Lab 3 Report

Name: 許哲維
Student ID: 111598066
Date: 2023.05.10

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 75%, 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	2	2023.04.20
2	Learn **basis path and**	4.5	2023.04.26

	graph coverage		
3	Design test cases for the selected methods	6	2023.04.30
4	Implement test cases	2.5	2023.05.06
5	Perform tests and check code coverage. If not satisfy, design more test cases…	3	2023.05.09
6	Complete Lab3 report	2	2023.05.10

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 80% of all test cases must pass) and *both statement and branch (or path) coverage should have achieved at least 80%, 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://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_encodeBase3	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_encodeBase3	**P1**: {n1, n3, n5, n6}, **P2**: {n1, n2, n3, n5, n7}, **P3**: {n1, n3, n4, n3, n5, n6}, **P4**: {n1, n2, n3, n4, n3, n5, n7}	T1:{i=75324, length=4}, T2:{i=-75324, length=4}, T3:{i=16, length=1}, T4:{i=-16, length=1	T1:{"29jw"}, T2:{"-29jw"}, T3:{"h"}, T4:{"-h"}

			2.jpg	2.jpg		}	
2	Base32	decodeBase32(String hash)	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_decodeBase32.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_decodeBase32.jpg	P1:{n1,n2,n4,n5,n7,n9}, P2:{n1,n3,n4,n5,n6,n5,n7,n8,n9}	T1:{hash=""}, T2:{hash="-1"}	T1:{0}, T2:{-1}
3	Base32	getCharIndex(char ch)	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_getCharIndex.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_getCharIndex.jpg	P1:{n1,n2,n3}, P2:{n1,n2,n4}	T1:{ch='a'}, T2:{ch='0'}	T1:{"not a base32 character: a"}, T2:{0}
4	Base32	padLeftWithZerosToLength(String s, int length)	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_padLeftWithZerosToLength.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Base32_padLeftWithZerosToLength.jpg	P1:{n1,n2,n3,n4,n3,n6}, P2:{n1,n6}	T1:{s="29jw",length=4}, T2:{s="29jw",length=5}	T1:{"29jw"}, T2:{"029jw"}
5	Direction	opposite()	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabR	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabR	P1: {n1, n2}, P2: {n1, n3, n4}, P3: {n1, n3, n5, n6}, P4: {n1, n3, n5, n7}	T1:{Direction.RIGHT}, T2:{Direction.LEFT}, T3:{Direction.BOTTOM}, T4:{Dire	T1:{Direction.LEFT}, T2:{Direction.RIGHT}, T3:{Direction.TOP}, T4:{Direction.BOTTOM}

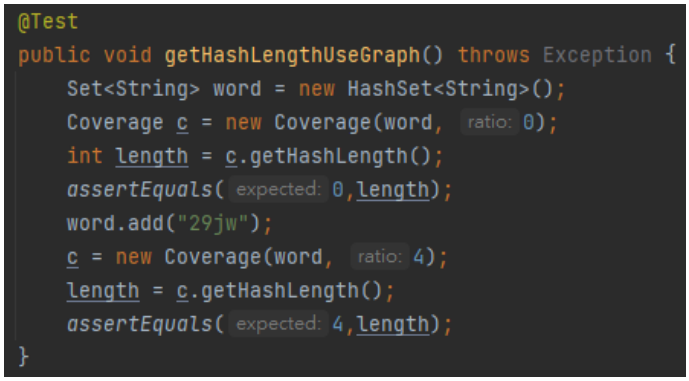
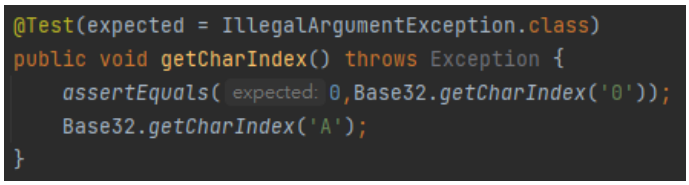
			eport/Lab3/Direction_opposite.jpg	eport/Lab3/Direction_opposite.jpg		ction.TOP}	
6	Coverage	getHashLength()	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Coverage_getHashLength.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/Coverage_getHashLength.jpg	P1:{n1,n2}, P2:{n1,n3}	T1:{hashes=""} T2:{hashes="29jw"}	T1:{0} T2:{4}
7	GeoHash	fromLongToString(long hash)	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/GeoHash_fromLongToString.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/GeoHash_fromLongToString.jpg	P1:{n1, n2, n4}, P2:{n1, n2, n3, n4}, P3:{n1, n2, n3, n5, n6, n7, n6, n8}	T1:{hash=12}, T2:{hash=0}, T3:{hash=1}	T1:{"invalid long geohash 12"}, T2:{"invalid long geohash 0"}, T3:{"0"}
8	GeoHash	heightDegrees(int n)	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/GeoHash_heightDegrees.jpg	https://course.selab.ml/stv-gitlab/11598066/GeoProject/blob/master/LabReport/Lab3/GeoHash_heightDegrees.jpg	P1:{n1,n2}, P2:{n1,n3}	T1:{n=0)} T2:{hashes=13}	T1:{180} T2:{0}

The details of the design are given below:

Lab3 (Graph Coverage test case design).xlsx

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	opposite()	
2	heightDegrees(int n)	
3	getHashLength()	
4	padLeftWithZerosToLength(String s, int length)	
5	getCharIndex(char ch)	

4 Test Results

4.1 JUnit test result snapshot

Test Summary

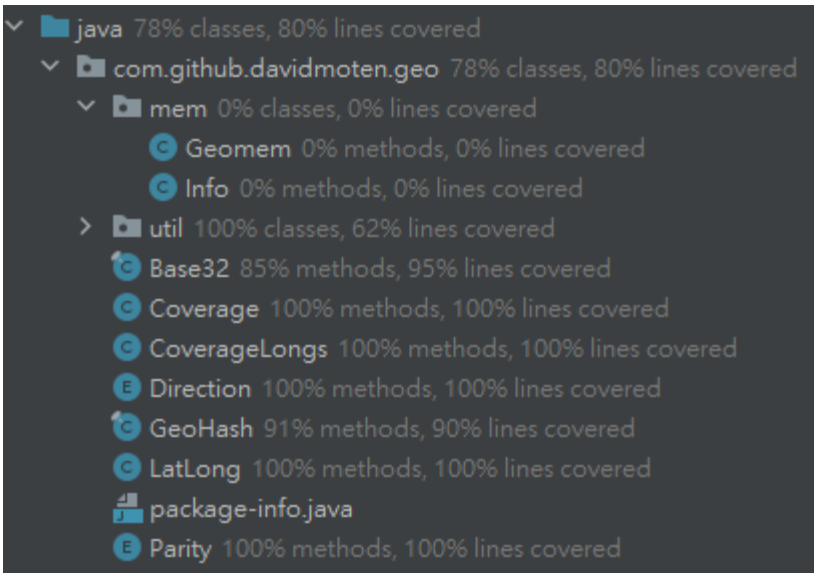
92 tests	0 failures	0 ignored	0.095s duration	100% successful

Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	92	0	0	0.095s	100%

4.2 Code coverage snapshot

- Coverage of each selected method under test



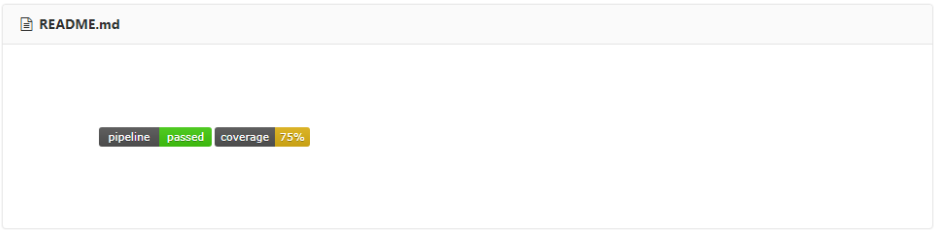
- Total coverage

geo

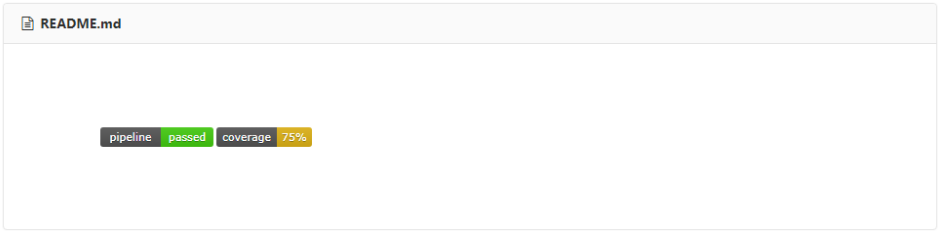
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.github.davidmoten.geo		94%		84%	25	149	17	348	1	68	0	10
com.github.davidmoten.geo.mem		0%		0%	30	30	61	61	20	20	3	3
com.github.davidmoten.geo.util		68%		75%	1	4	1	6	0	2	0	1
Total	447 of 2,326	80%	46 of 186	75%	56	183	79	415	21	90	3	14

4.3 CI result snapshot (3 iterations for CI)

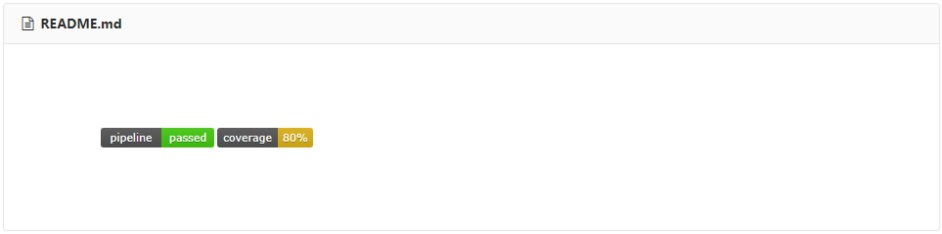
- CI#1



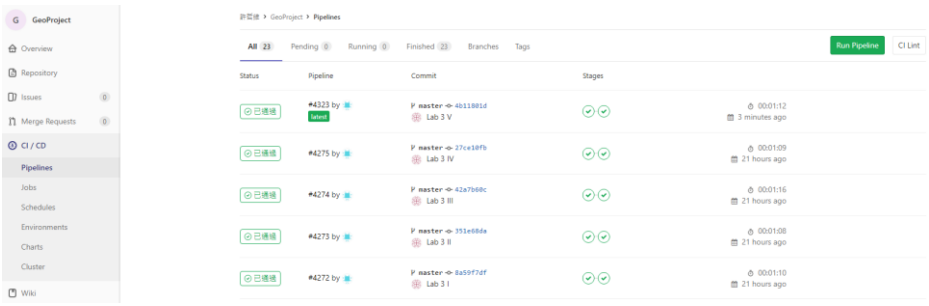
- CI#2



- CI#3



- CI Pipeline



5 The Coverage Comparison

The code coverage of Lab1 (and/or Lab2) and Lab3 are listed in the below Table. The results show that the statement and branch coverage are increased from 80% to 100% in Lab3.

No.	Test method	Lab1 (or Lab2)		Lab3	
		statement coverage	branch coverage	statement coverage	branch coverage
1	opposite()	100%	100%	100%	100%
2	heightDegrees(int n)	80%	81%	100%	100%
3	getHashLength()	100%	100%	100%	100%
4	padLeftWithZerosToLength(String s, int length)	100%	100%	100%	100%
5	getCharIndex(char ch)	100%	100%	100%	100%

6 Summary

In Lab 3, **8 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 94% and 84%, respectively.** Thus, the test requirements described in Section 1 are satisfied.

在 Lab 3 的 Graph Coverage 的練習，繪製出程式的路徑，來讓我們找到一些測試中可能被忽略的語句，並藉由此發現修正測試以提升測試覆蓋率讓測試能夠更完整，透過這次 Lab 令我收穫許多。