
     TestPercolation.test13(TestPercolation.java:522)
     TestPercolation.main(TestPercolation.java:552)

==> FAILED


Total: 0/13 tests passed!
================================================================


Testing methods in PercolationStats
*------------------------------------------------------------
Running 7 total tests.

Test 1a-1b: Test mean and standard deviation of percolation thres
hold

Creating new PercolationStats(100, 50)
------------------------------------------------


PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)
```

Submission

```
        Overall result: FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------

PercolationStats reports:
        mean():    0.000 (FAILED, outside of range)
        stddev():  0.000 (FAILED, outside of range)

        Overall result: FAILED


Test 1c-d: Test confidence interval of PercolationStats

Creating new PercolationStats(100, 50)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Creating new PercolationStats(200, 10)
------------------------------------------------
  *  confidenceLo() = 0.0
  *  confidenceHi() = 0.0
==> FAILED

Test 2: Check whether exception is called if N, T are out of boun
ds
  *  N = -23, T = 42
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N =  23, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
  *  N = -42, T =  0
     - IllegalArgumentException NOT thrown for PercolationStats()
==> FAILED

Test 3: Create multiple PercolationStats objects at the same time
 (to make sure you didn't store data in static variables)
  *  1mean = 0.0
  *  2mean = 0.0
  *  1mean = 0.0
  *  2mean = 0.0
```

Submission

```
  *  1mean = 0.0
  *  2mean = 0.0
==> FAILED

Test 4: Call the methods of PercolationStats in either order.
  *  order = mean(), stddev()
  *  mean = 0.0; stddev = 0.0
  *  order = stddev(), mean()
  *  mean = 0.0; stddev = 0.0
==> FAILED

Total: 0/7 tests passed!

================================================================

*****************************************************************
*************
*  memory usage
*****************************************************************
*************

Computing memory of Percolation
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of grid si
ze (max allowed: 17 N^2 + 128 N + 1024 bytes)

                 N        bytes
--------------------------------------------
=> passed       64        41864
=> passed      256       609032
=> passed      512      2397448
=> passed     1024      9513224
==> 4/4 tests passed


Estimated student memory = 9.00 N^2 + 74.00 N + 264.00  (R^2 = 1.
000)

Total: 4/4 tests passed!

================================================================
```

Submission

```
Computing memory of PercolationStats
*-----------------------------------------------------------
Running 4 total tests.

Test 1a-1d: Measuring total memory usage as a function of T (max
allowed: 8 T + 128 bytes)

                 T        bytes
--------------------------------------------
=> FAILED       16      1556384 (6e+03x)
=> FAILED       32      3112672 (8e+03x)
=> FAILED       64      6225248 (1e+04x)
=> FAILED      128     12450400 (1e+04x)
==> 0/4 tests passed


Estimated student memory = 97268.00 T + 96.00  (R^2 = 1.000)

Total: 0/4 tests passed!

================================================================



*****************************************************************
*************
*  timing
*****************************************************************
*************

Timing Percolation
*-----------------------------------------------------------
Running 9 total tests.

Tests 1a-1e: Measuring runtime and counting calls to connected(),
 union() and
             find() in WeightedQuickUnionUF.


For each N, a percolation object is generated and sites are rando
```

Submission

```
mly opened
until the system percolates. If you do not pass the correctness t
ests, these
results may be meaningless.

                                               2 * connected()
                 N   seconds      union()              + find()
       constructor
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> passed        8 Infinity          21                    32
                1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED       32 Infinity          68   (0.5x)           32
(0.2x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED      128 Infinity         263   (0.1x)          232
(0.1x)          1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> FAILED      512 Infinity         1029   (0.0x)         1556
(0.0x)         1
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.run(TimePercolation.java:16)
     TimePercolation.operationCountTest(TimePercolation.java:326)
     TimePercolation.testLite(TimePercolation.java:404)
     TimePercolation.main(TimePercolation.java:420)

=> FAILED     1024 Infinity         2054   (0.0x)         3278
(0.0x)         1
==> 1/5 tests passed

Running time in seconds depends on the machine on which the scrip
t runs,
and  varies each time that you submit. If one of the values in th
e table
violates the performance limits, the factor by which you failed t
he test
appears in parentheses. For example, (9.6x) in the union() column
indicates that it uses 9.6x too many calls.



Tests 2a-2d: This test checks whether you use a constant number o
f calls to
union(), connected(), and find() per call to open(), isFull(), an
d percolates().
The table below shows max(union(), connected(), find()) calls mad
e during a
single call to open(), isFull(), and percolates().

                 N     per open()      per isOpen()    per isFull
()    per percolates()
-----------------------------------------------------------------
----------------------------
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)
```

Submission

```
=> passed       32        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      128        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed      512        0               0               0
         0
     java.lang.ArrayIndexOutOfBoundsException
     Percolation.isOpen(Percolation.java:46)
     TimePercolation.countMaxOperations(TimePercolation.java:50)
     TimePercolation.maxOperationCountTest(TimePercolation.java:3
86)
     TimePercolation.testLite(TimePercolation.java:405)
     TimePercolation.main(TimePercolation.java:420)

=> passed     1024        0               0               0
         0
==> 4/4 tests passed

Total: 5/9 tests passed!
================================================================
```

Submission

Submission	
Submission time	Fri-25-Oct 23:59:04
Raw Score	0.00 / 100.00
Feedback	see below

```
Compilation:  PASSED

API:          FAILED

PercolationStats:


The following fields should be made private:
  *  public Percolation[] per
  *  public int[] counter
```

Submission	
Submission time	Fri-25-Oct 23:59:01
Raw Score	0.00 / 100.00
Feedback	see below

```
Compilation:  PASSED

API:          FAILED

PercolationStats:


The following fields should be made private:
  *  public Percolation[] per
  *  public int[] counter
```

Submission	
Submission time	Fri-25-Oct 20:29:27
Raw Score	0.00 / 100.00

Submission	
Feedback	Error extracting files! Zip file invalid.