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

ASSESSMENT SUMMARY

Compilation: PASSED

API: PASSED

Findbugs: FAILED (6 warnings)

Checkstyle: FAILED (9 warnings)

Correctness: 23/41 tests passed

Memory: 1/1 tests passed

Timing: 17/41 tests passed

Aggregate score: 61.95%

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

ASSESSMENT DETAILS

The following files were submitted:

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

3.7K Jun 13 17:14 BruteCollinearPoints.java

6.4K Jun 13 17:14 FastCollinearPoints.java

4.5K Jun 13 17:14 Point.java

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

\* COMPILING

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

% javac Point.java

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

% javac BruteCollinearPoints.java

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

% javac FastCollinearPoints.java

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

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

Checking the APIs of your programs.

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

Point:

BruteCollinearPoints:

FastCollinearPoints:

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

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

\* CHECKING STYLE AND COMMON BUG PATTERNS

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

% findbugs \*.class

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

H D DLS\_DEAD\_LOCAL\_STORE DLS: Assigns a value to the local variable 'p' but that value is never used. At Point.java:[line 153]

L D UC\_USELESS\_VOID\_METHOD UC: The void method 'main()' appears to serve no purpose. At Point.java:[line 154]

M C RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE RCN: Checks whether the variable 'points' is null [line 32], even though it can't be null because it was previously dereferenced. At BruteCollinearPoints.java:[line 29]

L D FE\_FLOATING\_POINT\_EQUALITY FE: Tests for exact floating-point equality. Because floating-point calculations may involve rounding, the calculated values may be imprecise. At BruteCollinearPoints.java:[line 48]

M C RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE RCN: Checks whether the variable 'lineSegMap' is null [line 100], even though it can't be null because it was previously dereferenced. At FastCollinearPoints.java:[line 100]

M C RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE RCN: Checks whether the variable 'points' is null [line 123], even though it can't be null because it was previously dereferenced. At FastCollinearPoints.java:[line 122]

Warnings generated: 6

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

% checkstyle \*.java

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

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

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

BruteCollinearPoints.java:26:5: Define constructors after static and instance variables but before methods. [DeclarationOrder]

BruteCollinearPoints.java:56:40: 'for' is not followed by whitespace. [WhitespaceAfter]

BruteCollinearPoints.java:80:12: 'for' is not followed by whitespace. [WhitespaceAfter]

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

FastCollinearPoints.java:64: Comment matches to-do format 'TODO:'. [TodoComment]

FastCollinearPoints.java:85: Comment matches to-do format 'TODO:'. [TodoComment]

FastCollinearPoints.java:106:16: Conditional logic can be removed. [SimplifyBooleanReturn]

Checkstyle ends with 9 errors.

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

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

\* TESTING CORRECTNESS

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

Testing correctness of Point

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

Running 3 total tests.

Test 1: p.slopeTo(q)

\* positive infinite slope, where p and q have coordinates in [0, 500)

\* positive infinite slope, where p and q have coordinates in [0, 32768)

\* negative infinite slope, where p and q have coordinates in [0, 500)

\* negative infinite slope, where p and q have coordinates in [0, 32768)

\* positive zero slope, where p and q have coordinates in [0, 500)

\* positive zero slope, where p and q have coordinates in [0, 32768)

\* symmetric for random points p and q with coordinates in [0, 500)

\* symmetric for random points p and q with coordinates in [0, 32768)

\* transitive for random points p, q, and r with coordinates in [0, 500)

\* transitive for random points p, q, and r with coordinates in [0, 32768)

\* slopeTo(), where p and q have coordinates in [0, 500)

\* slopeTo(), where p and q have coordinates in [0, 32768)

\* slopeTo(), where p and q have coordinates in [0, 10)

\* throw a java.lang.NullPointerException if argument is null

==> passed

Test 2: p.compareTo(q)

\* reflexive, where p and q have coordinates in [0, 500)

\* reflexive, where p and q have coordinates in [0, 32768)

\* antisymmetric, where p and q have coordinates in [0, 500)

\* antisymmetric, where p and q have coordinates in [0, 32768)

\* transitive, where p, q, and r have coordinates in [0, 500)

\* transitive, where p, q, and r have coordinates in [0, 32768)

\* sign of compareTo(), where p and q have coordinates in [0, 500)

\* sign of compareTo(), where p and q have coordinates in [0, 32768)

\* sign of compareTo(), where p and q have coordinates in [0, 10)

\* throw java.lang.NullPointerException exception if argument is null

==> passed

Test 3: p.slopeOrder().compare(q, r)

\* reflexive, where p and q have coordinates in [0, 500)

\* reflexive, where p and q have coordinates in [0, 32768)

\* antisymmetric, where p, q, and r have coordinates in [0, 500)

\* antisymmetric, where p, q, and r have coordinates in [0, 32768)

\* transitive, where p, q, r, and s have coordinates in [0, 500)

\* transitive, where p, q, r, and s have coordinates in [0, 32768)

\* sign of compare(), where p, q, and r have coordinates in [0, 500)

\* sign of compare(), where p, q, and r have coordinates in [0, 32768)

\* sign of compare(), where p, q, and r have coordinates in [0, 10)

\* throw java.lang.NullPointerException if either argument is null

==> passed

Total: 3/3 tests passed!

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

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

\* TESTING CORRECTNESS (substituting reference Point and LineSegment)

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

Testing correctness of BruteCollinearPoints

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

Running 17 total tests.

The inputs satisfy the following conditions:

- no duplicate points

- no 5 (or more) points are collinear

- all x- and y-coordinates between 0 and 32,767

Test 1: Points from a file

\* filename = input8.txt

\* filename = equidistant.txt

\* filename = input40.txt

\* filename = input48.txt

==> passed

Test 2a: Points from a file with horizontal line segments

\* filename = horizontal5.txt

\* filename = horizontal25.txt

==> passed

Test 2b: Random horizontal line segments

\* 1 random horizontal line segment

\* 5 random horizontal line segments

\* 10 random horizontal line segments

\* 15 random horizontal line segments

==> passed

Test 3a: Points from a file with vertical line segments

\* filename = vertical5.txt

\* filename = vertical25.txt

==> passed

Test 3b: Random vertical line segments

\* 1 random vertical line segment

\* 5 random vertical line segments

\* 10 random vertical line segments

\* 15 random vertical line segments

==> passed

Test 4a: Points from a file with no line segments

\* filename = random23.txt

\* filename = random38.txt

==> passed

Test 4b: Random points with no line segments

\* 5 random points

\* 10 random points

\* 20 random points

\* 50 random points

==> passed

Test 5: Points from a file with fewer than 4 points

\* filename = input1.txt

java.lang.IllegalArgumentException: Less than 4 points.

BruteCollinearPoints.<init>(BruteCollinearPoints.java:36)

TestBruteCollinearPoints.testSegments(TestBruteCollinearPoints.java:110)

TestBruteCollinearPoints.file(TestBruteCollinearPoints.java:154)

TestBruteCollinearPoints.test5(TestBruteCollinearPoints.java:287)

TestBruteCollinearPoints.main(TestBruteCollinearPoints.java:736)

\* filename = input2.txt

java.lang.IllegalArgumentException: Less than 4 points.

BruteCollinearPoints.<init>(BruteCollinearPoints.java:36)

TestBruteCollinearPoints.testSegments(TestBruteCollinearPoints.java:110)

TestBruteCollinearPoints.file(TestBruteCollinearPoints.java:154)

TestBruteCollinearPoints.test5(TestBruteCollinearPoints.java:288)

TestBruteCollinearPoints.main(TestBruteCollinearPoints.java:736)

\* filename = input3.txt

java.lang.IllegalArgumentException: Less than 4 points.

BruteCollinearPoints.<init>(BruteCollinearPoints.java:36)

TestBruteCollinearPoints.testSegments(TestBruteCollinearPoints.java:110)

TestBruteCollinearPoints.file(TestBruteCollinearPoints.java:154)

TestBruteCollinearPoints.test5(TestBruteCollinearPoints.java:289)

TestBruteCollinearPoints.main(TestBruteCollinearPoints.java:736)

==> FAILED

Test 6: Check for dependence on either compareTo() or compare()

returning { -1, +1, 0 } instead of { negative integer,

positive integer, zero }

\* filename = equidistant.txt

\* filename = input40.txt

\* filename = input48.txt

==> passed

Test 7: Check for fragile dependence on return value of toString()

\* filename = equidistant.txt

\* filename = input40.txt

\* filename = input48.txt

==> passed

Test 8: Random line segments, none vertical or horizontal

\* 1 random line segment

\* 5 random line segments

\* 10 random line segments

\* 15 random line segments

==> passed

Test 9: Random line segments

\* 1 random line segment

\* 5 random line segments

\* 10 random line segments

\* 15 random line segments

==> passed

Test 10: Check that data type is immutable by testing whether each method

returns the same value, regardless of any intervening operations

\* input8.txt

\* equidistant.txt

==> passed

Test 11: Check that data type does not mutate the constructor argument

\* input8.txt

\* equidistant.txt

==> passed

Test 12: numberOfSegments() is consistent with segments()

\* filename = input8.txt

\* filename = equidistant.txt

\* filename = input40.txt

\* filename = input48.txt

\* filename = horizontal5.txt

\* filename = vertical5.txt

\* filename = random23.txt

==> passed

Test 13: Throws exception either if argument to constructor is null

or if any entry in array is null

\* argument is null

\* Point[] of length 10, number of null entries = 1

\* Point[] of length 10, number of null entries = 10

\* Point[] of length 4, number of null entries = 1

\* Point[] of length 3, number of null entries = 1

\* Point[] of length 2, number of null entries = 1

\* Point[] of length 1, number of null entries = 1

==> passed

Test 14: Check that the constructor throws an exception if duplicate points

\* 50 points

\* 25 points

\* 5 points

\* 4 points

\* 3 points

\* 2 points

==> passed

Total: 16/17 tests passed!

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

Testing correctness of FastCollinearPoints

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

Running 21 total tests.

The inputs satisfy the following conditions:

- no duplicate points

- all x- and y-coordinates between 0 and 32,767

Test 1: Points from a file

\* filename = input8.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test1(TestFastCollinearPoints.java:159)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:779)

\* filename = equidistant.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test1(TestFastCollinearPoints.java:160)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:779)

\* filename = input40.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test1(TestFastCollinearPoints.java:161)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:779)

\* filename = input48.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test1(TestFastCollinearPoints.java:162)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:779)

\* filename = input299.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test1(TestFastCollinearPoints.java:163)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:779)

==> FAILED

Test 2a: Points from a file with horizontal line segments

\* filename = horizontal5.txt

- segments() contains the same segment more than once

- segment 0: (5067, 14118) -> (7821, 14118)

- segment 1: (5067, 14118) -> (7821, 14118)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (5067, 14118) -> (7821, 14118)

- reference segment 4: (2682, 14118) -> (5067, 14118) -> (7453, 14118) -> (7821, 14118)

- student solution has 20 non-null entries

- reference solution has 5 non-null entries

- 18 extra entries in student solution, including: (5067, 14118) -> (7821, 14118)

- 3 missing entries in student solution, including: (2682, 14118) -> (5067, 14118) -> (7453, 14118) -> (7821, 14118)

\* filename = horizontal25.txt

- segments() contains the same segment more than once

- segment 0: (10653, 13870) -> (18005, 13870)

- segment 1: (10653, 13870) -> (18005, 13870)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (10653, 13870) -> (18005, 13870)

- reference segment 13: (2566, 13870) -> (10653, 13870) -> (18005, 13870) -> (19459, 13870)

- student solution has 100 non-null entries

- reference solution has 25 non-null entries

- 84 extra entries in student solution, including: (9880, 20913) -> (16352, 20913)

- 9 missing entries in student solution, including: (6268, 18593) -> (11710, 18593) -> (12984, 18593) -> (19710, 18593)

\* filename = horizontal50.txt

- segments() contains the same segment more than once

- segment 0: (6518, 7218) -> (17289, 7218)

- segment 1: (6518, 7218) -> (17289, 7218)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (6518, 7218) -> (17289, 7218)

- reference segment 17: (6300, 7218) -> (6518, 7218) -> (15233, 7218) -> (17289, 7218)

- student solution has 200 non-null entries

- reference solution has 50 non-null entries

- 175 extra entries in student solution, including: (14800, 20754) -> (17428, 20754)

- 25 missing entries in student solution, including: (5249, 20754) -> (5559, 20754) -> (14800, 20754) -> (17428, 20754)

\* filename = horizontal75.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test2a(TestFastCollinearPoints.java:172)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:782)

\* filename = horizontal100.txt

- segments() contains the same segment more than once

- segment 0: (1238, 6140) -> (11759, 6140)

- segment 1: (1238, 6140) -> (11759, 6140)

- segments() contains a subsegment of a segment in reference solution

- student segment 2: (1238, 6140) -> (11219, 6140)

- reference segment 30: (1238, 6140) -> (1773, 6140) -> (11219, 6140) -> (11759, 6140)

- student solution has 400 non-null entries

- reference solution has 100 non-null entries

- 339 extra entries in student solution, including: (16154, 20698) -> (19642, 20698)

- 39 missing entries in student solution, including: (5835, 20698) -> (7673, 20698) -> (16154, 20698) -> (19642, 20698)

==> FAILED

Test 2b: Random horizontal line segments

\* 1 random horizontal line segment

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.horizontalSegments(TestFastCollinearPoints.java:185)

TestFastCollinearPoints.test2b(TestFastCollinearPoints.java:227)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:785)

- failed on trial 1 of 500

4

15787 1095

20138 1095

11640 1095

4432 1095

\* 5 random horizontal line segments

- segments() contains the same segment more than once

- segment 0: (18172, 12421) -> (20475, 12421)

- segment 6: (18172, 12421) -> (20475, 12421)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (18172, 12421) -> (20475, 12421)

- reference segment 2: (4745, 12421) -> (13855, 12421) -> (18172, 12421) -> (20475, 12421)

- student solution has 20 non-null entries

- reference solution has 5 non-null entries

- 17 extra entries in student solution, including: (18920, 19844) -> (20173, 19844)

- 2 missing entries in student solution, including: (2480, 19844) -> (3500, 19844) -> (18920, 19844) -> (20173, 19844)

- failed on trial 1 of 250

20

4745 12421

15453 3885

3500 19844

9379 14743

13911 3885

7690 6234

13855 12421

5608 14743

2480 19844

14936 3885

2940 14743

10329 14743

20475 12421

18920 19844

18172 12421

20173 19844

17556 6234

8815 6234

8782 6234

1027 3885

\* 10 random horizontal line segments

- segments() contains the same segment more than once

- segment 9: (12309, 12933) -> (19558, 12933)

- segment 25: (12309, 12933) -> (19558, 12933)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (1974, 18977) -> (9649, 18977)

- reference segment 7: (1974, 18977) -> (8659, 18977) -> (9649, 18977) -> (19739, 18977)

- student solution has 40 non-null entries

- reference solution has 10 non-null entries

- 34 extra entries in student solution, including: (10650, 20838) -> (14459, 20838)

- 4 missing entries in student solution, including: (1191, 19135) -> (15661, 19135) -> (16322, 19135) -> (18572, 19135)

- failed on trial 1 of 50

\* 15 random horizontal line segments

- segments() contains the same segment more than once

- segment 9: (4782, 18112) -> (13036, 18112)

- segment 21: (4782, 18112) -> (13036, 18112)

- segments() contains a subsegment of a segment in reference solution

- student segment 0: (7233, 13052) -> (18005, 13052)

- reference segment 8: (4205, 13052) -> (7233, 13052) -> (18005, 13052) -> (20029, 13052)

- student solution has 60 non-null entries

- reference solution has 15 non-null entries

- 53 extra entries in student solution, including: (9396, 19775) -> (20940, 19775)

- 8 missing entries in student solution, including: (4610, 18109) -> (8227, 18109) -> (9640, 18109) -> (10970, 18109)

- failed on trial 1 of 5

==> FAILED

Test 3a: Points from a file with vertical line segments

\* filename = vertical5.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test3a(TestFastCollinearPoints.java:236)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:788)

\* filename = vertical25.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test3a(TestFastCollinearPoints.java:237)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:788)

\* filename = vertical50.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test3a(TestFastCollinearPoints.java:238)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:788)

\* filename = vertical75.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test3a(TestFastCollinearPoints.java:239)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:788)

\* filename = vertical100.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test3a(TestFastCollinearPoints.java:240)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:788)

==> FAILED

Test 3b: Random vertical line segments

\* 1 random vertical line segment

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.verticalSegments(TestFastCollinearPoints.java:209)

TestFastCollinearPoints.test3b(TestFastCollinearPoints.java:246)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:791)

- failed on trial 1 of 500

4

5886 3177

5886 19732

5886 14358

5886 8635

\* 5 random vertical line segments

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.verticalSegments(TestFastCollinearPoints.java:209)

TestFastCollinearPoints.test3b(TestFastCollinearPoints.java:247)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:791)

- failed on trial 1 of 250

20

10035 3003

16272 1052

16272 5818

9925 3991

10035 3237

9925 20971

13092 12916

3684 4191

10035 9510

9925 3696

10035 5840

3684 6026

3684 1432

13092 6487

16272 7861

3684 8241

16272 10576

13092 5295

9925 9295

13092 12171

\* 10 random vertical line segments

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.verticalSegments(TestFastCollinearPoints.java:209)

TestFastCollinearPoints.test3b(TestFastCollinearPoints.java:248)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:791)

- failed on trial 1 of 50

\* 15 random vertical line segments

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.verticalSegments(TestFastCollinearPoints.java:209)

TestFastCollinearPoints.test3b(TestFastCollinearPoints.java:249)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:791)

- failed on trial 1 of 5

==> FAILED

Test 4a: Points from a file with no line segments

\* filename = random23.txt

\* filename = random38.txt

\* filename = random91.txt

\* filename = random152.txt

==> passed

Test 4b: Random points with no line segments

\* 5 random points

\* 10 random points

\* 20 random points

\* 50 random points

==> passed

Test 5a: Points from a file with 5 or more on some line segments

\* filename = input9.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test5a(TestFastCollinearPoints.java:296)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:800)

\* filename = input10.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastCollinearPoints.testSegments(TestFastCollinearPoints.java:110)

TestFastCollinearPoints.file(TestFastCollinearPoints.java:154)

TestFastCollinearPoints.test5a(TestFastCollinearPoints.java:297)

TestFastCollinearPoints.main(TestFastCollinearPoints.java:800)

\* filename = input20.txt

java.lang.UnsupportedOperationException: calling hashCode() is not permitted on this assignment

Point.hashCode(Point.java:274)

java.util.HashMap.hash(HashMap.java:338)

java.util.HashMap.put(HashMap.java:611)

FastCollinearPoints.<init>(FastCollinearPoints.java:67)

TestFastColli

...

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.