[陈](https://www.coursera.org/" \o "主页)



返回到第 4 周**课程**

[上一个](https://www.coursera.org/learn/algorithms-part1/quiz/rOobS/interview-questions-optional)

[下一个](https://www.coursera.org/learn/algorithms-part1/supplement/2kwpU/lecture-slides)

### Priority Queues

##### [Overview](https://www.coursera.org/learn/algorithms-part1/supplement/vVzFp/overview)

[10 分](https://www.coursera.org/learn/algorithms-part1/supplement/vVzFp/overview)

##### [Lecture Slides](https://www.coursera.org/learn/algorithms-part1/supplement/eHe3d/lecture-slides)

##### [APIs and Elementary Implementations](https://www.coursera.org/learn/algorithms-part1/lecture/A3kA3/apis-and-elementary-implementations)

[12 分](https://www.coursera.org/learn/algorithms-part1/lecture/A3kA3/apis-and-elementary-implementations)

##### [Binary Heaps](https://www.coursera.org/learn/algorithms-part1/lecture/Uzwy6/binary-heaps)

[23 分](https://www.coursera.org/learn/algorithms-part1/lecture/Uzwy6/binary-heaps)

##### [Heapsort](https://www.coursera.org/learn/algorithms-part1/lecture/ZjoSM/heapsort)

[14 分](https://www.coursera.org/learn/algorithms-part1/lecture/ZjoSM/heapsort)

##### [Event-Driven Simulation (optional)](https://www.coursera.org/learn/algorithms-part1/lecture/QVhGs/event-driven-simulation-optional)

[22 分](https://www.coursera.org/learn/algorithms-part1/lecture/QVhGs/event-driven-simulation-optional)

##### ****[练习测验:](https://www.coursera.org/learn/algorithms-part1/quiz/rOobS/interview-questions-optional)****

##### [Interview Questions (optional)](https://www.coursera.org/learn/algorithms-part1/quiz/rOobS/interview-questions-optional)

[3 questions](https://www.coursera.org/learn/algorithms-part1/quiz/rOobS/interview-questions-optional)

##### ****[编程作业:](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle)****

##### [8 Puzzle](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle)

[5小时](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle)

### Elementary Symbol Tables

## 编程作业: 8 Puzzle

您还未提交解答。您必须获得 80/100 分才能通过。

|  |  |
| --- | --- |
| **截止时间** | Pass this assignment by 二月 19, 11:59 晚上 PST |

1. [说明](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle)
2. [**我提交的作业**](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle/submission)
3. [讨论](https://www.coursera.org/learn/algorithms-part1/programming/iqOQi/8-puzzle/discussions)

  Create submission

### Your Submissions

**日期**

**分数**

**通过了吗？**

25 一月 2017 在 6:33 晚上

73/100

否

8 Puzzle

73/100

隐藏 评分反馈

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

ASSESSMENT SUMMARY

Compilation: FAILED (0 errors, 10 warnings)

API: PASSED

Findbugs: FAILED (5 warnings)

Checkstyle: FAILED (16 warnings)

Correctness: 32/42 tests passed

Memory: 8/11 tests passed

Timing: 12/17 tests passed

Aggregate score: 72.10%

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

ASSESSMENT DETAILS

The following files were submitted:

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

6.6K Jan 25 10:33 Board.java

4.0K Jan 25 10:33 Solver.java

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

\* COMPILING

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

% javac Board.java

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

% javac Solver.java

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

Solver.java:15: warning: [rawtypes] found raw type: MinPQ

private MinPQ pQueue = new MinPQ(new boardNodeComparator());

^

missing type arguments for generic class MinPQ<Key>

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:15: warning: [rawtypes] found raw type: MinPQ

private MinPQ pQueue = new MinPQ(new boardNodeComparator());

^

missing type arguments for generic class MinPQ<Key>

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:15: warning: [unchecked] unchecked call to MinPQ(Comparator<Key>) as a member of the raw type MinPQ

private MinPQ pQueue = new MinPQ(new boardNodeComparator());

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:16: warning: [rawtypes] found raw type: MinPQ

private MinPQ twin\_pQueue = new MinPQ(new boardNodeComparator());

^

missing type arguments for generic class MinPQ<Key>

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:16: warning: [rawtypes] found raw type: MinPQ

private MinPQ twin\_pQueue = new MinPQ(new boardNodeComparator());

^

missing type arguments for generic class MinPQ<Key>

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:16: warning: [unchecked] unchecked call to MinPQ(Comparator<Key>) as a member of the raw type MinPQ

private MinPQ twin\_pQueue = new MinPQ(new boardNodeComparator());

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:51: warning: [unchecked] unchecked call to insert(Key) as a member of the raw type MinPQ

pQueue.insert(iniNode);

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:52: warning: [unchecked] unchecked call to insert(Key) as a member of the raw type MinPQ

twin\_pQueue.insert(twin\_iniNode);

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:71: warning: [unchecked] unchecked call to insert(Key) as a member of the raw type MinPQ

pQueue.insert(bnode);

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

Solver.java:80: warning: [unchecked] unchecked call to insert(Key) as a member of the raw type MinPQ

twin\_pQueue.insert(bnode);

^

where Key is a type-variable:

Key extends Object declared in class MinPQ

10 warnings

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

Checking the APIs of your programs.

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

Board:

Solver:

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

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

\* CHECKING STYLE AND COMMON BUG PATTERNS

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

% findbugs \*.class

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

L B NM\_CLASS\_NAMING\_CONVENTION Nm: The class name 'Solver$boardNode' doesn't start with an upper-case letter. At Solver.java:[lines 23-32]

M B BC\_EQUALS\_METHOD\_SHOULD\_WORK\_FOR\_ALL\_OBJECTS BC: The 'equals()' method for 'Board' assumes the argument is of type 'Board'; it should be of type 'java.lang.Object'. At Board.java:[line 109]

L B NM\_CLASS\_NAMING\_CONVENTION Nm: The class name 'Solver$boardNodeComparator' doesn't start with an upper-case letter. At Solver.java:[lines 35-37]

M P UUF\_UNUSED\_FIELD UuF: The instance (or static) variable 'moves' is never used. Consider removing it from the class. In Solver.java

M P URF\_UNREAD\_FIELD UrF: The instance (or static) variable 'solution' is never read. Consider removing it from the class. At Solver.java:[line 19]

Warnings generated: 5

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

% checkstyle \*.java

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

Board.java:4:8: Unused import statement for 'edu.princeton.cs.algs4.StdOut'. [UnusedImports]

Board.java:5:8: Unused import statement for 'edu.princeton.cs.algs4.StdRandom'. [UnusedImports]

Board.java:74:25: Inner assignments should be avoided. [InnerAssignment]

Board.java:170:16: 'for' is not followed by whitespace. [WhitespaceAfter]

Solver.java:14:23: The instance variable 'twin\_iniNode' must start with a lowercase letter and use camelCase. [MemberName]

Solver.java:16:19: The instance variable 'twin\_pQueue' must start with a lowercase letter and use camelCase. [MemberName]

Solver.java:18:30: The instance variable 'twin\_deleted' must start with a lowercase letter and use camelCase. [MemberName]

Solver.java:23:19: The class 'boardNode' must start with an uppercase letter and use CamelCase. [TypeName]

Solver.java:35:19: The class 'boardNodeComparator' must start with an uppercase letter and use CamelCase. [TypeName]

Solver.java:48:44: Inner assignments should be avoided. [InnerAssignment]

Solver.java:49:50: Inner assignments should be avoided. [InnerAssignment]

Solver.java:50:42: Inner assignments should be avoided. [InnerAssignment]

Solver.java:60:19: The local variable 'twin\_del' must start with a lowercase letter and use camelCase. [LocalVariableName]

Solver.java:61:22: Inner assignments should be avoided. [InnerAssignment]

Solver.java:62:31: Inner assignments should be avoided. [InnerAssignment]

Solver.java:113:30: The local (or parameter) variable 'solution' has the same name as an instance variable. Use a different name. [HiddenField]

Checkstyle ends with 16 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

java.lang.ClassCastException: java.lang.String cannot be cast to Board

Board.equals(Board.java:109)

TestBoard.test10a(TestBoard.java:963)

TestBoard.main(TestBoard.java:1380)

- cannot check for equality with a String object

\* argument is object of type Object

\* argument is null

java.lang.NullPointerException

Board.equals(Board.java:109)

TestBoard.test10a(TestBoard.java:998)

TestBoard.main(TestBoard.java:1380)

- fails when argument is null

\* argument is Board of different dimension

java.lang.ArrayIndexOutOfBoundsException: 3

Board.equals(Board.java:109)

TestBoard.test10a(TestBoard.java:1021)

TestBoard.main(TestBoard.java:1380)

- cannot check for equality with a Board of different dimension

==> FAILED

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

java.lang.ArrayIndexOutOfBoundsException: 2

Board.equals(Board.java:109)

TestBoard.testEquals(TestBoard.java:1050)

TestBoard.test10b(TestBoard.java:1080)

TestBoard.main(TestBoard.java:1381)

==> FAILED

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: 20/22 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

\* 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() = 2

- moves() = 0

\* 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() = 3

- moves() = 0

\* 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() = 4

- moves() = 0

\* 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() = 5

- moves() = 0

\* 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() = 6

- moves() = 0

\* 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() = 7

- moves() = 0

\* 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() = 8

- moves() = 0

\* 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() = 9

- moves() = 0

\* 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() = 11

- moves() = 0

\* 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() = 16

- moves() = 0

==> 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

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-unsolvable2.txt

\* puzzle4x4-unsolvable.txt

==> passed

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

\* puzzle4x4-unsolvable.txt

==> passed

Test 6: Call solution() on unsolvable puzzles

\* puzzle2x2-unsolvable1.txt

\* puzzle2x2-unsolvable2.txt

\* puzzle3x3-unsolvable1.txt

\* puzzle3x3-unsolvable2.txt

\* puzzle4x4-unsolvable.txt

==> passed

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

return the same value, regardless of order in which called

\* puzzle3x3-00.txt

\* puzzle3x3-01.txt

- failed after 4 calls to random methods in Solver

- first call to student moves() = 1

- last call to student moves() = 0

- any call to reference moves() = 1

- sequence of Solver operations was:

Solver solver = new Solver(initial);

solver.isSolvable() -> true

solver.moves() -> 1

solver.solution()

solver.moves() -> 0

\* puzzle3x3-05.txt

- failed after 6 calls to random methods in Solver

- first call to student moves() = 5

- last call to student moves() = 0

- any call to reference moves() = 5

- sequence of Solver operations was:

Solver solver = new Solver(initial);

solver.moves() -> 5

solver.moves() -> 5

solver.moves() -> 5

solver.isSolvable() -> true

solver.solution()

solver.moves() -> 0

\* puzzle3x3-10.txt

- student length = 11

- reference length = 1

- failed after 8 calls to random methods in Solver

- first and last call to student solution() returned different iterables

- 'student' means value after first call and 'reference' means value after last call

- sequence of Solver operations was:

Solver solver = new Solver(initial);

solver.moves() -> 10

solver.moves() -> 10

solver.moves() -> 10

solver.moves() -> 10

solver.isSolvable() -> true

solver.isSolvable() -> true

solver.solution()

solver.solution()

\* random 2-by-2 solvable boards

- failed after 4 calls to random methods in Solver

- first call to student moves() = 4

- last call to student moves() = 0

- any call to reference moves() = 4

- sequence of Solver operations was:

Solver solver = new Solver(initial);

solver.moves() -> 4

solver.solution()

solver.isSolvable() -> true

solver.moves() -> 0

- failed on trial 1 of 10

==> 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

\* puzzle4x4-unsolvable.txt

\* random 2-by-2 unsolvable boards

==> passed

Test 8: Call moves() with more file inputs

\* puzzle14.txt

\* puzzle15.txt

\* puzzle16.txt

\* puzzle17.txt

\* puzzle18.txt

\* puzzle19.txt

\* puzzle20.txt

\* puzzle21.txt

\* puzzle22.txt

\* puzzle23.txt

\* puzzle24.txt

\* puzzle25.txt

\* puzzle26.txt

\* puzzle27.txt

\* puzzle28.txt

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

OperationCountLimitExceededException

Number of calls to methods in Board exceeds limit: 100000000

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

==> FAILED

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

with an argument of the wrong type

\* puzzle00.txt

\* 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() = 6

- moves() = 0

\* 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() = 11

- moves() = 0

\* 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() = 16

- moves() = 0

==> passed

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

\* 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() = 2

- moves() = 0

\* 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() = 3

- moves() = 0

\* 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() = 4

- moves() = 0

\* 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() = 5

- moves() = 0

\* 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() = 6

- moves() = 0

\* 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() = 7

- moves() = 0

==> 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

\* puzzle3x3-19.txt

\* puzzle3x3-20.txt

\* puzzle3x3-21.txt

\* puzzle3x3-22.txt

\* puzzle3x3-23.txt

\* puzzle3x3-24.txt

\* puzzle3x3-25.txt

\* puzzle3x3-26.txt

\* puzzle3x3-27.txt

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

OperationCountLimitExceededException

Number of calls to methods in Board exceeds limit: 100000000

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

==> FAILED

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

\* puzzle3x3-00.txt

\* 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() = 2

- moves() = 0

\* 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() = 3

- moves() = 0

\* 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() = 4

- moves() = 0

\* 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() = 5

- moves() = 0

\* 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() = 6

- moves() = 0

\* 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() = 7

- moves() = 0

\* 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() = 8

- moves() = 0

\* 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() = 9

- moves() = 0

\* 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() = 10

- moves() = 0

\* 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() = 11

- moves() = 0

\* 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() = 12

- moves() = 0

\* 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() = 13

- moves() = 0

\* 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() = 14

- moves() = 0

\* 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() = 15

- moves() = 0

\* 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() = 16

- moves() = 0

\* 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() = 17

- moves() = 0

\* 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() = 18

- moves() = 0

\* puzzle3x3-18.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() = 19

- moves() = 0

\* puzzle3x3-19.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() = 20

- moves() = 0

\* puzzle3x3-20.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() = 21

- moves() = 0

\* puzzle3x3-21.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() = 22

- moves() = 0

\* puzzle3x3-22.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() = 23

- moves() = 0

\* puzzle3x3-23.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() = 24

- moves() = 0

\* puzzle3x3-24.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() = 25

- moves() = 0

\* puzzle3x3-25.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() = 26

- moves() = 0

\* puzzle3x3-26.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() = 27

- moves() = 0

\* puzzle3x3-27.txt

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

OperationCountLimitExceededException

Number of calls to methods in Board exceeds limit: 100000000

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

==> 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

\* puzzle4x4-19.txt

\* puzzle4x4-20.txt

\* puzzle4x4-21.txt

\* puzzle4x4-22.txt

\* puzzle4x4-23.txt

\* puzzle4x4-24.txt

\* puzzle4x4-25.txt

\* puzzle4x4-26.txt

\* puzzle4x4-27.txt

\* puzzle4x4-28.txt

\* puzzle4x4-29.txt

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

OperationCountLimitExceededException

Number of calls to methods in Board exceeds limit: 100000000

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

...

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.

25 一月 2017 在 6:23 晚上

70/100

否

25 一月 2017 在 1:33 下午

50/100

否