

# Lab 2 Report

Name : 劉宏德

Student ID : 108598004

Date : 2020/04/22

## 1 Test Plan

### 1.1 Test requirements

The Lab 2 requires to (1) select 19 methods from 6 classes of the SUT (GeoProject), (2) design Unit test cases by using **input space partitioning (ISP)** 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 ISP technique.

In particular, based on the statement coverage criterion, the **test requirements** for Lab 2 are to design test cases *with **ISP*** 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 70% (greater than Lab 1)*”.

### 1.2 Test Strategy

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

- (1) select **those 10 methods that were chosen in Lab1** and **5 new methods** that are NOT selected previously. If possible, some of the methods do NOT have primitive types of input or output parameters (if possible).
- (2) set the objective of the minimum statement coverage to be greater than that of Lab 1 and adjust the test objective based on the time available (if necessary).
- (3) design the test cases for those selected methods by using the **input space partitioning (ISP)** 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	5	2020/04/15
2	Learn <b>ISP</b> and JUnit	4	2020/04/15~04/18
3	Design test cases for the selected methods	8	2020/04/16~04/18
4	Implement test cases	8	2020/04/17~04/18

5	Perform tests	4	2020/04/18
6	Complete Lab2 report	2	2020/04/22

#### 1.4 Design Approach

The **ISP** technique will be used to design the test cases. Specifically, the possible partitions and boundary values of input parameters shall be identified first using the **Mine Map** and **domain knowledge** (if applicable). The possible **valid combinations of the partitions** (i.e., **all combination coverage**) as well as the boundary values shall be computed for the input parameters of each selected method. Each of the partition combination can be a possible test case. *Add more test cases by considering the possible values and boundary of the outputs for the methods or by using test experiences.*

#### 1.5 Success criteria

All test cases designed for the selected methods must pass and *the statement coverage should have achieved at least 70%.*

## 2 Test Design

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

N o.	Class	Method	Test Object ive	Inputs	Expected Outputs
1	Base32	encodeBase32(long l, int length)	將 long 值以 Base32 進行編碼, 答案長度以 length 控制	(-10, 5)	-0000b
2	Base32	encodeBase32(long i)	將 long 值以 Base32 進行編碼	-10	- 00000000 000b
3	Base32	decodeBase32(String hash)	將 hash 值解碼為 long 值	-29jw	-75324

4	Base32	getCharIndex(char ch)	取得輸入字元為base32索引中第幾個	b	10
5	Base32	padLeftWithZerosToLength(String s, int length)	將s左邊補零到指定長度	1234, 6	001234
6	CoverageLongs	getHashes()	取得hashes	CoverageLongs({5, 9, 1}, count=3, ratio=1.2)	{5, 9, 1}
7	CoverageLongs	getRatio()	取得指定區域中hash所占區域比例	CoverageLongs({5, 9, 1}, count=3, ratio=1.2)	1.2
8	CoverageLongs	getHashLength()	取得在hash set中第一個字元代表的長度	CoverageLongs({5, 9, 1}, count=1, ratio=1.2)	5
9	CoverageLongs	getCount()	取得count	CoverageLongs({5, 9, 1}, count=1, ratio=1.2)	1
10	GeoHash	adjacentHash(String hash, Direction direction)	取得指定方向的hash的鄰居	dg, right	e5
11	GeoHash	right(String hash)	取得hash的右鄰居	dg	E5
12	GeoHash	left(String hash)	取得hash的左鄰居	d4	9f
13	GeoHash	top(String hash)	取得hash的上	dx	f8

			鄰居		
14	GeoHash	bottom(String hash)	取得hash下鄰居	d2	6r
15	GeoHash	adjacentHash(String hash, Direction direction, int steps)	取得hash的指定方向的steps格遠的鄰居	d2, bottom, 2	6q
16	GeoHash	neighbours(String hash)	取得hash附近的8個鄰居	d2	<"d0", "d8", "d3", "6r", "d1", "6p", "d9", "6x">
17	GeoHash	encodeHash(double latitude, double longitude)	利用經緯度轉換出長度為12的GeoHash	50, 50	v0gs3y0zh 7w1
18	GeoHash	encodeHash(LatLong p, int length)	利用經緯度轉換出指定長度的GeoHash	50, 50, 12	v0gs3y0zh 7w1
19	GeoHash	encodeHash(LatLong p)	利用經緯度轉換出長度為12的GeoHash	LatLong(50, 50)	v0gs3y0zh 7w1

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).

N o.	Test method	Source code
1	testEncodeBase32WithISP_2()	<pre>// test encodeBase32(long i) with ISP @Test public void testEncodeBase32WithISP_2() {     // i &lt;= -32     String encodeHash = Base32.encodeBase32(i: -33);     assertEquals( expected: "-000000000011", encodeHash);      // -32 &lt; i &lt; 32     encodeHash = Base32.encodeBase32(i: -10);     assertEquals( expected: "-00000000000b", encodeHash);      // i &gt;= 32     encodeHash = Base32.encodeBase32(i: 33);     assertEquals( expected: "000000000011", encodeHash); }</pre>
2	testGetCharIndexWithISP()	<pre>// test getCharIndex(char ch) with ISP // C1: the character in the characterIndexes map @Test public void testGetCharIndexWithISP() {     // C1: T     int ch_Index = Base32.getCharIndex('b');     assertEquals( expected: 10, ch_Index);      // C1: F     try {         ch_Index = Base32.getCharIndex('*');         fail("no exception");     } catch (Exception message) {         assertTrue(message.getMessage().contains("not a base32 character: *"));     } }</pre>
3	testPadLeftWithZerosToLengthWithISP()	<pre>// test padLeftWithZerosToLength(String s, int length) with ISP // C1: s is null, C2: s.length &gt;= length @Test public void testPadLeftWithZerosToLengthWithISP() {     // C1: T, C2: T or F     expectedEx.expect(NullPointerException.class);     String s = Base32.padLeftWithZerosToLength(s: null, length: 8);      // C1: F, C2: T     s = Base32.padLeftWithZerosToLength(s: "1234", length: 2);     assertEquals( expected: "1234", s);      // C1: F, C2: F     s = Base32.padLeftWithZerosToLength(s: "1234", length: 6);     assertEquals( expected: "001234", s); }</pre>

## 4 Test Results

### 4.1 JUnit test result snapshot

Test Results	399 ms
▶ ✓ com.github.davidmoten.geo.Base32Test	36 ms
▶ ✓ com.github.davidmoten.geo.CoverageLongsTest	7 ms
▶ ✓ com.github.davidmoten.geo.CoverageTest	26 ms
▶ ✓ com.github.davidmoten.geo.GeoHashTest	197 ms
▶ ✓ com.github.davidmoten.geo.mem.GeomemTest	132 ms
▶ ✓ com.github.davidmoten.geo.mem.InfoTest	1 ms

## Test Summary

<b>57</b>	<b>0</b>	<b>0</b>	<b>0.272s</b>
tests	failures	ignored	duration

**100%**  
successful

Packages Classes

Package	Tests	Failures	Ignored	Duration	Success rate
<a href="#">com.github.davidmoten.geo</a>	50	0	0	0.166s	100%
<a href="#">com.github.davidmoten.geo.mem</a>	7	0	0	0.106s	100%

## 4.2 Code coverage snapshot

- Coverage of each selected method

Project
<ul style="list-style-type: none"> <li>src           <ul style="list-style-type: none"> <li>main               <ul style="list-style-type: none"> <li>java 93% classes, 95% lines covered                   <ul style="list-style-type: none"> <li>com.github.davidmoten.geo 93% classes, 95% lines covered                       <ul style="list-style-type: none"> <li>mem 100% classes, 100% lines covered</li> <li>util 100% classes, 83% lines covered                           <ul style="list-style-type: none"> <li>Base32 100% methods, 100% lines covered</li> <li>Coverage 100% methods, 100% lines covered</li> <li>CoverageLongs 83% methods, 92% lines covered</li> <li>Direction 100% methods, 100% lines covered</li> <li>GeoHash 100% methods, 96% lines covered</li> <li>LatLong 60% methods, 42% lines covered</li> <li>package-info.java</li> <li>Parity 100% methods, 100% lines covered</li> </ul> </li> </ul> </li> </ul> </li> </ul> </li> </ul> </li></ul>

- Total coverage  
geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
<a href="#">com.github.davidmoten.geo</a>	<div><div></div></div>	94%	<div><div></div></div>	90%	19 149	18 348	3 68	0 10
<a href="#">com.github.davidmoten.geo.util</a>	<div><div></div></div>	68%	<div><div></div></div>	75%	1 4	1 6	0 2	0 1
<a href="#">com.github.davidmoten.geo.mem</a>	<div><div></div></div>	99%	<div><div></div></div>	75%	5 30	0 61	0 20	0 3
Total	118 of 2,326	94%	22 of 186	88%	25 183	19 415	3 90	0 14

### 4.3 CI result snapshot (3 iterations for CI)

- CI#1

<div>passed</div>	#4155 P master -> efca8c1e	#1803 by	test	test	⌚ 00:35 📅 3 days ago	94.0%	<div>C</div>
-------------------	----------------------------	----------	------	------	-------------------------	-------	--------------

- CI#2

<div>passed</div>	#4166 P master -> 90435c5a	#1806 by	test	test	⌚ 00:32 📅 3 days ago	94.0%	<div>C</div>
-------------------	----------------------------	----------	------	------	-------------------------	-------	--------------

- CI#3

<div>passed</div>	#4173 P master -> eda9e301	#1809 by	test	test	⌚ 00:35 📅 2 days ago	94.0%	<div>C</div>
-------------------	----------------------------	----------	------	------	-------------------------	-------	--------------

- CI Pipeline

<div>passed</div>	#1809 by latest	P master -> eda9e301 test with ISP in CoverageLongsTest co...	✓✓	⌚ 00:01:07 📅 2 days ago
<div>passed</div>	#1806 by	P master -> 90435c5a test with ISP in Base32Test complete	✓✓	⌚ 00:01:04 📅 3 days ago
<div>passed</div>	#1803 by	P master -> efca8c1e add test with ISP in Base32Test initial	✓✓	⌚ 00:01:09 📅 3 days ago

## 5 Summary

In Lab 2, **19 test cases** have been designed and implemented using JUnit and the ISP technique. The test is conducted in **3 CI** and the execution results of the **15 test methods** are **all passed**. The total statement coverage of the test is **70%**. Thus, the test requirements described in Section 1 are satisfied.