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

ASSESSMENT SUMMARY

Compilation: PASSED
API: PASSED

SpotBugs: PASSED
PMD: PASSED
Checkstyle: FAILED (0 errors, 13 warnings)

Correctness: 34/34 tests passed
Memory: 2/2 tests passed
Timing: 51/51 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:

2.2K Aug 28 02:43 Clock.java
1.9K Aug 28 02:43 ColorHSB.java

****
* COMPILING
****

% javac ColorHSB.java
*---

% javac Clock.java
*---

==

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

Clock:

==

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


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


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


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


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


% checkstyle *.java
*---------------------------------------------------------
[WARN] Clock.java:38:81: '{' is not preceded with whitespace. [WhitespaceAround]
[WARN] Clock.java:40:16: Conditional logic can be removed. [SimplifyBooleanReturn]
[WARN] Clock.java:40:50: '{' is not preceded with whitespace. [WhitespaceAround]
Checkstyle ends with 0 errors and 3 warnings.

% custom checkstyle checks for ColorHSB.java
*---------------------------------------------------------
[WARN] ColorHSB.java:5:26: '359' looks like an unnecessary constant. [MagicNumber]
Checkstyle ends with 0 errors and 1 warning.

% custom checkstyle checks for Clock.java
*---------------------------------------------------------
[WARN] Clock.java:1:1: '23' is an unnecessary numeric literal because it is derived from the
number of hours per day (24) or the number of minutes per hour (60). [NumericLiteralCount]
[WARN] Clock.java:1:1: '59' is an unnecessary numeric literal because it is derived from the
number of hours per day (24) or the number of minutes per hour (60). [NumericLiteralCount]
[WARN] Clock.java:1:1: The numeric literal '23' appears 2 times. Define a constant variable
(such as 'HOURS_PER_DAY'). [NumericLiteralCount]
[WARN] Clock.java:1:1: The numeric literal '59' appears 2 times. Define a constant variable
(such as 'MINUTES_PER_HOUR'). [NumericLiteralCount]
[WARN] Clock.java:1:1: The numeric literal '60' appears 3 times. Define a constant variable
(such as 'MINUTES_PER_HOUR'). [NumericLiteralCount]
[WARN] Clock.java:63:24: The 'main()' method must directly call the public constructor
'Clock()'. [MainCallsAllPublicMethods]
```

[WARN] Clock.java:63:24: The 'main()' method must directly call the public method 'isEarlierThan()'. [MainCallsAllPublicMethods]
[WARN] Clock.java:63:24: The 'main()' method must directly call the public method 'tic()'. [MainCallsAllPublicMethods]
[WARN] Clock.java:63:24: The 'main()' method must directly call the public method 'toc()'. [MainCallsAllPublicMethods]
Checkstyle ends with 0 errors and 9 warnings.

==========

****
* TESTING CORRECTNESS
****

Testing correctness of ColorHSB
*---
Running 17 total tests.

Test 1a: construct a ColorHSB object and call toString()
  * (25, 84, 97)
  * (0, 0, 0)
  * (359, 100, 100)
==> passed

Test 1b: construct random ColorHSB objects and call toString()
  * h <= 359, s <= 100, b <= 100
  * h <= 60, s <= 10, b <= 10
==> passed

Test 2a: construct a ColorHSB object and call isGrayscale()
  * (25, 84, 97)
  * (0, 0, 0)
  * (0, 50, 0)
  * (0, 0, 50)
  * (359, 100, 100)
==> passed

Test 2b: construct random ColorHSB objects and call isGrayscale()
  * h <= 359, s <= 100, b <= 100
  * h =   0, s <= 100, b <= 100
  * h <= 359, s =   0, b <= 100
  * h <= 359, s <= 100, b =   0
  * h <= 359, s =   0, b =   0
  * h <= 10, s <= 10, b <= 10
==> passed

Test 3a: construct two ColorHSB objects and call distanceSquaredTo()
  * (350, 100, 45) to (0, 100, 50)
  * (25, 84, 97) to (0, 100, 100)
  * (25, 84, 97) to (26, 85, 96)
  * (180, 100, 100) to (0, 0, 0)
==> passed

Test 3b: construct random pairs of ColorHSB objects and call distanceSquaredTo()
  * h <= 359, s <= 100, b <= 100
  * h <= 60, s <= 10, b <= 10
  * h <= 359, s <= 100, b =   0
  * h <= 359, s =   0, b <= 100
  * h =   0, s <= 100, b <= 100
==> passed

Test 3c: construct random pairs of ColorHSB objects and check that
         distanceSquaredTo() is symmetric
  * h <= 359, s <= 100, b <= 100
  * h <= 60, s <= 10, b <= 10
  * h <= 359, s <= 100, b =   0
  * h <= 359, s =   0, b <= 100
  * h =   0, s <= 100, b <= 100
==> passed

Test 4: create two ColorHSB objects and call the methods isGrayscale(),
        toString(), and distanceSquaredTo() with probabilities (p1, p2, p3)
        and check that they return the same values in each call
  * 100 random calls (0.8, 0.1, 0.1)
  * 100 random calls (0.8, 0.1, 0.1)
  * 100 random calls (0.1, 0.8, 0.1)
  * 100 random calls (0.1, 0.8, 0.1)
  * 100 random calls (0.1, 0.1, 0.8)
  * 100 random calls (0.1, 0.1, 0.8)
==> passed

Test 5a: check formatting of main() for inputs from exam specification
  % java-introcs ColorHSB 25 84 97 < web.txt
  Red (0, 100, 100)

  % java-introcs ColorHSB 350 100 45 < web.txt
  Maroon (0, 100, 50)

  % java-introcs ColorHSB 25 84 97 < wiki.txt
  Princeton_Orange (26, 85, 96)

==> passed

Test 5b: check that main() reads all data from standard input

```
  * java-introcs ColorHSB 25 84 97 < web.txt
  * java-introcs ColorHSB 350 100 45 < web.txt
  * java-introcs ColorHSB 25 84 97 < wiki.txt
==> passed

Test 5c: check correctness of main() for inputs from exam specification
  * java-introcs ColorHSB 25 84 97 < web.txt
  * java-introcs ColorHSB 350 100 45 < web.txt
  * java-introcs ColorHSB 25 84 97 < wiki.txt
==> passed

Test 6: check main() computes closest color
  * web.txt
  * crayola.txt
  * wiki.txt
==> passed


Tests 7a to 10 test that main() computes the closest color with respect
to the student's distanceSquaredTo() method, even if that method returns
incorrect values.

Test 7a: check main() with random command-line arguments
  * web.txt
  * crayola.txt
  * wiki.txt
==> passed

Test 7b: check main() with random input files
  * random input files with 10 pre-defined colors
  * random input files with 20 pre-defined colors
  * random input files with 2 pre-defined colors
  * random input files with 1 pre-defined color
==> passed

Test 8: check main() with ties possible for closest color
  * wiki.txt
  * grayscale.txt
  * random input files with 10 pre-defined colors
  * random input files with 20 pre-defined colors
==> passed

Test 9: check main() with large input files
  * random input files with 2000 pre-defined colors
  * random input files with 5000 pre-defined colors
  * random input files with 10000 pre-defined colors
==> passed
```

```
Test 10: check main() when distance is very large
  * colors1.txt
  * colors8.txt
  * colors10.txt
==> passed

Total: 17/17 tests passed!


================================================================
Testing correctness of Clock
*----------------------------------------------------------
Running 17 total tests.

Test 1a: construct a Clock object; check formatting of toString()
  * hours = 11, minutes = 59
  * hours =  3, minutes = 30
  * hours = 16, minutes =  5
  * hours =  0, minutes =  0
  * hours = 23, minutes = 59
==> passed

Test 1b: construct random Clock objects; check formatting of toString()
  * 10000 trials with 10 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with  0 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with 10 <= hours < 24,  0 <= minutes < 60
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60
==> passed

Test 2a: construct a Clock object; check toString()
  * hours = 11, minutes = 59
  * hours =  3, minutes = 30
  * hours =  0, minutes =  0
  * hours = 23, minutes = 59
==> passed

Test 2b: construct random Clock objects; check toString()
  * 10000 trials with 10 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with  0 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with 10 <= hours < 24,  0 <= minutes < 60
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60
==> passed

Test 3: construct two random Clock objects; check isEarlierThan()
  * 10000 trials (earlier)
  * 10000 trials (later)
  * 10000 trials (equal)
  * 10000 trials (reference equal)
```

==> passed

Test 4a: construct a Clock object; call tic(); check toString()
  * hours = 12, minutes = 34
  * hours = 3, minutes = 30
  * hours = 0, minutes = 0
  * hours = 23, minutes = 0
==> passed

Test 4b: construct a Clock object; call tic(); check toString()
  * hours = 0, minutes = 59
  * hours = 1, minutes = 59
  * hours = 2, minutes = 59
  * hours = 11, minutes = 59
  * hours = 12, minutes = 59
  * hours = 22, minutes = 59
  * hours = 23, minutes = 59
==> passed

Test 4c: construct random Clock objects; call tic(); check toString()
  * 10000 trials with 10 <= hours < 23, 10 <= minutes < 59
  * 10000 trials with 0 <= hours < 23, 10 <= minutes < 59
  * 10000 trials with 10 <= hours < 23, 0 <= minutes < 59
  * 10000 trials with 0 <= hours < 24, 0 <= minutes < 60
==> passed

Test 5a: construct a Clock object; call toc(); check toString()
  * hours = 12, minutes = 34, delta = 5
  * hours = 3, minutes = 30, delta = 10
  * hours = 0, minutes = 0, delta = 45
  * hours = 23, minutes = 0, delta = 1
  * hours = 23, minutes = 0, delta = 0
==> passed

Test 5b: construct a Clock object; call toc(); check toString()
  * hours = 0, minutes = 59, delta = 10
  * hours = 1, minutes = 59, delta = 60
  * hours = 2, minutes = 59, delta = 1
  * hours = 11, minutes = 59, delta = 45
  * hours = 12, minutes = 59, delta = 120
  * hours = 22, minutes = 59, delta = 1440
  * hours = 23, minutes = 59, delta = 100
  * hours = 0, minutes = 0, delta = 30000
==> passed

Test 5c: construct random Clock objects; call toc(); check toString()
  * 10000 trials with 10 <= hours < 23, 10 <= minutes < 59, 0 <= delta < 60
  * 10000 trials with 0 <= hours < 23, 10 <= minutes < 59, 0 <= delta < 60

```
  * 10000 trials with 10 <= hours < 23,  0 <= minutes < 59,  0 <= delta < 60
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60,  0 <= delta < 60
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60,  0 <= delta < 1440
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60,  0 <= delta < 14400
==> passed

Test 6a: construct random Clock objects with 1-argument constructor; check toString()
  * time = 11:59
  * time = 03:30
  * time = 16:05
  * time = 00:00
  * time = 23:59
==> passed

Test 6b: construct random Clock objects with 1-argument constructor; check toString()
  * 10000 trials with 10 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with  0 <= hours < 24, 10 <= minutes < 60
  * 10000 trials with 10 <= hours < 24,  0 <= minutes < 60
  * 10000 trials with  0 <= hours < 24,  0 <= minutes < 60
==> passed

Test 7: check two-argument constructor with invalid arguments
  * hours = 24, minutes =  0
  * hours = 24, minutes = 60
  * hours =  0, minutes = 60
  * hours = -1, minutes = 30
  * hours = 12, minutes = -1
  * hours =  0, minutes = -1
  * hours = -1, minutes = -1
  * hours = 59, minutes = 23
  * hours = -2147483648, minutes = -2147483648
  * hours = -2147483648, minutes = 2147483647
  * hours = 2147483647, minutes = -2147483648
  * hours = 2147483647, minutes = 2147483647
==> passed

Test 8: check 1-argument constructor with invalid arguments
  * time = "24:00"
  * time = "25:10"
  * time = "12:60"
  * time = "12:060"
  * time = "12:100"
  * time = "2:56"
  * time = "2:5"
  * time = "02:5"
  * time = "0:0"
  * time = "00007:23"
  * time = "07:000023"
```

```
  * time = "12,34"
  * time = "1234"
  * time = "HH:MM"
  * time = "0"
  * time = ""
  * time = "-12:34"
  * time = "0x06:34"
  * time = " 12:34"
  * time = "12:34 "
  * time = ":1234"
  * time = "1:234"
  * time = "123:4"
  * time = "1234:"
  * time = "12345"
==> passed

Test 9: construct Clock with 2-argument constructor; call toc() with negative delta
  * hours = 11, minutes = 59, delta = -1
  * hours = 23, minutes = 59, delta = -5
  * hours =  1, minutes = 23, delta = -10
  * hours = 16, minutes = 45, delta = -100
  * hours = 12, minutes =  0, delta = -1440
  * hours = 21, minutes = 30, delta = -14400
  * hours = 17, minutes = 18, delta = -2147483648
==> passed

Test 10: create two Clock objects from 2-argument constructor; check random
         intermixed sequence of calls to toString(), isEarlierThan(), tic(),
         and toc(), with probabilities (p1, p2, p3, p4), respectively
  * p = (0.5, 0.5, 0.0, 0.0)
  * p = (0.5, 0.0, 0.5, 0.0)
  * p = (0.5, 0.0, 0.0, 0.5)
  * p = (0.0, 0.5, 0.5, 0.0)
  * p = (0.0, 0.5, 0.0, 0.5)
  * p = (0.2, 0.2, 0.3, 0.3)
  * p = (0.2, 0.2, 0.3, 0.3)
==> passed

Total: 17/17 tests passed!


================================================================
********************************************************************************
*  MEMORY
********************************************************************************

Analyzing memory of ColorHSB
*-----------------------------------------------------------
```

```
Running 1 total tests.

Test 1: Memory usage per ColorHSB object
  * number bytes used by student   ColorHSB object = 32
  * number bytes used by reference ColorHSB object = 32
==> passed

Total: 1/1 tests passed!


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


Analyzing memory of Clock
*-----------------------------------------------------------
Running 1 total tests.

Test 1: Memory usage per Clock object
  * number bytes used by student   Clock object = 24
  * number bytes used by reference Clock object = 24
==> passed

Total: 1/1 tests passed!


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


********************************************************************************
*  TIMING
********************************************************************************

Timing Clock
*-----------------------------------------------------------
Running 51 total tests.

Test 1: create n Clock object; call toString() n times

                n  seconds
-----------------------------------
=> passed       1000     0.01
=> passed       2000     0.01
=> passed       4000     0.02
=> passed       8000     0.03
=> passed      10000     0.04
=> passed      20000     0.09
```

```
=> passed       40000     0.11
=> passed       80000     0.12
=> passed      100000     0.08
=> passed      200000     0.16

==> 10/10 tests passed


Test 2: create n Clock object; call isEarlierThan() n^2 times

                  n  seconds
-----------------------------------
=> passed         100     0.00
=> passed         200     0.00
=> passed         400     0.00
=> passed         800     0.00
=> passed        1000     0.00
=> passed        2000     0.01
=> passed        4000     0.06
=> passed        8000     0.23
=> passed       10000     0.36

==> 9/9 tests passed


Test 3: create a Clock object; call tic() n times

                  n  seconds
-----------------------------------
=> passed        1000     0.00
=> passed        2000     0.00
=> passed        4000     0.00
=> passed        8000     0.00
=> passed       10000     0.00
=> passed       20000     0.00
=> passed       40000     0.00
=> passed       80000     0.00
=> passed      100000     0.00
=> passed      200000     0.00
=> passed      400000     0.00
=> passed      800000     0.00
=> passed     1000000     0.00
=> passed     2000000     0.00
=> passed     4000000     0.00
=> passed     8000000     0.00

==> 16/16 tests passed
```

Test 4: create a Clock object; call toc(n) n times

```
                  n  seconds
---------------------------------
=> passed        1000     0.00
=> passed        2000     0.00
=> passed        4000     0.00
=> passed        8000     0.00
=> passed       10000     0.00
=> passed       20000     0.00
=> passed       40000     0.00
=> passed       80000     0.00
=> passed      100000     0.00
=> passed      200000     0.00
=> passed      400000     0.00
=> passed      800000     0.00
=> passed     1000000     0.00
=> passed     2000000     0.00
=> passed     4000000     0.00
=> passed     8000000     0.00
==> 16/16 tests passed
```

Total: 51/51 tests passed!

==========