

Lab 3 Report

Name

Student ID

Date May 27, 2020

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	May 17, 2020
2	Learn basis path and	2	May 18, 2020

	graph coverage		
3	Select Suitable Methods of Classes	2	May 19, 2020
4	Draw Graphs and Verify Testability	10	May 20-23, 2020
5	Arrange Test Paths	1	May 24, 2020
6	Design test cases for the selected methods	2	May 25, 2020
7	Implement test cases	3	May 26, 2020
8	Perform tests and check code coverage	1	May 27, 2020
9	If not satisfy, design more test cases	2	May 27, 2020
10	Complete Lab3 report	2	May 27, 2020

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.

No.	Class	Method	Source Code Links	CFG Links	Test Paths	Inputs	Expected Outputs
1	Base32	encodeBase32(long i, int length)	https://stv.csi.e.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com	https://stv.csie.ntut.edu.tw/rojeanlin/GeoProject/blob/master	P1:{1, 2,3,4, 3,5,6} P2:{1, 2,3,4, 3,5,7}	T1: {inputs:(i=75324), (length=4); expected:29jw} T2: {inputs:(i=-122), (length=4); expected:-003u} T3: {inputs:(i=5), (length=0); expected:5}	

			/github/davidmoten/geo/Base32Test.java	r/LabRepository/Lab3/Graphs.pdf	P3:{1, 2,3,5, 6} P4:{1, 2,3,5, 7} P5:{1, 3,4,3, 5,6} P6:{1, 3,4,3, 5,7} P7:{1, 3,5,6} P8:{1, 3,5,7}	T4: {inputs:(i=-5), (length=0); expected:-5}
		decodeBase32(String hash)	https://stvie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java	https://stvie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java	P1:{1, 2,4,5, 7,4,6, 9} P2:{1, 2,4,5, 7,4,6, 8,9} P3:{1, 2,4,6, 9} P4:{1, 2,4,6, 8,9} P5:{1, 3,4,5, 7,4,6, 9} P6:{1, 3,4,5, 7,4,6,	T1:{input:(hash="w");expected:28} T2:{input:(hash="-j");expected:-17} T3:{input:(hash="-");expected:-0} T4:{input:(hash="");expected:0}

					8,9} P7:{1, 3,4,6, 9} P8:{1, 3,4,6, 8,9}	
2	GeoHash	adjacentHashAtBorder(String hash, Direction direction)	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/LabReport/Lab3/Graphs.pdf	P1:{1, 2,4} P2:{1, 2,12} P3:{1, 3,5,7} P4:{1, 3,5,12} P5:{1, 3,6,8, 10} P6:{1, 3,6,8, 12} P7:{1, 3,6,9, 11} P8:{1, 3,6,9, 12}	T1:{inputs:(hash="000"),(direction=Direction.BOTTOM);expected:"h00"} T2:{inputs:(hash="zzz"),(direction=Direction.TOP);expected:"gzz"} T3:{inputs:(hash="000"),(direction=Direction.LEFT);expected:"pdp"} T4:{inputs:(hash="rfr"),(direction=Direction.RIGHT);expected:"242"} T5:{inputs:(hash="11w"),(direction=Direction.TOP);expected:"11y"} T6:{inputs:(hash="11w"),(direction=Direction.BOTTOM);expected:"11q"} T7:{inputs:(hash="11w"),(direction=Direction.LEFT);expected:"11t"} T8:{inputs:(hash="11w"),(direction=Direction.RIGHT);expected:"11x"}
		fromLongToString(long hash)	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master	P1:{1, 2,3} P2:{1, 2,4,5, 6,8,5, 7}	T1:{input:(hash=0);expected:throw "invalid long geohash 0"} T2:{input:(hash=-8845069668155654141);expected:"hp0"}

			/github/davidmoten/geo/GeoHashTest.java	r/LabRepository/Lab3/Graphs.pdf	P3:{1,2,4,5,7}	T3:{ <u>input</u> :(hash=1); <u>expected</u> :"0"}
		encodeHashToLong(double latitude, double longitude, int length)	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java	P1:{1,2,4} P2:{1,2,3,5,7,11,2,4} P3:{1,2,3,5,8,11,2,4} P4:{1,2,3,6,9,11,2,4} P5:{1,2,3,6,10,11,2,4}	T1:{ <u>inputs</u> :(latitude=0),(longitude=0),(length=0); <u>expected</u> :"0"} T2:{ <u>inputs</u> :(latitude=55),(longitude=55),(length=5); <u>expected</u> :"-2846829668114366459L"} T3:{ <u>inputs</u> :(latitude=55),(longitude=-55),(length=5); <u>expected</u> :"8274428630898049029L"}
		hashLengthToCoverBoundingBox(double topLeftLat, double topLeftLon, double bottomRightLat, double bottomRightLon)	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java	https://stvc.sie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java	P1:{1,2,3,5,7,9} P2:{1,2,3,5,7,10,1,9,20,2,...} P3:{1,2,3,5,8,11} P4:{1,2,3,5,8,12,1,9,20,2}	T1:{ <u>inputs</u> :(topLeftLat=25),(topLeftLon=25),(bottomRightLat=25),(bottomRightLon=25); <u>expected</u> :12} T2:{ <u>inputs</u> :(topLeftLat=25),(topLeftLon=-25),(bottomRightLat=25),(bottomRightLon=25); <u>expected</u> :0} T3:{ <u>inputs</u> :(topLeftLat=25),(topLeftLon=25),(bottomRightLat=25),(bottomRightLon=-25); <u>expected</u> :0} T4:{ <u>inputs</u> :(topLeftLat=25),(topLeftLon=-25),(bottomRightLat=25),(bottomRightLon=25); <u>expected</u> :0}

					<pre> ····} P5:{1, 2,3,6, 13,15} P6:{1, 2,3,6, 13,16, 19,,20 ,2,···· .} P7:{1, 2,3,6, 14,17} P8:{1, 2,3,6, 14,18, 19,20, 2,···· } P9:{1, 2,4} </pre>	25),(bottomRightLat=25),(bottomRightLon=-25); <u>expected</u> :12}
--	--	--	--	--	--	--

The details of the design are given below:

The Excel file of test cases...

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 [link](#) (or JUnit files).**

No.	Test method	Source test code
1	adjacentHashAtBorder(String hash, Direction direction)	https://stv.csie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java

2	hashLengthToCover erBoundingBox(double topLeftLat, double topLeftLon, double bottomRightLat, double bottomRightLon)	https://stv.csie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java
3	encodeBase32(long i, int length)	https://stv.csie.ntut.edu.tw/rojeanlin/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java

4 Test Results

4.1 JUnit test result snapshot

▼ ✓ Test Results	604 ms
▶ ✓ com.github.davidmoten.geo.CoverageLongsTest	12 ms
▶ ✓ com.github.davidmoten.geo.LatLongTest	3 ms
▶ ✓ com.github.davidmoten.geo.Base32Test	27 ms
▶ ✓ com.github.davidmoten.geo.CoverageTest	39 ms
▶ ✓ com.github.davidmoten.geo.GeoHashTest	63 ms
▶ ✓ com.github.davidmoten.geo.DirectionTest	1 ms
▶ ✓ com.github.davidmoten.geo.mem.GeomemTest	435 ms
▶ ✓ com.github.davidmoten.geo.mem.InfoTest	24 ms

Test Summary

57

0

0

0.604s

tests

failures

ignored

duration

100%

successful

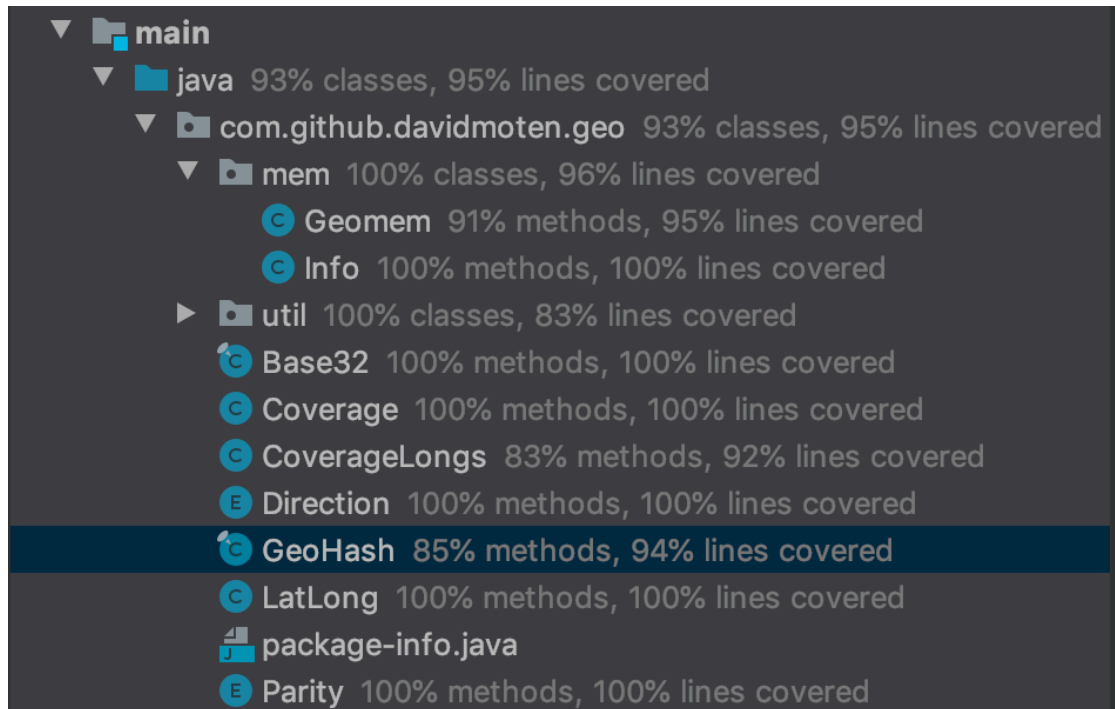
Packages

Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	45	0	0	0.145s	100%
com.github.davidmoten.geo.mem	12	0	0	0.459s	100%

4.2 Code coverage snapshot

- Coverage of each selected method under test

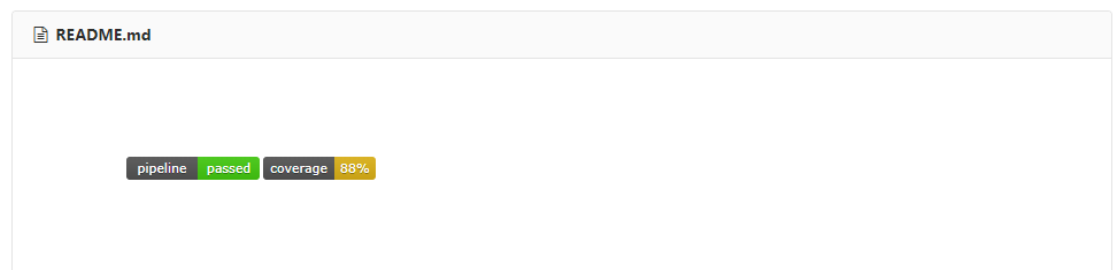


- Total coverage

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
com.github.davidmoten.geo	<div><div></div></div>	94%	<div><div></div></div>	90%	20 149	15 348	7 68	0 10
com.github.davidmoten.geo.mem	<div><div></div></div>	97%	<div><div></div></div>	85%	4 30	2 61	1 20	0 3
com.github.davidmoten.geo.util	<div><div></div></div>	68%	<div><div></div></div>	75%	1 4	1 6	0 2	0 1
Total	129 of 2,326	94%	19 of 186	89%	25 183	18 415	8 90	0 14

4.3 CI result snapshot (3 iterations for CI)

- CI#1



- CI#2

passed	#5269	master -> c276f0d7	#2243 by	test	test	00:33 about 2 hours ago	90.0%	
failed	#5268	master -> c276f0d7	#2243 by	test	test	00:12 about 2 hours ago		
passed	#5267	master -> c276f0d7	#2243 by	build	build	00:25 about 2 hours ago		

● CI#3

README.md





















pipeline

passed

coverage

94%

- CI Pipeline(由於網站數次無法連上，故雖分數次 commit 仍一次 push 較大量更動)

Status	Pipeline	Commit	Stages	
	#2271 by  latest	master -> 7faf34d8  add Graphs.pdf, test case desi...	 	⌚ 00:01:04 📅 37 minutes ago
	#2243 by 	master -> c276f0d7  add Base32Test:encodeBase3...	 	⌚ 00:00:59 📅 about 2 hours ago
	#2056 by 	master -> ba607be1  adjust Lab2 report format	 	⌚ 00:01:07 📅 a month ago
	#2055 by 	master -> d66ea085  add Lab2 report	 	⌚ 00:01:15 📅 a month ago

5 The Coverage Comparison

The code coverage of Lab1 (and/or Lab2) and Lab3 are listed in the below Table. The results show that the statement and branch coverage are increased from 88% to 94% in Lab3.

No.	Test method	Lab1 (or Lab2)		Lab3	
		statement coverage	branch coverage	statement coverage	branch coverage
1	adjacentHashAtBorder(String hash, Direction direction)	20%		100%	
2	hashLengthToCoverBoundingBox(double topLeftLat, double topLeftLon, double bottomRightLat, double bottomRightLon)	0%		100%	
3	encodeBase32(long i, int length)	50%		100%	

6 Summary

In Lab 3, **26** 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 26 test methods are **all passed**. The total statement and branch coverage of the test are **95%** and **100%**, respectively. Thus, the test requirements described in Section 1 are satisfied. **Some lessons learned in this Lab are Graph Coverage Techniques**