Lab 3 Report

吳俊青
108598014
2020/05/23

1 Test Plan

1.1 Test requirements

The Lab 3 requires to (1) select 6 methods from 6 classes of the SUT (GeoProject), (2) design Unit test cases by using **basis path or graph coverage** technique for the selected methods, (3) develop test scripts to implement the test cases, (4) execute the test scripts on the selected methods, (5) report the test results, and (6) specify your experiences of designing test cases systematically using the graph coverage technique.

In particular, based on the target coverage criteria (i.e., statement, branch, or others), the **test requirements** for Lab 3 are to design test cases *with* **graph coverage technique** for each selected method so that "*each statement and branch (or path) of the method under test will be covered by at least one test case* and *the both minimum* ***statement*** *(node) and* ***branch*** *(edge) coverage are greater than those of Lab 2 and 90%, respectively*."

1.2 Test Strategy

To satisfy the test requirements listed in Section 1, a proposed strategy is to

(1) select **3 methods that were chosen in Lab1 or Lab2** and **3 new methods** that are NOT selected previously. The selected methods MUST contain **predicate** and/or **loop** structures (as many as possible).

(2) set the objective of the minimum statement or branch (or path) coverage to be greater than that of Lab 2 and adjust the test objective (e.g., 90%, 95% or 100%) based on the time available (if necessary).

(3) design the test cases for those selected methods by using the **basis path or graph coverage** testing technique.

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	1	2020/05/14
2	Learn **basis path and**	3	2020/05/04~2020/05/06

	graph coverage		
3	Design test cases for the selected methods	10	2020/05/14~2020/05/15
4	Implement test cases	8	2020/05/16~2020/05/17
5	Perform tests and check code coverage. If not satisfy, design more test cases…	1	2020/05/16~2020/05/17
6	Complete Lab3 report	4	2020/05/22~2020/05/25

1.4 Design Approach

The **basis path and graph coverage** technique will be used to design the test cases. Specifically, the control flow graph (CFG) of each selected method shall be drawn first, and the possible test paths that satisfy the test requirements (i.e., **statement (node), branch (edge), or path coverage**) shall be derived from the CFG. The possible **inputs** and **expected outputs** for the derived test paths shall be computed from the specification of SUT for each method under test. *Add more test cases by considering to satisfy other coverage criteria, such as edge-pair, all-use, or prime-path coverage criteria*.

1.5 Success criteria

All test cases designed for the selected methods must pass (or 90% of all test cases must pass) and *both statement and branch (or path) coverage should have achieved at least 90%, respectively*.

2 Test Design

To fulfill the test requirements listed in section 1.1, the following methods are selected and corresponding test cases are designed. 紅色 method 為新增的 method，並且因版面關係，每一個 method 只列出第一個 test case 的資訊，Test Paths 紅色則為 Infeasible path。

No	Class	Method	Source Code Links	CFG Links	Test Paths	Inputs	Expected Outputs
1	Geomem	find()	Source Code	請參考 Excel File	P1: {n1,n2,n3,n5} P2: {n1,n2,n3,n4,n3,n5}	topLeftLat = -5 topLeftLon=100 bottomRightLat=-45 bottomRightLon=170 start=0 finish=1000	符合條件的 info，並且為所預期的資料
2	Geomem	addToMap()	Source Code		P1: {n1,n2,n8} P2: {n1,n2,n3,n4,n5,n7,n2,n8} P3: {n1,n2,n3,n4,n5,n6,n7,n2,n8}	map=Maps.newConcurrentMap() info=new Info<String, String>(-15, 120, 500, "A1", Optional.of("A")) hash="qtm6dtm6dtm6"	Add 後透過 find()找出資料確定有確實 add 資料進入 Map
3	GeoHash	gridAsString()	Source Code		P1: {n1,n2,n3,n13} P2: {n1,n2,n3,n4,n5,n6,n12,n3,n13} P3: {n1,n2,n3,n4,n5,n6,n7,n8,n9,n11,n6,n12,n3,n13} P4: {n1,n2,n3,n4,n5,n6,n7,n8,n9,n10,n11,n6,n12,n3,n13}	hash="dred" fromRight=-5 fromBottom=50 toRight=5 toBottom=5 highlightThese=Collections.<String>emptySet()	""
4	Base32	encodeBase32()	Source Code		P1: {n1,n2,n4,n6,n7,n8} P2: {n1,n2,n4,n6,n7,n9} P3: {n1,n2,n4,n5,n4,n6,n7,n8} P4: {n1,n2,n4,n5,n4,n6,n7,n9} P5: {n1,n2,n3,n4,n5,n4,n6,n7,n8} P6:	i=-1 length=5	"-00001"

				請參考 Excel File	{n1,n2,n3,n4,n5,n4,n6,n7,n9} P7: {n1,n2,n3,n4,n6,n7,n8} P8: {n1,n2,n3,n4,n6,n7,n9}		
5	GeoHash	heightDegrees()	Source Code		P1: {n1,n2} P2: {n1,n3}	n=24	1.561251128379126 4E-16
6	Base32	decodeBase32()	Source Code		P1: {n1,n2,n3,n6,n8} P2: {n1,n2,n3,n6,n7,n8} P3: {n1,n2,n3,n4,n5,n3,n6,n8} P4: {n1,n2,n3,n4,n5,n3,n6,n7,n8}	hash=""	0

The details of the design are given below:

The Excel file of test case:

108598014 Lab3 (Graph Coverage test case design).xlsx

此檔與 Lab3Report 在同一資料夾中

3 Test Implementation

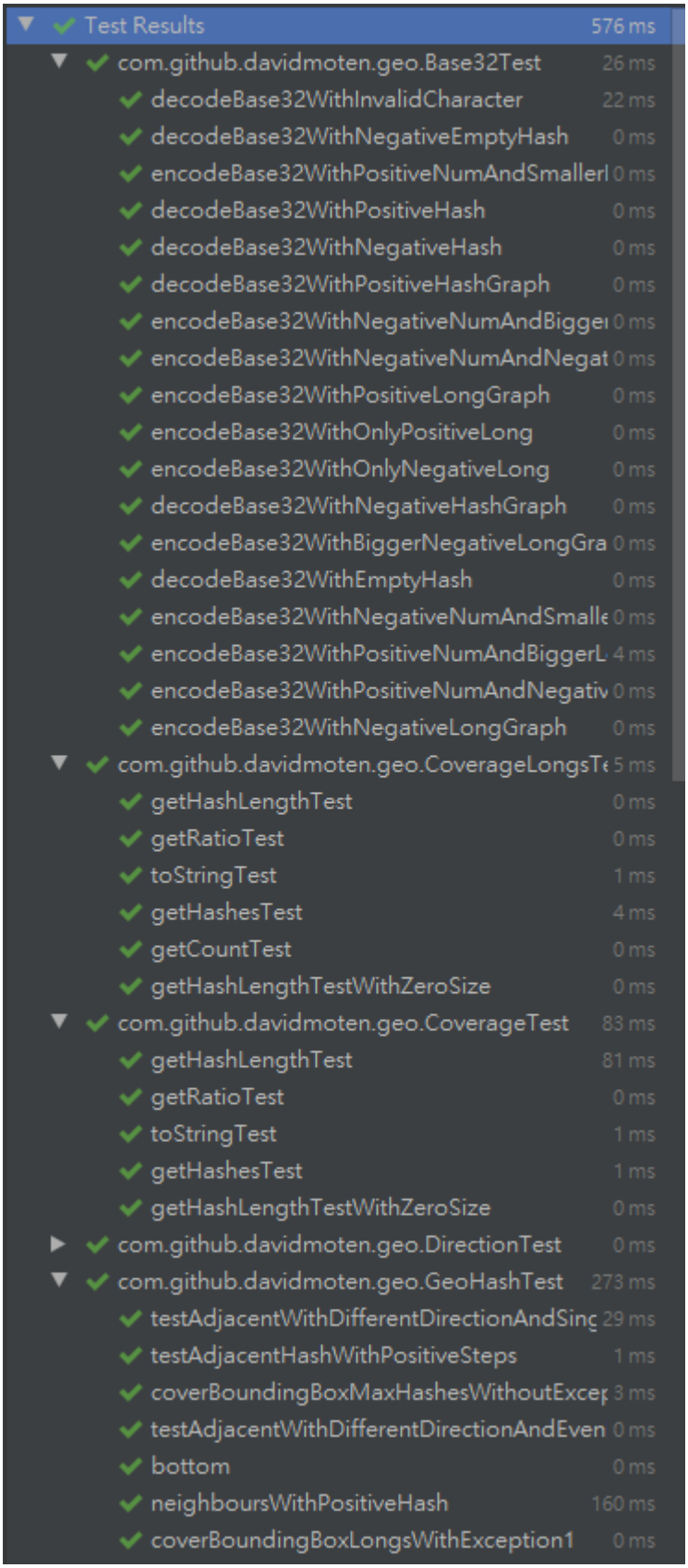
The design of test cases specified in Section 2 was implemented using JUnit 4. The test scripts of 3 selected test cases are given below. The rest of the test script implementations can be found in the below link.

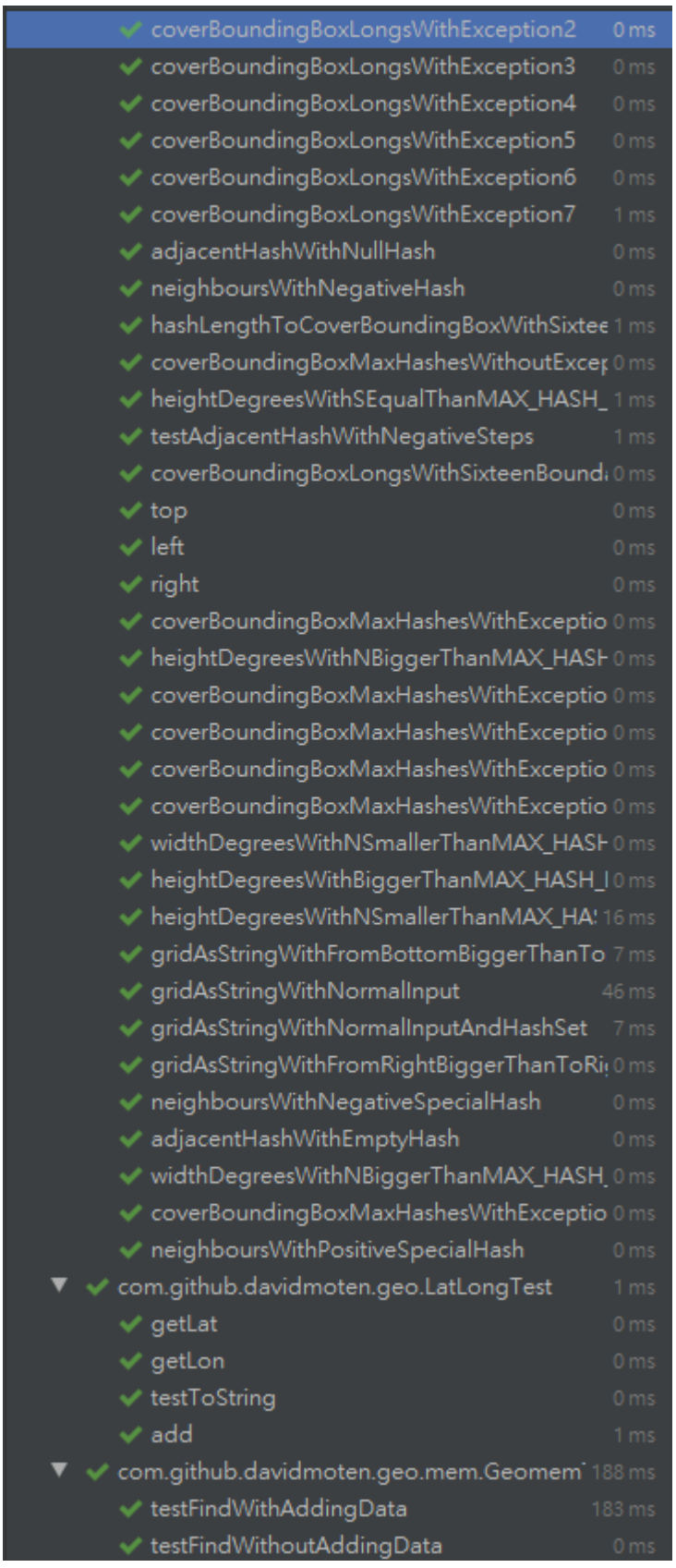
https://stv.csie.ntut.edu.tw/108598014/GeoProject

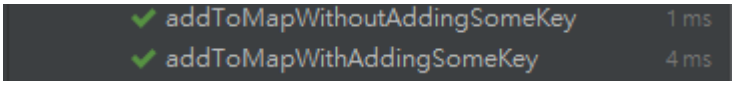
No.	Test method	Source test code
1	testFindWithAddingData()	GeomemTest.java
2	gridAsStringWithNormalInput()	GeoHashTest.java
3	decodeBase32WithNegativeEmptyHash()	Base32Test.java

4 Test Results

4.1 JUnit test result snapshot




coverBoundingBoxLongsWithException2 0 ms
coverBoundingBoxLongsWithException3 0 ms
coverBoundingBoxLongsWithException4 0 ms
coverBoundingBoxLongsWithException5 0 ms
coverBoundingBoxLongsWithException6 0 ms
coverBoundingBoxLongsWithException7 1 ms
adjacentHashWithNullHash 0 ms
neighboursWithNegativeHash 0 ms
hashLengthToCoverBoundingBoxWithSixtee 1 ms
coverBoundingBoxMaxHashesWithoutExcep 0 ms
heightDegreesWithSEqualThanMAX_HASH_ 1 ms
testAdjacentHashWithNegativeSteps 1 ms
coverBoundingBoxLongsWithSixteenBoundi 0 ms
top 0 ms
left 0 ms
right 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
heightDegreesWithNBiggerThanMAX_HASH 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
widthDegreesWithNSmallerThanMAX_HASH 0 ms
heightDegreesWithBiggerThanMAX_HASH_I 0 ms
heightDegreesWithNSmallerThanMAX_HA 16 ms
gridAsStringWithFromBottomBiggerThanTo 7 ms
gridAsStringWithNormalInput 46 ms
gridAsStringWithNormalInputAndHashSet 7 ms
gridAsStringWithFromRightBiggerThanToRi 0 ms
neighboursWithNegativeSpecialHash 0 ms
adjacentHashWithEmptyHash 0 ms
widthDegreesWithNBiggerThanMAX_HASH_ 0 ms
coverBoundingBoxMaxHashesWithExceptio 0 ms
neighboursWithPositiveSpecialHash 0 ms
com.github.davidmoten.geo.LatLongTest 1 ms
getLat 0 ms
getLon 0 ms
testToString 0 ms
add 1 ms
com.github.davidmoten.geo.mem.Geomem 188 ms
testFindWithAddingData 183 ms
testFindWithoutAddingData 0 ms


addToMapWithoutAddingSomeKey 1 ms
addToMapWithAddingSomeKey 4 ms

Test Summary

79	0	0	0.515s
tests	failures	ignored	duration

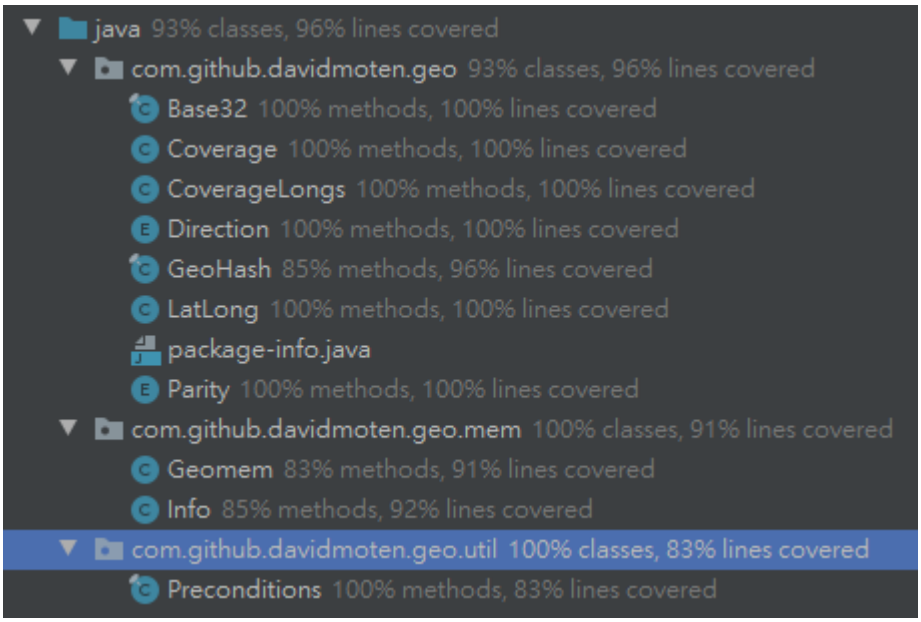
100% successful

Packages Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	75	0	0	0.398s	100%
com.github.davidmoten.geo.mem	4	0	0	0.117s	100%

4.2 Code coverage snapshot

- Coverage of each selected method under test



- Total coverage

geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.github.davidmoten.geo		95%		93%	14	149	10	348	6	68	0	10
com.github.davidmoten.geo.mem		84%		85%	6	30	5	61	3	20	0	3
com.github.davidmoten.geo.util		68%		75%	1	4	1	6	0	2	0	1
Total	145 of 2,326	93%	14 of 186	92%	21	183	16	415	9	90	0	14

com.github.davidmoten.geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
GeoHash		92%		91%	14	95	10	227	6	36	0	1
Base32		100%		100%	0	17	0	43	0	6	0	1
Coverage		100%		100%	0	8	0	16	0	6	0	1
GeoHash.LongSet		100%		100%	0	5	0	13	0	2	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
GeoHash.HashHeights		100%		100%	0	3	0	5	0	2	0	1
Total	89 of 1,975	95%	10 of 162	93%	14	149	10	348	6	68	0	10

com.github.davidmoten.geo.mem

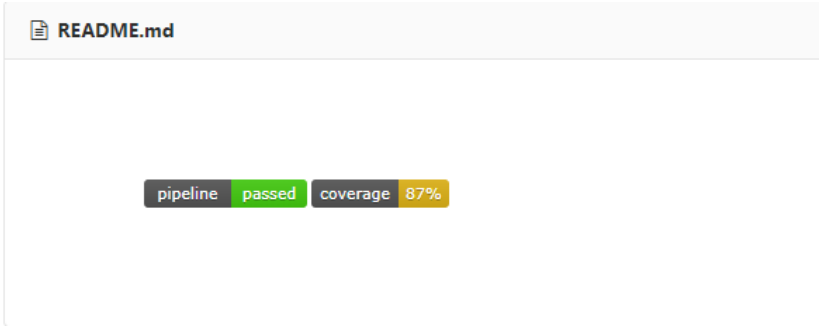
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Info		50%		n/a	1	7	1	13	1	7	0	1
Geomem		91%		100%	2	17	4	46	2	11	0	1
Geomem.new Predicate() {...}		100%		62%	3	6	0	3	0	2	0	1
Total	50 of 332	84%	3 of 20	85%	6	30	5	61	3	20	0	3

com.github.davidmoten.geo.util

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Preconditions		68%		75%	1	4	1	6	0	2	0	1
Total	6 of 19	68%	1 of 4	75%	1	4	1	6	0	2	0	1

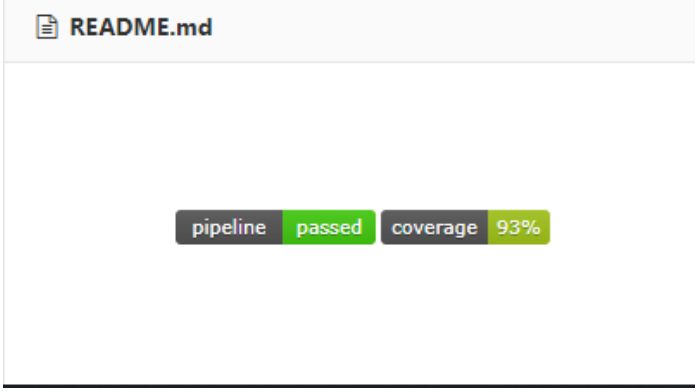
4.3 CI result snapshot (3 iterations for CI)

- CI#1

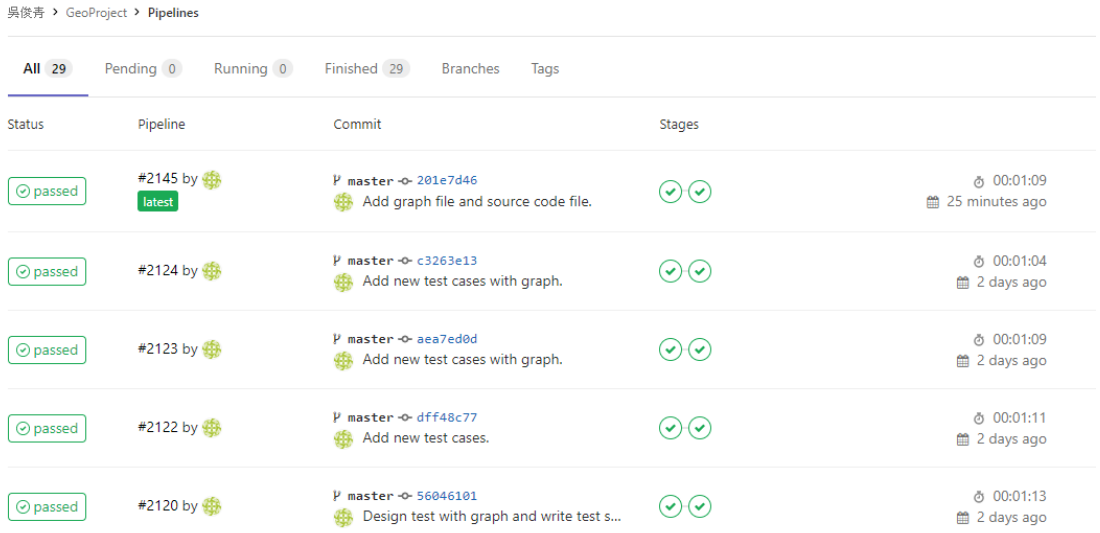


- CI#2

- CI#3



- CI Pipeline



5 The Coverage Comparison

The code coverage of Lab1 and Lab3 are listed in the below Table.

The results show that the statement coverage is increased from 56% to 93% in Lab3.

The results show that the branch coverage is increased from 46% to 92% in Lab3.

No.	Test method	Lab1		Lab3	
		statement coverage	branch coverage	statement coverage	branch coverage
1	fromLongToString(long)	100	83	76	66
2	encodeHash(double, double, int)	93	50	93	50
3	coverBoundingBoxLongs(double, double, double, double, int)	0	0	99	92
4	adjacentHashAtBorder(String, Direction)	84	71	100	100
5	decodeHash(String)	100	100	100	100
6	hashLengthToCoverBoundingBox(double, double, double, double)	0	0	100	100
7	encodeHashToLong(double, double, int)	97	90	100	100
8	adjacentHash(String, Direction)	92	75	100	100
9	neighbours(String)	0	0	100	100
10	addOddParityEntries(Map)	100	100	100	100
11	gridAsString(String, int, int, int, int, Set)	0	0	100	100
12	coverBoundingBoxMaxHashes(double, double, double, double,	0	0	100	100

	int)				
13	createBorders()	100	100	100	100
14	createNeighbours()	100	100	100	100
15	refineInterval(double[], int, int)	60	50	100	100
16	createDirectionParityMap()	100	100	100	100
17	adjacentHash(String, Direction, int)	0	0	100	100
18	calculateHeightDegrees(int)	0	0	100	100
19	calculateWidthDegrees(int)	100	100	100	100
20	heightDegrees(int)	0	0	100	100
21	widthDegrees(int)	70	50	100	100
22	checkHash(String)	100	100	100	100
23	coverBoundingBox(double, double, double, double)	0	0	100	100
24	encodeHash(double, double)	0	0	100	100
25	newHashMap()	100	100	100	100
26	right(String)	0	0	100	100
27	left(String)	0	0	100	100
28	top(String)	0	0	100	100
29	bottom(String)	0	0	100	100
30	**GeoHash(Total)**	53	39	92	91
31	encodeBase32(long, int)	100	100	100	100
32	decodeBase32(String)	100	100	100	100
33	padLeftWithZerosToLength(String, int)	100	100	100	100
34	getCharIndex(char)	100	100	100	100
35	encodeBase32(long)	100	100	100	100
36	**Base32(Total)**	100	100	100	100
37	Coverage(CoverageLongs)	100	100	100	100
38	toString()	100	100	100	100
39	getHashLength()	100	100	100	100
40	Coverage(Set, double)	100	100	100	100
41	getHashes()	100	100	100	100
42	getRatio()	100	100	100	100
43	**Coverage(Total)**	100	100	100	100
44	add(long)	0	0	100	100
45	GeoHash.LongSet()	0	0	100	100
46	**GeoHash.LongSet(Total)**	0	0	100	100
47	toString()	100	100	100	100
48	getHashes()	100	100	100	100

49	getHashLength()	100	100	100	100
50	CoverageLongs(long[], int, double)	100	100	100	100
51	getRatio()	100	100	100	100
52	getCount()	100	100	100	100
53	**CoverageLongs(Total)**	100	100	100	100
54	opposite()	100	100	100	100
55	**Direction(Total)**	100	100	100	100
56	toString()	100	100	100	100
57	add(double, double)	100	100	100	100
58	LatLong(double, double)	100	100	100	100
59	getLat()	100	100	100	100
60	getLon()	100	100	100	100
61	**LatLong(Total)**	100	100	100	100
62	createValues()	100	100	100	100
63	**GeoHash.HashWidths(Total)**	100	100	100	100
64	createValues()	0	0	100	100
65	**GeoHash.HashHeights(Total)**	0	0	100	100
com.github.davidmoten.geo		66	51	95	93

No.	**Test method**	**Lab1**		**Lab3**	
		statement coverage	**branch coverage**	**statement coverage**	**branch coverage**
1	Info(double, double, long, Object, Optional)	0	0	100	100
2	id()	0	0	100	100
3	lat()	0	0	100	100
4	lon()	0	0	100	100
5	time()	0	0	100	100
6	value()	0	0	100	100
7	**Info(Total)**	0	0	100	100
8	addToMap(Map, Info, String)	0	0	100	100
9	addToMapById(Map, Info, String)	0	0	100	100
10	find(double, double, double, double, long, long)	0	0	100	100
11	find(long, long, String)	0	0	100	100
12	add(Info)	0	0	100	100
13	find(double, double, double, double, long, long, String)	0	0	100	100

14	add(double, double, long, Object, Optional)	0	0	100	100
15	Geomem()	0	0	100	100
16	createRegionFilter(double, double, double, double)	0	0	100	100
17	**Geomem(Total)**	0	0	100	100
18	apply(Info)	0	0	100	62
19	**Geomem.new Predicate(Total)**	0	0	100	62
com.github.davidmoten.geo.mem		0	0	84	85

No.	Test method	Lab1		Lab3	
		statement coverage	branch coverage	statement coverage	branch coverage
1	checkNotNull(Object, Object)	40	50	40	50
2	checkArgument(boolean, Object)	100	100	100	100
3	**Preconditions(Total)**	68	75	68	75
com.github.davidmoten.geo.util		68	75	68	75

	Lab1		Lab3	
	statement coverage	branch coverage	statement coverage	branch coverage
Project Total	56	46	93	92

6 Summary

In Lab 3, **6 test cases have been designed and implemented using JUnit and the basis path/graph coverage technique**. The test is conducted in 3 CI and **the execution results of the 6 test methods are all passed**. **The total statement and branch coverage of the test are 93% and 92%, respectively.** Thus, the test requirements described in Section 1 are satisfied.

在這次的 Lab 中，我挑選出三個已經在之前 Lab 中的 method，以及三個新的 method，依照 basis path 對這六個 method 做測試案例的設計，在上課所學到的技術，我將它利用在本次 Lab 中，讓我更加地了解 Graph 的用法，依照上述的原則，也使最後測試覆蓋率達到了 93%，Branch Coverage 也達到了 92%，在這次的 Lab 中讓我充分練習了所學到的不同設計方法，使我收益良多。

在比較兩次 Lab 中的 Coverage 時，我也了解到依靠經驗去做測試，布見得可以將測試品質做到最好，但是如果依照著不同的設計方法，或許會多

花點時間，但是相反的卻可以得到很好的測試品質，我覺得這樣反而是更好的。