

# Lab1

## 1 Test plan

### 1.1 Test requirements

(1)Select 24 methods from 6 classes of SUT (GeoProject).(2)Design Unit test cases (3)Develop test scripts to implement the test cases(4)Execute test script on the selected methods(5)Report results

### 1.2 Strategy

(1)Select that are easy to understand and have primitive types of input and output parameters.

(2) Learn necessary skills and tools.

(3)Set the objective of the minimum statement coverage to be 50% initially.

(4)Design the test cases for those selected methods.

1.use valid values and combinations of the input parameters

2.use boundary values of the input data.

### 1.3 Activites

Activities	Hours	Date
Study GeoProject	3	3/1/2023
Learn JUnit	4	3/3/2023
Design test case	1	3/4/2023
Implement test case	3	3/4/2023
Complete Report	2	3/5/2023

### 1.4 Success criteria

All test cases designed for the selected methods must pass and the statement coverage must be achieved at least 50%.

## 2 Test design

No	Class	Method	Test Objective	Input	Expected Outputs
1	Base32	encodeBase32()		75324,4	29jw
2	Base32	encodeBase32()		-75324,4	-29jw
3	Base32	encodeBase32()		75324,9	0000029jw
4	Base32	encodeBase32()		-75324,9	-0000029jw
5	Base32	encodeBase32()		10	00000000000b
6	Base32	decodeBase32()		29jw	75324

7	Base32	decodeBase32()		-29jw	-75324
8	GeoHash	right()		29jw	29jy
9	GeoHash	left()		29jw	29jq
10	GeoHash	top()		29jw	29jx
11	GeoHash	bottom()		29jw	29jt
12	GeoHash	adjacentHash()		29jw,Direction.TOP,1	29jx
13	GeoHash	adjacentHash()		29jw,Direction.LEFT,-1	29jy
14	GeoHash	neighbors()		29jw	<pre>"29jq", "29jy", "29jx", "29jt", "29jr", "29jm", "29jz", "29jv"</pre>
15	Coverage	Coverage()	CoverageLongs		CoverageLongs
16	Coverage	Coverage()	CoverageLongs		CoverageLongs
17	Coverage	getHashes()	Coverage	"29jw"	"29jw"
18	Coverage	getHashes()	Coverage	"-29jw"	"-29jw"
19	Coverage	getHashes()	Coverage	""	""
20	Coverage	getRatio()	Coverage	1.0	1.0
21	Coverage	getRatio()	Coverage	0.0	0.0
22	Coverage	getRatio()	Coverage	-1.0	-1.0
23	Coverage	getHashLength()	Coverage	"29jw"	4
24	Coverage	getHashLength()	Coverage	"-29jw"	5

25	Coverage	getHashLength()	Coverage	""	0
26	Coverage	ToString()	Coverage		<pre>"Coverage [hashes=" + "[29jw]" + ", ratio=" + 1.0 + "]"</pre>
27	Coverage	ToString()	Coverage		<pre>"Coverage [hashes=" + "[-29jw]" + ", ratio=" + 1.0 + "]"</pre>
28	Coverage	ToString()	Coverage		<pre>"Coverage [hashes=" + "[]" + ", ratio=" + 1.0 + "]"</pre>

29	CoverageLongs	getHashLength()	CoverageLongs		1
30	CoverageLongs	getHashLength()	CoverageLongs		0
31	CoverageLongs	getCount()	CoverageLongs		1
32	CoverageLongs	getCount()	CoverageLongs		0
33	CoverageLongs	getCount()	CoverageLongs		-1
34	CoverageLongs	testToString()	CoverageLongs		<pre>Coverage [hashes=[J@408 dd8eb, ratio=1.0]</pre>
35	CoverageLongs	testToString()	CoverageLongs		<pre>Coverage [hashes=[J@16a 311bd, ratio=1.0]</pre>
36	Direction	opposite()	Direction.TOP		Direction.BOTTOM
37	Direction	opposite()	Direction.BOTTOM		Direction.TOP

38	Direction	opposite()	Direction.RIGHT		Direction.LEFT
39	Direction	opposite()	Direction.LEFT		Direction.RIGHT
40	LatLong	getLat()	LatLong	1.0	1.0
41	LatLong	getLon()	LatLong	2.0	2.0
42	LatLong	add()	LatLong	1	
43	LatLong	add()	LatLong	1,1	2,3
44	LatLong	add()	LatLong	-1,-1	0,1
45	LatLong	add()	LatLong	0,0	1,2
46	LatLong	toString()	LatLong		"LatLong [lat=1.0, lon=2.0]"

### 3 Test Implementation

The rest of test scripts can be found in [link](#).

Test Methods	Source code
encodeBase32()	<pre> public void encodeBase32() throws Exception {     String encode = Base32.encodeBase32( i: 75324, length: 4);     assertEquals( expected: "29jw", encode);     encode = Base32.encodeBase32( i: -75324, length: 4);     assertEquals( expected: "-29jw", encode);     encode = Base32.encodeBase32( i: 75324, length: 9);     assertEquals( expected: "0000029jw", encode);     encode = Base32.encodeBase32( i: -75324, length: 9);     assertEquals( expected: "-0000029jw", encode);     encode = Base32.encodeBase32( i: 10);     assertEquals( expected: "0000000000b", encode); } </pre>
decodeBase32()	<pre> public void decodeBase32() {     long decode = Base32.decodeBase32( hash: "29jw");     assertEquals( expected: 75324, decode);     decode = Base32.decodeBase32( hash: "-29jw");     assertEquals( expected: -75324, decode); } </pre>

right()	<pre> @Test public void right() {     String rightstr = GeoHash.right( hash: "29jw");     assertEquals( expected: "29jy",rightstr); } </pre>
left()	<pre> no usages @Test public void left() {     String leftstr = GeoHash.left( hash: "29jw");     assertEquals( expected: "29jq",leftstr); } </pre>
bottom()	<pre> no usages @Test public void bottom() {     String bottomstr = GeoHash.bottom( hash: "29jw");     assertEquals( expected: "29jt",bottomstr); } </pre>
top()	<pre> @Test public void top() {     String topstr = GeoHash.top( hash: "29jw");     assertEquals( expected: "29jx",topstr); } </pre>

## 4 Test Result

### 4.1 JUnit test result snapshot

✓ <default package>	14 ms
> ✓ CoverageLongsTest	5 ms
> ✓ Base32Test	1 ms
> ✓ CoverageTest	2 ms
> ✓ DirectionTest	
> ✓ GeoHashTest	5 ms
✓ LatLongTest	1 ms
✓ getLat	
✓ getLon	
✓ testToString	1 ms
✓ add	

## Test Summary

2	0	0	0.005s
tests	failures	ignored	duration

**100%**  
successful

Packages

Classes

Package	Tests	Failures	Ignored	Duration	Success rate
<a href="#">com.github.davidmoten.geo</a>	2	0	0	0.005s	100%

## 4.2 Code coverage snapshot

▼	java	64% classes, 49% lines covered
▼	com.github.davidmoten.geo	64% classes, 49% lines covered
▼	mem	0% classes, 0% lines covered
	Geomem	0% methods, 0% lines covered
	Info	0% methods, 0% lines covered
>	util	100% classes, 50% lines covered
	Base32	85% methods, 93% lines covered
	Coverage	83% methods, 56% lines covered
	CoverageLongs	100% methods, 100% lines covered
	Direction	100% methods, 100% lines covered
	GeoHash	48% methods, 43% 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
<a href="#">com.github.davidmoten.geo.mem</a>	<div><div></div></div>	0%	<div><div></div></div>	0%	30 30	61 61	20 20	3 3
<a href="#">com.github.davidmoten.geo</a>	<div><div></div></div>	87%	<div><div></div></div>	75%	41 149	40 348	9 68	0 10
<a href="#">com.github.davidmoten.geo.util</a>	<div><div></div></div>	36%	<div><div></div></div>	50%	2 4	2 6	0 2	0 1
Total	596 of 2,326	74%	62 of 186	66%	73 183	103 415	29 90	3 14

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

CI#1:

📄 README.md

pipeline

passed

coverage

51%

📄 README.md

pipeline

passed

coverage

52%

📄 README.md

pipeline

passed

coverage

54%

📄 README.md

pipeline

passed

coverage

57%

📄 README.md

pipeline

passed

coverage

62%



#3761 by 🌿

🔗 3b411f3c  
second commit



⌚ 00:01:48  
📅 11 minutes ago




#3700 by 🌿

🔗 1ab875af  
🌿 Update README.md



📅 about 14 hours ago



#3762 by   
latest

1b8cbbaa  
third commit



00:01:56  
2 minutes ago

## 5. Summary

In Lab1, 46 test cases have been designed and implemented using Junit

The test is conducted in 3 CI and the execution results of 24 test methods are all passed. The total statement coverage of test is 54%.



