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

ASSESSMENT SUMMARY

```
Compilation:  PASSED
API:          PASSED

SpotBugs:     PASSED
PMD:          PASSED
Checkstyle:   PASSED

Correctness:  51/51 tests passed
Memory:       22/22 tests passed
Timing:       125/125 tests passed

Aggregate score: 100.00%
[ Compilation: 5%, API: 5%, Style: 0%, Correctness: 60%, Timing: 10%, Memory: 20% ]
```

ASSESSMENT DETAILS

```
The following files were submitted:
----------------------------------
7.0K Jul  7 16:25 Board.java
3.2K Jul  7 16:25 Solver.java


*********************************************************************************
*  COMPILING
*********************************************************************************


% javac Board.java
*-----------------------------------------------------------

% javac Solver.java
*-----------------------------------------------------------


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


Checking the APIs of your programs.
*-----------------------------------------------------------
Board:

Solver:

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


*********************************************************************************
*  CHECKING STYLE AND COMMON BUG PATTERNS
*********************************************************************************


% spotbugs *.class
*-----------------------------------------------------------
```

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


% pmd .
*-----------------------------------------------------------


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


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

% custom checkstyle checks for Board.java
*-----------------------------------------------------------

% custom checkstyle checks for Solver.java
*-----------------------------------------------------------


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


********************************************************************************
*  TESTING CORRECTNESS
********************************************************************************

Testing correctness of Board
*-----------------------------------------------------------
Running 26 total tests.

Tests 4-7 and 14-17 rely upon toString() returning results in prescribed format.

Test 1a: check hamming() with file inputs
  * puzzle04.txt
  * puzzle00.txt
  * puzzle07.txt
  * puzzle17.txt
  * puzzle27.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 1b: check 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: check manhattan() with file inputs
  * puzzle04.txt
  * puzzle00.txt
  * puzzle07.txt
  * puzzle17.txt
  * puzzle27.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 2b: check 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: check dimension() with random n-by-n boards
  * 2-by-2
  * 3-by-3
  * 4-by-4
  * 5-by-5
  * 6-by-6
==> passed

Test 4a: check toString() with file inputs
  * puzzle04.txt
  * puzzle00.txt
  * puzzle06.txt
  * puzzle09.txt
  * puzzle23.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 4b: check 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: check neighbors() with file inputs
  * puzzle04.txt
  * puzzle00.txt
  * puzzle06.txt
  * puzzle09.txt
  * puzzle23.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 5b: check 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: check neighbors() of neighbors() with file inputs
  * puzzle04.txt
  * puzzle00.txt
  * puzzle06.txt
  * puzzle09.txt
  * puzzle23.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 6b: check 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: check twin() with file inputs
```

```
  * puzzle04.txt
  * puzzle00.txt
  * puzzle06.txt
  * puzzle09.txt
  * puzzle23.txt
  * puzzle2x2-unsolvable1.txt
==> passed

Test 7b: check 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: check isGoal() with 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: check 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 that 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
  * transitive
  * argument is null
  * argument is of type String
  * argument is of type UncastableString
  * Board object stored in a variable of type Object
==> passed

Test 10b: check correctness of equals() on random n-by-n boards
  * n = 2
  * n = 3
  * n = 4
  * 5 <= n < 10
==> passed

Test 10c: check equals() when board sizes m and n are different
  * m = 4, n = 5
  * m = 2, n = 5
  * m = 5, n = 3
  * m = 2, n = 3
```

```
  * m = 3, 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: check dimension() on a board that is kth neighbor of a board
  * 0th neighbor of puzzle27.txt
  * 1st neighbor of puzzle27.txt
  * 2nd neighbor of puzzle27.txt
  * 13th neighbor of puzzle27.txt
  * 13th neighbor of puzzle00.txt
  * 13th neighbor of puzzle2x2-unsolvable1.txt
==> passed

Test 14: check hamming() on a board that is kth neighbor of a board
  * 0th neighbor of puzzle27.txt
  * 1st neighbor of puzzle27.txt
  * 2nd neighbor of puzzle27.txt
  * 13th neighbor of puzzle27.txt
  * 13th neighbor of puzzle00.txt
  * 13th neighbor of puzzle2x2-unsolvable1.txt
==> passed

Test 15: check manhattan() on a board that is a kth neighbor of a board
  * 0th neighbor of puzzle27.txt
  * 1st neighbor of puzzle27.txt
  * 2nd neighbor of puzzle27.txt
  * 13th neighbor of puzzle27.txt
  * 13th neighbor of puzzle00.txt
  * 13th neighbor of puzzle2x2-unsolvable1.txt
==> passed

Test 16: check hamming() on a board that is a kth twin of a board
  * 0th twin of puzzle27.txt
  * 1st twin of puzzle27.txt
  * 2nd twin of puzzle27.txt
  * 13th twin of puzzle27.txt
  * 13th twin of puzzle00.txt
  * 13th twin of puzzle2x2-unsolvable1.txt
==> passed

Test 17: check manhattan() on a board that is a kth twin of a board
  * 0th twin of puzzle27.txt
  * 1st twin of puzzle27.txt
  * 2nd twin of puzzle27.txt
  * 13th twin of puzzle27.txt
  * 13th twin of puzzle00.txt
  * 13th twin of puzzle2x2-unsolvable1.txt
==> passed


Total: 26/26 tests passed!


================================================================
********************************************************************************
*  MEMORY
```

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

Analyzing memory of Board
*-----------------------------------------------------------
Running 10 total tests.

Memory usage of an n-by-n board
[ must be at most 4n^2 + 32n + 64 bytes ]


              n       student (bytes)    reference (bytes)
---------------------------------------------------------
=> passed     2           152                  128
=> passed     3           216                  192
=> passed     4           264                  240
=> passed     8           584                  560
=> passed    12          1032                 1008
=> passed    16          1608                 1584
=> passed    20          2312                 2288
=> passed    37          6880                 6856
=> passed    72         23112                23088
=> passed   120         61512                61488
==> 10/10 tests passed

Total: 10/10 tests passed!

Student   memory = 4.00 n^2 + 32.00 n + 72.00   (R^2 = 1.000)
Reference memory = 4.00 n^2 + 32.00 n + 48.00   (R^2 = 1.000)


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



********************************************************************************
*  TESTING CORRECTNESS (substituting reference Board)
********************************************************************************

Testing correctness of Solver
*-----------------------------------------------------------
Running 25 total tests.

Test 1a: check 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 1b: check solution() 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 2a: check 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
  * puzzle29.txt
  * puzzle30.txt
  * puzzle31.txt
==> passed

Test 2b: check solution() 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
  * puzzle29.txt
  * puzzle30.txt
  * puzzle31.txt
==> passed

Test 3a: check moves() with random solvable n-by-n boards
  * 1000 random 3-by-3 boards that are exactly 1 move from goal
  * 1000 random 3-by-3 boards that are exactly 2 moves from goal
  * 1000 random 3-by-3 boards that are exactly 3 moves from goal
  * 1000 random 3-by-3 boards that are exactly 4 moves from goal
  * 1000 random 3-by-3 boards that are exactly 5 moves from goal
  * 1000 random 3-by-3 boards that are exactly 6 moves from goal
  * 1000 random 3-by-3 boards that are exactly 7 moves from goal
  * 1000 random 3-by-3 boards that are exactly 8 moves from goal
  * 1000 random 3-by-3 boards that are exactly 9 moves from goal
  * 1000 random 3-by-3 boards that are exactly 10 moves from goal
  * 1000 random 3-by-3 boards that are exactly 11 moves from goal
  * 1000 random 3-by-3 boards that are exactly 12 moves from goal
==> passed

Test 3b: check solution() with random solvable n-by-n boards
  * 1000 random 3-by-3 boards that are exactly 1 move from goal
  * 1000 random 3-by-3 boards that are exactly 2 moves from goal
  * 1000 random 3-by-3 boards that are exactly 3 moves from goal
  * 1000 random 3-by-3 boards that are exactly 4 moves from goal
```

```
  * 1000 random 3-by-3 boards that are exactly 5 moves from goal
  * 1000 random 3-by-3 boards that are exactly 6 moves from goal
  * 1000 random 3-by-3 boards that are exactly 7 moves from goal
  * 1000 random 3-by-3 boards that are exactly 8 moves from goal
  * 1000 random 3-by-3 boards that are exactly 9 moves from goal
  * 1000 random 3-by-3 boards that are exactly 10 moves from goal
  * 1000 random 3-by-3 boards that are exactly 11 moves from goal
  * 1000 random 3-by-3 boards that are exactly 12 moves from goal
==> passed

Test 4: 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 5a: call isSolvable() with file inputs
  * puzzle01.txt
  * puzzle03.txt
  * puzzle04.txt
  * puzzle17.txt
  * puzzle3x3-unsolvable1.txt
  * puzzle3x3-unsolvable2.txt
  * puzzle4x4-unsolvable.txt
==> passed

Test 5b: call isSolvable() on random n-by-n boards
  * 100 random 2-by-2 boards
==> passed

Test 6: check moves() on unsolvable puzzles
  * puzzle2x2-unsolvable1.txt
  * puzzle2x2-unsolvable2.txt
  * puzzle3x3-unsolvable1.txt
  * puzzle3x3-unsolvable2.txt
  * puzzle4x4-unsolvable.txt
==> passed

Test 7: check solution() on unsolvable puzzles
  * puzzle2x2-unsolvable1.txt
  * puzzle2x2-unsolvable2.txt
  * puzzle3x3-unsolvable1.txt
  * puzzle3x3-unsolvable2.txt
  * puzzle4x4-unsolvable.txt
==> passed

Test 8a: 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
  * puzzle3x3-05.txt
  * puzzle3x3-10.txt
  * random 2-by-2 solvable boards
==> passed

Test 8b: 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 9a: check that equals() method in Board is called
  * puzzle04.txt
  * puzzle05.txt
  * puzzle10.txt
==> passed
```

```
Test 9b: check that equals() method in Board is called only
         with an argument of type Board
  * puzzle00.txt
  * puzzle04.txt
  * puzzle05.txt
  * puzzle10.txt
==> passed

Test 9c: check that equals() method in Board is called only
         with a neighbor of a neighbor as an argument
  * puzzle00.txt
  * puzzle04.txt
  * puzzle05.txt
  * puzzle10.txt
  * puzzle27.txt
==> passed

Test 10: check that constructor throws exception if board is null
==> passed

Test 11a: check 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 11b: check solution() 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 12a: check 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
```

```
  * puzzle3x3-28.txt
  * puzzle3x3-29.txt
  * puzzle3x3-30.txt
==> passed

Test 12b: check solution() 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
  * puzzle3x3-28.txt
  * puzzle3x3-29.txt
  * puzzle3x3-30.txt
==> passed

Test 13a: check 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
  * puzzle4x4-30.txt
==> passed

Test 13b: check solution() 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
  * puzzle4x4-30.txt
==> passed

Test 14a: check moves() with random solvable n-by-n boards
  * 100 random 2-by-2 boards that are <= 6 moves from goal
  * 200 random 3-by-3 boards that are <= 20 moves from goal
  * 200 random 4-by-4 boards that are <= 20 moves from goal
  * 200 random 5-by-5 boards that are <= 20 moves from goal
==> passed

Test 14b: check solution() with random solvable n-by-n boards
  * 100 random 2-by-2 boards that are <= 6 moves from goal
  * 200 random 3-by-3 boards that are <= 20 moves from goal
  * 200 random 4-by-4 boards that are <= 20 moves from goal
  * 200 random 5-by-5 boards that are <= 20 moves from goal
==> passed


Total: 25/25 tests passed!


================================================================
********************************************************************************
*  MEMORY (substituting reference Board)
********************************************************************************

Analyzing memory of Solver
*-----------------------------------------------------------
Running 12 total tests.

Maximum allowed time per puzzle is 5.0 seconds.
Maximum allowed memory per puzzle = 200000000 bytes.

Test 1: Measure memory of Solver.
```

```
               filename   moves   memory
-----------------------------------------
=> passed  puzzle10.txt      10     4640
=> passed  puzzle15.txt      15     5568
=> passed  puzzle20.txt      20     2752
=> passed  puzzle25.txt      25     3392
=> passed  puzzle30.txt      30     4032
=> passed  puzzle35.txt      35     5536
==> 6/6 tests passed



Test 2: Measure memory of MinPQ.

                             deep              max           ending
               filename     memory             size           size
-----------------------------------------------------------------------
=> passed  puzzle10.txt      12560              18             17
=> passed  puzzle15.txt      15928              27             26
=> passed  puzzle20.txt     171128             440            439
=> passed  puzzle25.txt    1147832            2921           2920
=> passed  puzzle30.txt    5232400           13043          13042
=> passed  puzzle35.txt   42273328          117445         117444
==> 6/6 tests passed


Total: 12/12 tests passed!


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



*********************************************************************************
*  TIMING (substituting reference Board)
*********************************************************************************

Timing Solver
*-----------------------------------------------------------
Running 125 total tests.

Maximum allowed time per puzzle is 5.0 seconds.

Test 1: Measure CPU time and check correctness

               filename   moves    n  seconds
-----------------------------------------------
=> passed  puzzle20.txt      20    3     0.01
=> passed  puzzle22.txt      22    3     0.00
=> passed  puzzle21.txt      21    3     0.00
=> passed  puzzle23.txt      23    3     0.01
=> passed  puzzle24.txt      24    3     0.00
=> passed  puzzle25.txt      25    3     0.01
=> passed  puzzle27.txt      27    3     0.01
=> passed  puzzle29.txt      29    3     0.01
=> passed  puzzle26.txt      26    3     0.00
=> passed  puzzle28.txt      28    3     0.01
=> passed  puzzle30.txt      30    3     0.02
=> passed  puzzle31.txt      31    3     0.02
=> passed  puzzle39.txt      39    4     0.03
=> passed  puzzle41.txt      41    5     0.02
=> passed  puzzle34.txt      34    4     0.10
=> passed  puzzle37.txt      37    4     0.04
=> passed  puzzle44.txt      44    5     0.11
=> passed  puzzle32.txt      32    4     0.26
=> passed  puzzle35.txt      35    4     0.16
=> passed  puzzle33.txt      33    4     0.30
=> passed  puzzle43.txt      43    4     0.43
```

```
=> passed  puzzle46.txt      46    4     0.34
=> passed  puzzle40.txt      40    4     0.20
=> passed  puzzle36.txt      36    4     1.39
=> passed  puzzle45.txt      45    4     0.65
==> 25/25 tests passed


Test 2: Count MinPQ operations

               filename   insert()         delMin()
-------------------------------------------------------
=> passed  puzzle20.txt       1094              655
=> passed  puzzle22.txt       2785             1637
=> passed  puzzle21.txt       1926             1146
=> passed  puzzle23.txt       4356             2595
=> passed  puzzle24.txt       2432             1477
=> passed  puzzle25.txt       7330             4410
=> passed  puzzle27.txt       5927             3624
=> passed  puzzle29.txt      10820             6679
=> passed  puzzle26.txt       5195             3163
=> passed  puzzle28.txt      15548             9490
=> passed  puzzle30.txt      33556            20514
=> passed  puzzle31.txt      39672            24356
=> passed  puzzle39.txt      40441            20085
=> passed  puzzle41.txt      28268            12148
=> passed  puzzle34.txt     146923            70661
=> passed  puzzle37.txt      63048            30332
=> passed  puzzle44.txt     150941            67527
=> passed  puzzle32.txt     397992           191266
=> passed  puzzle35.txt     234308           116864
=> passed  puzzle33.txt     463133           221486
=> passed  puzzle43.txt     647906           319163
=> passed  puzzle46.txt     517612           257214
=> passed  puzzle40.txt     312170           154929
=> passed  puzzle36.txt    1982629           967198
=> passed  puzzle45.txt     925599           459483
==> 25/25 tests passed


Test 3: Count Board operations (that should not get called)

               filename    hamming()   toString()
-------------------------------------------------
=> passed  puzzle20.txt            0            0
=> passed  puzzle22.txt            0            0
=> passed  puzzle21.txt            0            0
=> passed  puzzle23.txt            0            0
=> passed  puzzle24.txt            0            0
=> passed  puzzle25.txt            0            0
=> passed  puzzle27.txt            0            0
=> passed  puzzle29.txt            0            0
=> passed  puzzle26.txt            0            0
=> passed  puzzle28.txt            0            0
=> passed  puzzle30.txt            0            0
=> passed  puzzle31.txt            0            0
=> passed  puzzle39.txt            0            0
=> passed  puzzle41.txt            0            0
=> passed  puzzle34.txt            0            0
=> passed  puzzle37.txt            0            0
=> passed  puzzle44.txt            0            0
=> passed  puzzle32.txt            0            0
=> passed  puzzle35.txt            0            0
=> passed  puzzle33.txt            0            0
=> passed  puzzle43.txt            0            0
=> passed  puzzle46.txt            0            0
=> passed  puzzle40.txt            0            0
=> passed  puzzle36.txt            0            0
```

```
=> passed  puzzle45.txt            0            0
==> 25/25 tests passed


Test 4a: Count Board operations (that should get called)

               filename    Board()           equals()         manhattan()
--------------------------------------------------------------------------
=> passed  puzzle20.txt       2401               1738               2403
=> passed  puzzle22.txt       6056               4415               6058
=> passed  puzzle21.txt       4215               3063               4217
=> passed  puzzle23.txt       9543               6942               9545
=> passed  puzzle24.txt       5383               3898               5385
=> passed  puzzle25.txt      16147              11731              16149
=> passed  puzzle27.txt      13172               9542              13174
=> passed  puzzle29.txt      24175              17490              24177
=> passed  puzzle26.txt      11518               8351              11520
=> passed  puzzle28.txt      34525              25027              34527
=> passed  puzzle30.txt      74581              54063              74583
=> passed  puzzle31.txt      88381              64019              88383
=> passed  puzzle39.txt      80608              60517              80610
=> passed  puzzle41.txt      52561              40405              52563
=> passed  puzzle34.txt     288242             217577             288244
=> passed  puzzle37.txt     123709              93371             123711
=> passed  puzzle44.txt     285992             218457             285994
=> passed  puzzle32.txt     780521             589247             780523
=> passed  puzzle35.txt     468033             351161             468035
=> passed  puzzle33.txt     906102             684610             906104
=> passed  puzzle43.txt    1286229             967060            1286231
=> passed  puzzle46.txt    1032037             774817            1032039
=> passed  puzzle40.txt     622025             467092             622027
=> passed  puzzle36.txt    3917022            2949816            3917024
=> passed  puzzle45.txt    1844562            1385073            1844564
==> 25/25 tests passed


Test 4b: count Board operations (that should get called),
         rejecting if doesn't adhere to stricter caching limits

               filename    Board()           equals()         manhattan()
--------------------------------------------------------------------------
=> passed  puzzle20.txt       2401               1738               2403
=> passed  puzzle22.txt       6056               4415               6058
=> passed  puzzle21.txt       4215               3063               4217
=> passed  puzzle23.txt       9543               6942               9545
=> passed  puzzle24.txt       5383               3898               5385
=> passed  puzzle25.txt      16147              11731              16149
=> passed  puzzle27.txt      13172               9542              13174
=> passed  puzzle29.txt      24175              17490              24177
=> passed  puzzle26.txt      11518               8351              11520
=> passed  puzzle28.txt      34525              25027              34527
=> passed  puzzle30.txt      74581              54063              74583
=> passed  puzzle31.txt      88381              64019              88383
=> passed  puzzle39.txt      80608              60517              80610
=> passed  puzzle41.txt      52561              40405              52563
=> passed  puzzle34.txt     288242             217577             288244
=> passed  puzzle37.txt     123709              93371             123711
=> passed  puzzle44.txt     285992             218457             285994
=> passed  puzzle32.txt     780521             589247             780523
=> passed  puzzle35.txt     468033             351161             468035
=> passed  puzzle33.txt     906102             684610             906104
=> passed  puzzle43.txt    1286229             967060            1286231
=> passed  puzzle46.txt    1032037             774817            1032039
=> passed  puzzle40.txt     622025             467092             622027
=> passed  puzzle36.txt    3917022            2949816            3917024
=> passed  puzzle45.txt    1844562            1385073            1844564
==> 25/25 tests passed
```

Total: 125/125 tests passed!

==========