Lab 1 Report

吳俊青
108598014
2020/03/21

1 Test Plan

1.1 Test requirements

The Lab 1 requires to

(1) select **15 methods** from **6 classes** of the SUT (GeoProject),

(2) design Unit test cases based on the experience or intuition for theselected methods,

(3) develop test scripts to implement the test cases,

(4) execute the test script on the selected methods, and

(5) report the test results.

In particular, based on the statement coverage criterion, the **test requirements** for Lab 1 are to design test cases for each selected method so that "*each statement of the method will be covered by at least one test case* and *the minimum statement coverage is 40%*".

1.2 Strategy

To satisfy the test requirements listed in Section 1, I decided to use the following strategy.

(1) Select those public methods that are easy to understand and have primitive types of input and output parameters (if possible).

(2) Set the objective of the minimum statement coverage to be 50% initially and (if necessary) adjust the objective based on the time available.

(3) Learn the necessary skills and tools as soon as possible.

(4) design the test cases for those selected methods by considering

  i. the possible **valid values** and **combinations** of the input parameters.

  ii. the **boundary values** of the input parameters.

1.3 Test activities

To implement the proposed strategy, the following activities are planned to perform.

No.	Activity Name	Plan hours	Schedule Date
1	Study GeoProject	2	2020/03/14
2	Learn JUnit	1	2020/03/14

3	Design test cases for the selected methods	3	2020/03/15
4	Implement test cases	6	2020/03/17
5	Perform test	1	2020/03/17
6	Refactor test cases	2	2020/03/22
7	Complete Lab1 report	1	2020/03/22

1.4 Success criteria

All test cases designed for the selected methods must pass (or "90% of all test cases must pass) and *the statement coverage should have achieved at least 50%*.

2 Test Design

To fulfill the test requirements listed in section 1.1, the following methods are selected and corresponding test cases are designed.

No	Class	Method	Test Objective	Inputs	Expected Outputs

1-1	Base32	encodeBase32()
		padLeftWithZerosToLength()
			Boundary value 能夠產生正確 output	1.i=9223372036854775807L
				2.i=9223372036854775807L,lenth=14
				3.i=-9223372036854775808L,lenth=9
					1.“7zzzzzzzzzzzz”
					2.”07zzzzzzzzzzzz”
					3.-8000000000000

1-2	Base32	decodeBase32()
		getCharIndex()
			Boundary value 能夠產生正確 output，且輸入例外資料，能夠產生預期 throw	1.hash="7zzzzzzzzzzzz"
				2.hash="-8000000000000"
				3.hash="-037a4ry"
					1.9223372036854775807L
					2.-9223372036854775808L
					3. expected = IllegalArgumentException.class

2-1	Coverage	toString()	利用不同建構式產生物件，並取得正確 output	1.Coverage({“037k4ry”,”37k4ry”,”7k4ry”}, 2.2)
				2.CoverageLongs({108, 598, 140}, 3, 1.7976931348623157E308)
					1."Coverage [hashes=[37k4ry, 037k4ry, 7k4ry], ratio=2.2]"
					2. "Coverage [hashes=[000000,000000000006,000000000008],ratio=1.7976931348623157E308]"

2-2	Coverage	getHashLength()	利用不同建構式產生物件，並取得正確 output	1.Coverage({“037k4ry”,”37k4ry”,”7k4ry”}, 2.2)
				2.CoverageLongs({108, 598, 140}, 3, 1.7976931348623157E308)
				3.Coverage({}, 2.2)
					1. 6
					2. 6
					3. 0

2-3	Coverage	getRatio()	利用不同建構式產生物件，並取得正確 output	1.Coverage({“037k4ry”,”37k4ry”,”7k4ry”}, 2.2)
				2.CoverageLongs({108, 598, 140}, 3, 1.7976931348623157E308)
					1. 2.2
					2.1.7976931348623157E308

2-4	Coverage	getHashes()	利用不同建構式產生物件，並取得正確 output	1.Coverage({“037k4ry”,”37k4ry”,”7k4ry”}, 2.2)
				2.CoverageLongs({108, 598, 140}, 3, 1.7976931348623157E308)
					1.” [37k4ry, 037k4ry, 7k4ry]”
					2.”[000000,000000000006,000000000008]”

3-1	Coverage Longs	getHashes()	依照建構式所傳入值，取得預期 output	1.CoverageLongs({108,598,140}, 3, 2.2)	1.{108,598,140}
3-2	Coverage Longs	getRatio()	依照建構式所傳入值，取得預期 output	1.CoverageLongs({108,598,140}, 3, 2.2)	1. 2.2

3-3	Coverage Longs	getHashLength()	依照建構式所傳入值並特別建立 Count 為 0 的物件確保額外處理，取得預期 output	1.CoverageLongs({108,598,140}, 3, 2.2)
				2.CoverageLongs({108,598,140}, 0, 2.2)
					1. 12
					2. 0

3-4	Coverage Longs	toString()	依照建構式所傳入值，取得預期 output	1.CoverageLongs({108,598,140}, 3, 2.2)	1. "Coverage [hashes=" & " ratio=2.2]"
3-5	Coverage Longs	getCount()	依照建構式所傳入值，取得預期 output	1.CoverageLongs({108,598,140}, 3, 2.2)	1. 3

4-1	Direction	Opposite()	經過執行 method 後，取得預期 output	1.Direction.RIGHT.opposite()
				2.Direction.LEFT.opposite()
					1.Direction.LEFT
					2.Direction.RIGHT
					3.Direction.TO

				3.Direction.BOTTOM.opposite()
				4.Direction.TOP.opposite()
					P
					4.Direction.BOTTOM

5-1	LanLong	getLat()	依照建構式所傳入值，取得預期 output	1.LatLong(1.7976931348623157E308, 4.9E-324)	1. 1.7976931348623157E308
5-2	LanLong	getLon()	依照建構式所傳入值，取得預期 output	1.LatLong(1.7976931348623157E308, 4.9E-324)	1. 4.9E-324
5-3	LanLong	add()	依照建構式所傳入值並執行 method 後，取得預期 output	1.LatLong(1.7976931348623157E308, 4.9E-324) & deltaLat=1 deltaLon=-1	1.LatLong(1.7976931348623157E308 + 1, 4.9E-324 - 1)
5-4	LanLong	toString()	依照建構式所傳入值，取得預期 output	1.LatLong(1.7976931348623157E308, 4.9E-324)	1."LatLong [lat=1.7976931348623157E308,lon=4.9E-324]"
6-1	GeoHash	fromLongToString()	傳入例外值，並取得預期 throw	1.hash=0	Expected = IllegalArgumentException.class
6-2	GeoHash	adjacentHash()	傳入例外值，並取得預期 throw，傳入正常值，取得預期 output	1.”STV” 2.”STV” 3.”STV” 4.”STV” 5.null 6.””	1.”stt” 2.”gzz” 3.”stu” 4.”bpb” 5. expected = IllegalArgumentException.class 6. expected = IllegalArgumentException.class

3 Test Implementation

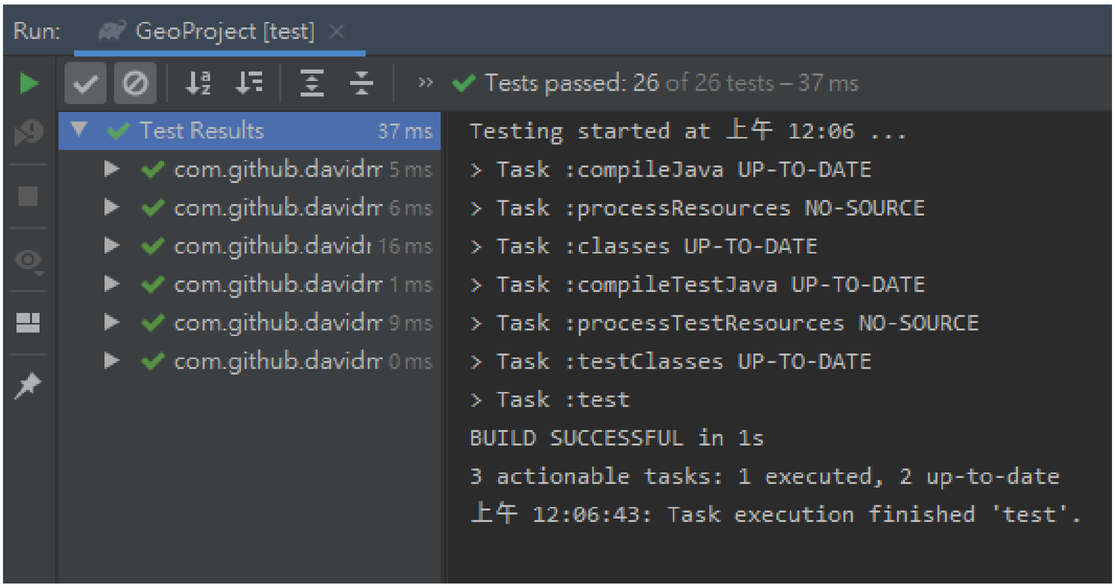
The design of test cases specified in Section 2 was implemented using JUnit 4. The test scripts of 26 selected test cases are given below. The rest of test script implementations can be found in the link (or JUnit files).

No.	Test method	Source code
1	encodeBase32WithOnlyNum()	Base32Test.java
2	encodeBase32WithPositiveN	Base32Test.java

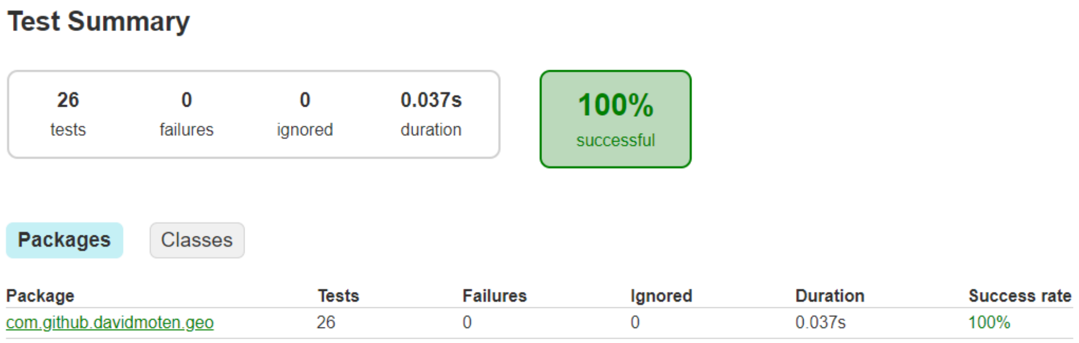
	umAndBiggerLength()	
3	encodeBase32WithNegativeNumAndNormalLength()	Base32Test.java
4	decodeBase32WithPositiveHash()	Base32Test.java
5	decodeBase32WithNegativeHash()	Base32Test.java
6	decodeBase32WithInvalidCharacter()	Base32Test.java
7	toStringTest()	CoverageTest.java
8	getHashLengthTest()	CoverageTest.java
9	getHashLengthTestWithZeroSize()	CoverageTest.java
10	getRatioTest()	CoverageTest.java
11	getHashesTest()	CoverageTest.java
12	getCountTest()	CoverageLongsTest.java
13	toStringTest()	CoverageLongsTest.java
14	getHashLengthTest()	CoverageLongsTest.java
15	getHashLengthTestWithZeroSize()	CoverageLongsTest.java
16	getRatioTest()	CoverageLongsTest.java
17	getHashesTest()	CoverageLongsTest.java
18	opposite()	DirectionTest.java
19	getLat()	LatLongTest.java
20	getLon()	LatLongTest.java
21	add()	LatLongTest.java
22	testToString()	LatLongTest.java
23	fromLongToStringWithInvalidLong()	GeoHashTest.java
24	adjacentHashWithNullHash()	GeoHashTest.java
25	adjacentHashWithEmptyHash()	GeoHashTest.java
26	testAdjacentWithDifferentDirection()	GeoHashTest.java

4 Test Results

4.1 JUnit test result snapshot



```
Testing started at 上午 12:06 ...
> Task :compileJava UP-TO-DATE
> Task :processResources NO-SOURCE
> Task :classes UP-TO-DATE
> Task :compileTestJava UP-TO-DATE
> Task :processTestResources NO-SOURCE
> Task :testClasses UP-TO-DATE
> Task :test
BUILD SUCCESSFUL in 1s
3 actionable tasks: 1 executed, 2 up-to-date
上午 12:06:43: Task execution finished 'test'.
```



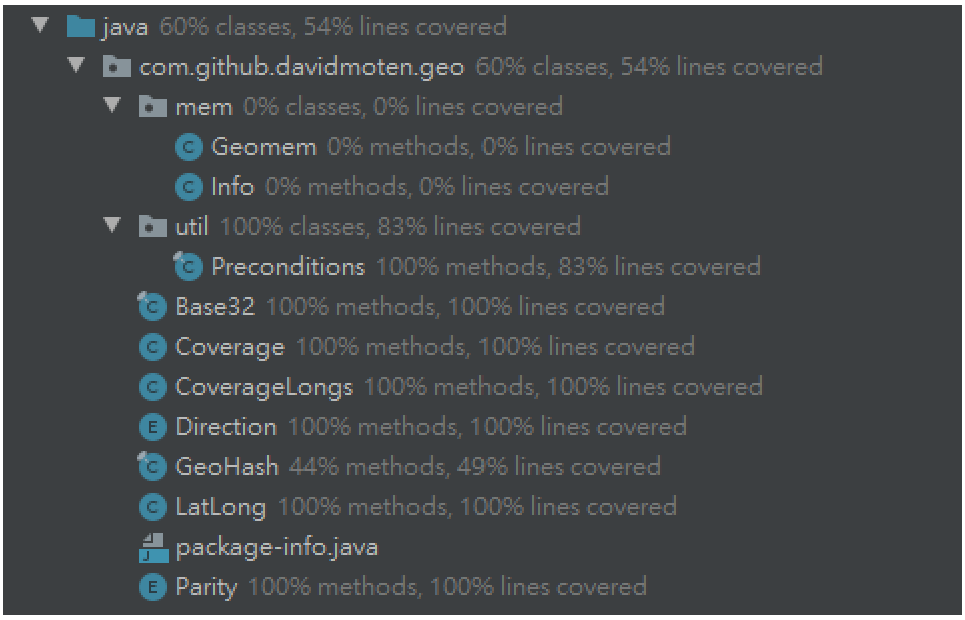
Test Summary

26	0	0	0.037s	100%
tests	failures	ignored	duration	successful

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	26	0	0	0.037s	100%

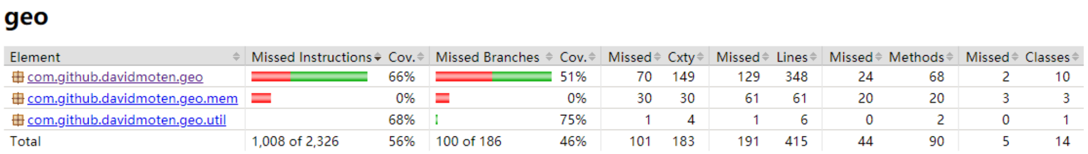
4.2 Code coverage snapshot

- Coverage of each selected method



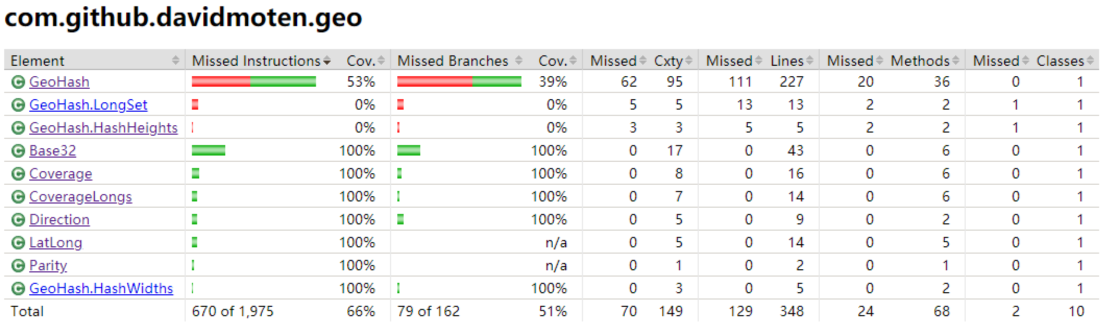
```
java 60% classes, 54% lines covered
  com.github.davidmoten.geo 60% classes, 54% lines covered
    mem 0% classes, 0% lines covered
      Geomem 0% methods, 0% lines covered
      Info 0% methods, 0% lines covered
    util 100% classes, 83% lines covered
      Preconditions 100% methods, 83% lines covered
    Base32 100% methods, 100% lines covered
    Coverage 100% methods, 100% lines covered
    CoverageLongs 100% methods, 100% lines covered
    Direction 100% methods, 100% lines covered
    GeoHash 44% methods, 49% lines covered
    LatLong 100% methods, 100% lines covered
    package-info.java
    Parity 100% methods, 100% lines covered
```

- Total coverage



geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.github.davidmoten.geo	🟥🟩	66%	🟥🟩	51%	70	149	129	348	24	68	2	10
com.github.davidmoten.geo.mem	🟥	0%	🟥	0%	30	30	61	61	20	20	3	3
com.github.davidmoten.geo.util		68%	🟩	75%	1	4	1	6	0	2	0	1
Total	1,008 of 2,326	56%	100 of 186	46%	101	183	191	415	44	90	5	14

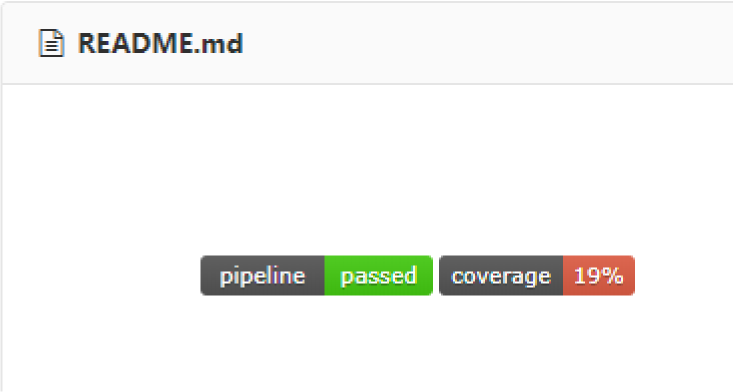


com.github.davidmoten.geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
GeoHash	🟥🟩	53%	🟥🟩	39%	62	95	111	227	20	36	0	1
GeoHash.LongSet	🟥	0%	🟥	0%	5	5	13	13	2	2	1	1
GeoHash.HashHeights	🟥	0%	🟥	0%	3	3	5	5	2	2	1	1
Base32	🟩	100%	🟩	100%	0	17	0	43	0	6	0	1
Coverage	🟩	100%	🟩	100%	0	8	0	16	0	6	0	1
CoverageLongs	🟩	100%	🟩	100%	0	7	0	14	0	6	0	1
Direction	🟩	100%	🟩	100%	0	5	0	9	0	2	0	1
LatLong	🟩	100%		n/a	0	5	0	14	0	5	0	1
Parity	🟩	100%		n/a	0	1	0	2	0	1	0	1
GeoHash.HashWidths	🟩	100%	🟩	100%	0	3	0	5	0	2	0	1
Total	670 of 1,975	66%	79 of 162	51%	70	149	129	348	24	68	2	10

4.3 CI result snapshot (4 iterations for CI)

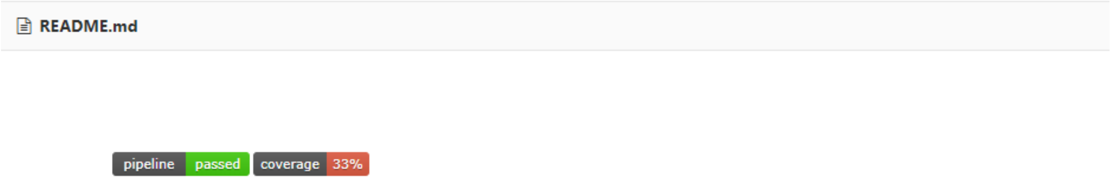
- CI#1



README.md

pipeline passed coverage 19%

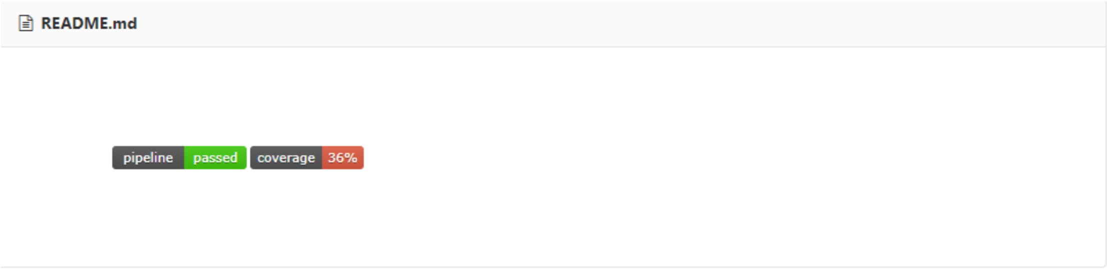
- CI#2



README.md

pipeline passed coverage 33%

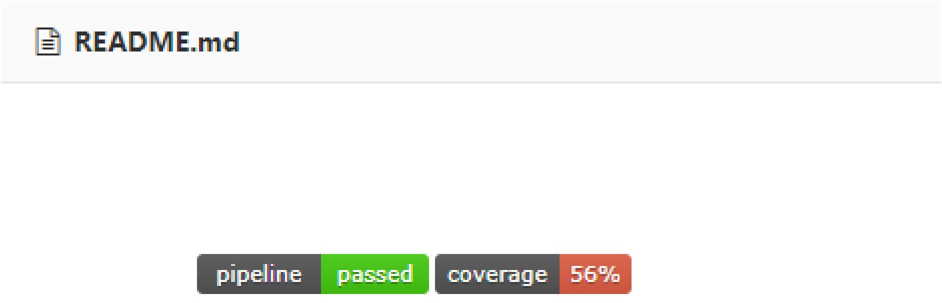
- CI#3



README.md

pipeline passed coverage 36%

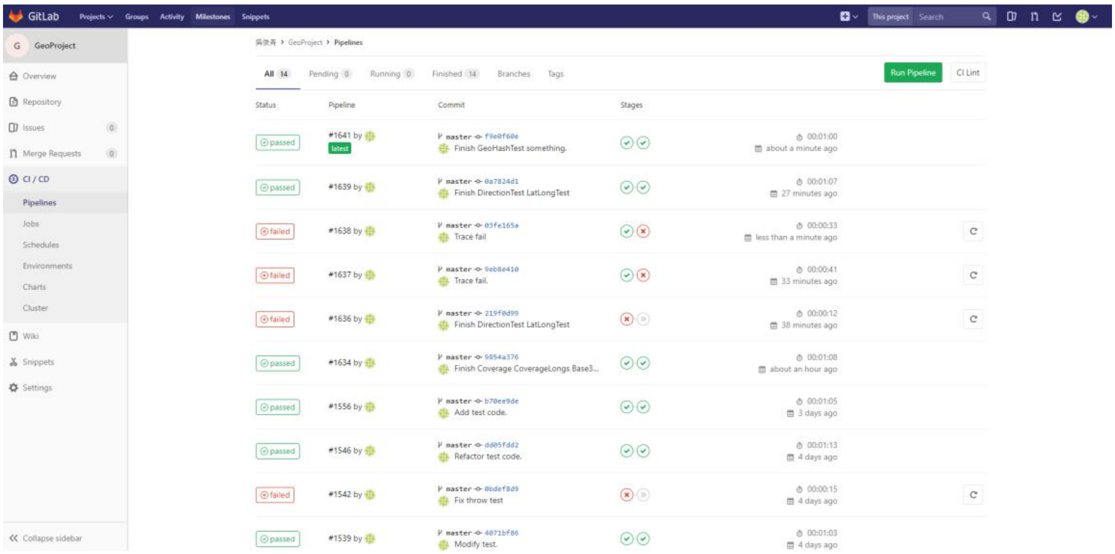
- CI#4



README.md

pipeline passed coverage 56%

- CI Pipeline



Status	Pipeline	Commit	Stages		
passed	#1641 by latest	master f9e0f60e Finish GeoHashTest something.	✓ ✓	00:01:00 about a minute ago	
passed	#1639 by	master 0a7824d1 Finish DirectionTest LatLongTest	✓ ✓	00:01:07 27 minutes ago	
failed	#1638 by	master 03fe165a Trace fail	✓ ✗	00:00:33 less than a minute ago	↻
failed	#1637 by	master 9eb8e410 Trace fail.	✓ ✗	00:00:41 33 minutes ago	↻
failed	#1636 by	master 219f0d99 Finish DirectionTest LatLongTest	✗ »	00:00:12 38 minutes ago	↻
passed	#1634 by	master 9854a376 Finish Coverage CoverageLongs Base3...	✓ ✓	00:01:08 about an hour ago	
passed	#1556 by	master b70ee9de Add test code.	✓ ✓	00:01:05 3 days ago	
passed	#1546 by	master dd05fdd2 Refactor test code.	✓ ✓	00:01:13 4 days ago	
failed	#1542 by	master 0bdef8d9 Fix throw test	✗ »	00:00:15 4 days ago	↻
passed	#1539 by	master 4071bf86 Modify test.	✓ ✓	00:01:03 4 days ago	

5 Summary

In Lab 1, **26 test cases have been designed and implemented using JUnit**. The test is conducted in 4 CI and **the execution results of the 22 test methods are all passed**. **The total statement coverage of the test is 56%.** Thus, the test requirements described in Section 1 are satisfied. Some lessons learned in this Lab are that Geo project is what and how to use intellij IDEA. I also learned how to design the test cases and implemented the test cases, but I need to continue learning how to design test cases reliably.