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

ASSESSMENT SUMMARY

Compilation: PASSED

API: PASSED

Findbugs: FAILED (1 warning)

Checkstyle: FAILED (7 warnings)

Correctness: 28/42 tests passed

Memory: 0/11 tests passed

Timing: 0/17 tests passed

Aggregate score: 50.00%

[Compilation: 5%, API: 5%, Findbugs: 0%, Checkstyle: 0%, Correctness: 60%, Memory: 10%, Timing: 20%]

ASSESSMENT DETAILS

The following files were submitted:

----------------------------------

7.5K Jun 17 22:45 Board.java

6.0K Jun 17 22:45 Solver.java

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* COMPILING

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% javac Board.java

\*-----------------------------------------------------------

% javac Solver.java

\*-----------------------------------------------------------

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

Checking the APIs of your programs.

\*-----------------------------------------------------------

Board:

Solver:

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* CHECKING STYLE AND COMMON BUG PATTERNS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% findbugs \*.class

\*-----------------------------------------------------------

H D DLS\_DEAD\_LOCAL\_STORE DLS: Assigns a value to the local variable 'initial' but that value is never used. At Board.java:[line 262]

Warnings generated: 1

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

% checkstyle \*.java

\*-----------------------------------------------------------

Board.java:1:3: '//' or '/\*' is not followed by whitespace. [IllegalTokenText]

Board.java:3: Do not use .\* in import statements. [AvoidStarImport]

Solver.java:1:3: '//' or '/\*' is not followed by whitespace. [IllegalTokenText]

Solver.java:29:9: Declare instance variables after static variables but before constructors and methods. [DeclarationOrder]

Solver.java:37:9: Declare instance variables after static variables but before constructors and methods. [DeclarationOrder]

Solver.java:44:9: Declare instance variables after static variables but before constructors and methods. [DeclarationOrder]

Solver.java:50:9: Define constructors after static and instance variables but before methods. [DeclarationOrder]

Checkstyle ends with 7 errors.

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* TESTING CORRECTNESS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Testing correctness of Board

\*-----------------------------------------------------------

Running 22 total tests.

Tests 5, 6, 13, and 14 rely upon toString() returning results in prescribed format.

Test 1a: Test hamming() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle07.txt

\* puzzle17.txt

\* puzzle27.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 1b: Test hamming() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

\* 127-by-127

==> passed

Test 2a: Test manhattan() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle07.txt

\* puzzle17.txt

\* puzzle27.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 2b: Test manhattan() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

\* 127-by-127

==> passed

Test 3: Test dimension() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

==> passed

Test 4a: Test toString() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle06.txt

\* puzzle09.txt

\* puzzle23.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 4b: Test toString() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

\* 127-by-127

==> passed

Test 5a: Test neighbors() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle06.txt

\* puzzle09.txt

\* puzzle23.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 5b: Test neighbors() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

\* 127-by-127

==> passed

Test 6a: Test neighbors() of neigbors() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle06.txt

\* puzzle09.txt

\* puzzle23.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 6b: Test neighbors() of neighbors() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

==> passed

Test 7a: Test twin() with file inputs

\* puzzle04.txt

\* puzzle00.txt

\* puzzle06.txt

\* puzzle09.txt

\* puzzle23.txt

\* puzzle2x2-unsolvable1.txt

==> passed

Test 7b: Test twin() with random n-by-n boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 9-by-9

\* 10-by-10

==> passed

Test 8a: Test isGoal() on file inputs

\* puzzle00.txt

\* puzzle04.txt

\* puzzle16.txt

\* puzzle06.txt

\* puzzle09.txt

\* puzzle23.txt

\* puzzle2x2-unsolvable1.txt

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-00.txt

\* puzzle4x4-00.txt

==> passed

Test 8b: Test isGoal() on n-by-n goal boards

\* 2-by-2

\* 3-by-3

\* 4-by-4

\* 5-by-5

\* 6-by-6

\* 100-by-100

==> passed

Test 9: Check whether two Board objects can be created at the same time

\* random 3-by-3 and 3-by-3 boards

\* random 4-by-4 and 4-by-4 boards

\* random 2-by-2 and 2-by-2 boards

\* random 3-by-3 and 4-by-4 boards

\* random 4-by-4 and 3-by-3 boards

==> passed

Test 10a: Check equals()

\* reflexive

\* symmetric

\* checks that individual entries of array are equal

\* argument is object of type String

\* argument is object of type Object

\* argument is null

\* argument is Board of different dimension

==> passed

Test 10b: Test equals() on m-by-m vs. n-by-n goal boards

\* m = 2, n = 2

\* m = 3, n = 3

\* m = 4, n = 4

\* m = 2, n = 5

\* m = 5, n = 2

==> passed

Test 11: Check that Board is immutable by changing argument array after

construction and making sure Board does not mutate

==> passed

Test 12: Check that Board is immutable by testing whether methods

return the same value, regardless of order in which called

\* puzzle10.txt

\* puzzle20.txt

\* puzzle30.txt

\* 2-by-2

\* 3-by-3

\* 4-by-4

==> passed

Test 13: Call hamming() on a board that is kth-neighbor of a board

\* 0th neighbor of puzzle27.txt

\* 1th neighbor of puzzle27.txt

\* 2th neighbor of puzzle27.txt

\* 13th neighbor of puzzle27.txt

\* 13th neighbor of puzzle00.txt

\* 13th neighbor of puzzle2x2-unsolvable1.txt

==> passed

Test 14: Call manhattan() on a board that is a kth-neighbor of a board

\* 0th neighbor of puzzle27.txt

\* 1th neighbor of puzzle27.txt

\* 2th neighbor of puzzle27.txt

\* 13th neighbor of puzzle27.txt

\* 13th neighbor of puzzle00.txt

\* 13th neighbor of puzzle2x2-unsolvable1.txt

==> passed

Total: 22/22 tests passed!

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* TESTING CORRECTNESS (substituting reference Board)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Testing correctness of Solver

\*-----------------------------------------------------------

Running 20 total tests.

Test 1: Call moves() with file inputs

\* puzzle00.txt

\* puzzle01.txt

\* puzzle02.txt

\* puzzle03.txt

\* puzzle04.txt

\* puzzle05.txt

\* puzzle06.txt

\* puzzle07.txt

\* puzzle08.txt

\* puzzle09.txt

\* puzzle10.txt

\* puzzle11.txt

\* puzzle12.txt

\* puzzle13.txt

==> passed

Test 2: Call solution() with file inputs

\* puzzle00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testSolution(TestSolver.java:221)

TestSolver.testSolution(TestSolver.java:212)

TestSolver.test2(TestSolver.java:303)

TestSolver.main(TestSolver.java:973)

\* puzzle01.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 0

- moves() = 1

\* puzzle02.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 1

- moves() = 2

\* puzzle03.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 2

- moves() = 3

\* puzzle04.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 3

- moves() = 4

\* puzzle05.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 4

- moves() = 5

\* puzzle06.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 5

- moves() = 6

\* puzzle07.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 6

- moves() = 7

\* puzzle08.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 7

- moves() = 8

\* puzzle10.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 9

- moves() = 10

\* puzzle15.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 14

- moves() = 15

==> FAILED

Test 3: Create two Solver objects at the same time

\* puzzle04.txt and puzzle04.txt

\* puzzle00.txt and puzzle04.txt

\* puzzle04.txt and puzzle00.txt

==> passed

Test 4a: Call isSolvable() with file inputs

\* puzzle01.txt

\* puzzle03.txt

\* puzzle04.txt

\* puzzle17.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 4b: Call isSolvable() on random n-by-n boards

\* 100 random 2-by-2 boards

==> passed

Test 5: Call moves() on unsolvable puzzles

\* puzzle2x2-unsolvable1.txt

\* puzzle2x2-unsolvable2.txt

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-unsolvable2.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 6: Call solution() on unsolvable puzzles

\* puzzle2x2-unsolvable1.txt

\* puzzle2x2-unsolvable2.txt

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-unsolvable2.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 7a: Check that Solver is immutable by testing whether methods

return the same value, regardless of order in which called

\* puzzle3x3-00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testImmutableSolver(TestSolver.java:512)

TestSolver.testImmutableSolver(TestSolver.java:601)

TestSolver.test7a(TestSolver.java:609)

TestSolver.main(TestSolver.java:991)

- sequence of Solver operations was:

Solver solver = new Solver(initial);

solver.solution()

\* puzzle3x3-01.txt

\* puzzle3x3-05.txt

\* puzzle3x3-10.txt

\* random 2-by-2 solvable boards

==> FAILED

Test 7b: Check that Solver is immutable by testing whether methods

return the same value, regardless of order in which called

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-unsolvable2.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

- sequence of Solver operations was:

Solver solver = new Solver(initial);

==> FAILED

Test 8: Call moves() with more file inputs

\* puzzle14.txt

\* puzzle15.txt

\* puzzle16.txt

\* puzzle17.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 9: Check whether equals() method in Board is called

with an argument of the wrong type

\* puzzle00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testSolution(TestSolver.java:221)

TestSolver.testEqualsWrongType(TestSolver.java:664)

TestSolver.test9(TestSolver.java:679)

TestSolver.main(TestSolver.java:1000)

\* puzzle05.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 4

- moves() = 5

- detected call to equals() method in Board with

an argument that is not of type Board

- likely a bug in the critical optimization

\* puzzle10.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 9

- moves() = 10

- detected call to equals() method in Board with

an argument that is not of type Board

- likely a bug in the critical optimization

\* puzzle15.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 14

- moves() = 15

- detected call to equals() method in Board with

an argument that is not of type Board

- likely a bug in the critical optimization

==> FAILED

Test 10: Check that constructor throws exception if board is null

==> passed

Test 11: Check for fragile dependence on toString()

\* puzzle00.txt

\* puzzle04.txt

\* puzzle08.txt

\* puzzle12.txt

==> passed

Test 12a: Call moves() with 2-by-2 file inputs

\* puzzle2x2-00.txt

\* puzzle2x2-01.txt

\* puzzle2x2-02.txt

\* puzzle2x2-03.txt

\* puzzle2x2-04.txt

\* puzzle2x2-05.txt

\* puzzle2x2-06.txt

==> passed

Test 12b: Call solution() with 2-by-2 file inputs

\* puzzle2x2-00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testSolution(TestSolver.java:221)

TestSolver.testSolution(TestSolver.java:212)

TestSolver.test12b(TestSolver.java:767)

TestSolver.main(TestSolver.java:1012)

\* puzzle2x2-01.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 0

- moves() = 1

\* puzzle2x2-02.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 1

- moves() = 2

\* puzzle2x2-03.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 2

- moves() = 3

\* puzzle2x2-04.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 3

- moves() = 4

\* puzzle2x2-05.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 4

- moves() = 5

\* puzzle2x2-06.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 5

- moves() = 6

==> FAILED

Test 13a: Call moves() with 3-by-3 file inputs

\* puzzle3x3-00.txt

\* puzzle3x3-01.txt

\* puzzle3x3-02.txt

\* puzzle3x3-03.txt

\* puzzle3x3-04.txt

\* puzzle3x3-05.txt

\* puzzle3x3-06.txt

\* puzzle3x3-07.txt

\* puzzle3x3-08.txt

\* puzzle3x3-09.txt

\* puzzle3x3-10.txt

\* puzzle3x3-11.txt

\* puzzle3x3-12.txt

\* puzzle3x3-13.txt

\* puzzle3x3-14.txt

\* puzzle3x3-15.txt

\* puzzle3x3-16.txt

\* puzzle3x3-17.txt

\* puzzle3x3-18.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 13b: Call solution() with 3-by-3 file inputs

\* puzzle3x3-00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testSolution(TestSolver.java:221)

TestSolver.testSolution(TestSolver.java:212)

TestSolver.test13b(TestSolver.java:821)

TestSolver.main(TestSolver.java:1018)

\* puzzle3x3-01.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 0

- moves() = 1

\* puzzle3x3-02.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 1

- moves() = 2

\* puzzle3x3-03.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 2

- moves() = 3

\* puzzle3x3-04.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 3

- moves() = 4

\* puzzle3x3-05.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 4

- moves() = 5

\* puzzle3x3-06.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 5

- moves() = 6

\* puzzle3x3-07.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 6

- moves() = 7

\* puzzle3x3-08.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 7

- moves() = 8

\* puzzle3x3-09.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 8

- moves() = 9

\* puzzle3x3-10.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 9

- moves() = 10

\* puzzle3x3-11.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 10

- moves() = 11

\* puzzle3x3-12.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 11

- moves() = 12

\* puzzle3x3-13.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 12

- moves() = 13

\* puzzle3x3-14.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 13

- moves() = 14

\* puzzle3x3-15.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 14

- moves() = 15

\* puzzle3x3-16.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 15

- moves() = 16

\* puzzle3x3-17.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 16

- moves() = 17

\* puzzle3x3-18.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 14a: Call moves() with 4-by-4 file inputs

\* puzzle4x4-00.txt

\* puzzle4x4-01.txt

\* puzzle4x4-02.txt

\* puzzle4x4-03.txt

\* puzzle4x4-04.txt

\* puzzle4x4-05.txt

\* puzzle4x4-06.txt

\* puzzle4x4-07.txt

\* puzzle4x4-08.txt

\* puzzle4x4-09.txt

\* puzzle4x4-10.txt

\* puzzle4x4-11.txt

\* puzzle4x4-12.txt

\* puzzle4x4-13.txt

\* puzzle4x4-14.txt

\* puzzle4x4-15.txt

\* puzzle4x4-16.txt

\* puzzle4x4-17.txt

\* puzzle4x4-18.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 14b: Call solution() with 4-by-4 file inputs

\* puzzle4x4-00.txt

java.lang.NullPointerException

Solver$SearchNode.access$000(Solver.java:19)

Solver.solution(Solver.java:175)

TestSolver.testSolution(TestSolver.java:221)

TestSolver.testSolution(TestSolver.java:212)

TestSolver.test14b(TestSolver.java:907)

TestSolver.main(TestSolver.java:1024)

\* puzzle4x4-01.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 0

- moves() = 1

\* puzzle4x4-02.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 1

- moves() = 2

\* puzzle4x4-03.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 2

- moves() = 3

\* puzzle4x4-04.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 3

- moves() = 4

\* puzzle4x4-05.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 4

- moves() = 5

\* puzzle4x4-06.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 5

- moves() = 6

\* puzzle4x4-07.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 6

- moves() = 7

\* puzzle4x4-08.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 7

- moves() = 8

\* puzzle4x4-09.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 8

- moves() = 9

\* puzzle4x4-10.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 9

- moves() = 10

\* puzzle4x4-11.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 10

- moves() = 11

\* puzzle4x4-12.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 11

- moves() = 12

\* puzzle4x4-13.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 12

- moves() = 13

\* puzzle4x4-14.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 13

- moves() = 14

\* puzzle4x4-15.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 14

- moves() = 15

\* puzzle4x4-16.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 15

- moves() = 16

\* puzzle4x4-17.txt

- number of boards in solution() does not equal to 1 + moves()

(it should be 1 greater because solution() starts with the inital board)

- length of solution() = 16

- moves() = 17

\* puzzle4x4-18.txt

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

==> FAILED

Test 15: Call moves() with random solvable n-by-n boards

\* 100 random 2-by-2 boards

\* 200 random 3-by-3 boards that are <= 20 moves from goal

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

OperationCountLimitExceededException

Number of calls to insert(), delMin(), and min() in MinPQ exceeds limit: 10000000

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

- failed on trial 33 of 200

==> FAILED

Total: 6/20 tests passed!

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* MEMORY

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Computing memory of Board

\*-----------------------------------------------------------

Running 8 total tests.

Memory usage of an n-by-n board

n student (bytes) reference (bytes)

----------------------------------------------------------

=> FAILED 4 472 240

=> FAILED 8 1112 560

=> FAILED 12 2008 1008

=> FAILED 16 3160 1584

=> FAILED 20 4568 2288

=> FAILED 36 12760 6384

=> FAILED 72 46168 23088

=> FAILED 120 122968 61488

==> 0/8 tests passed

Total: 0/8 tests passed!

Student memory = 8.00 n^2 + 64.00 n + 88.00 (R^2 = 1.000)

Reference memory = 4.00 n^2 + 32.00 n + 48.00 (R^2 = 1.000)

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

Computing memory of Solver

\*------------------------------------------------

...

WARNING: the grading output was truncated due to excessive length.

Typically, this is because you have a method that has an unanticipated side effect

(such as printing to standard output or throwing an exception). A large amount of output

can also arise from failing many tests.