Lab1

1 Test plan

1.1 Test requirements

(1)Select 21 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

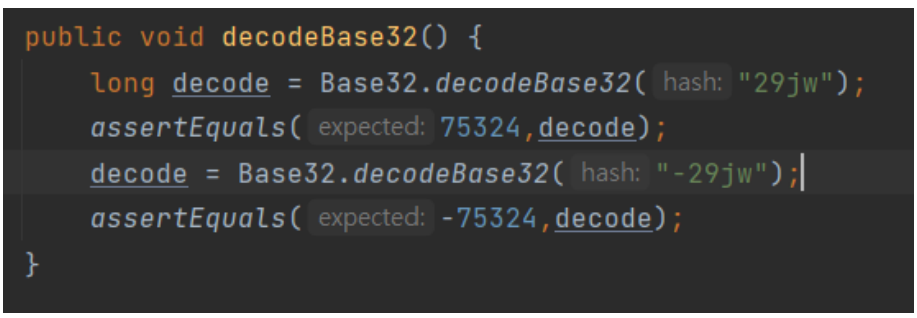
7	Base32	decodeBase32()		-29jw	-75324
8	GeoHash	right()		29jw	29jy
9	GeoHash	left()		29jw	29jq
10	GeoHash	top()		29jw	29jx
11	GeoHash	bottom()		29jw	29jt
12	GeoHash	adjacentHash()		29jw,Direction.TOP,1	29jx
13	GeoHash	adjacentHash()		29jw,Direction.LEFT,-1	29jy
14	GeoHash	neighbors()		29jw	"29jq", "29jy", "29jx", "29jt", "29jr", "29jm", "29jz", "29jv"
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		`"Coverage [hashes=" + "[29jw]" + ", ratio=" + 1.0 + "]"`
27	Coverage	ToString()	Coverage		`"Coverage [hashes=" + "[-29jw]" + ", ratio=" + 1.0 + "]"`
28	Coverage	ToString()	Coverage		`"Coverage [hashes=" + "[]" + ", ratio=" + 1.0 + "]"`

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		`Coverage [hashes=[J@408dd8eb, ratio=1.0]`
35	CoverageLongs	testToString()	CoverageLongs		`Coverage [hashes=[J@16a311bd, ratio=1.0]`
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()	
decodeBase32()	

```
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: "00000000000b",encode);

}
```

```
public void decodeBase32() {
    long decode = Base32.decodeBase32( hash: "29jw");
    assertEquals( expected: 75324,decode);
    decode = Base32.decodeBase32( hash: "-29jw");
    assertEquals( expected: -75324,decode);
}
```

right()	`@Test`
	`public void right() {`
	`  String rightstr = GeoHash.right( hash: "29jw");`
	`  assertEquals( expected: "29jy",rightstr);`
	`}`

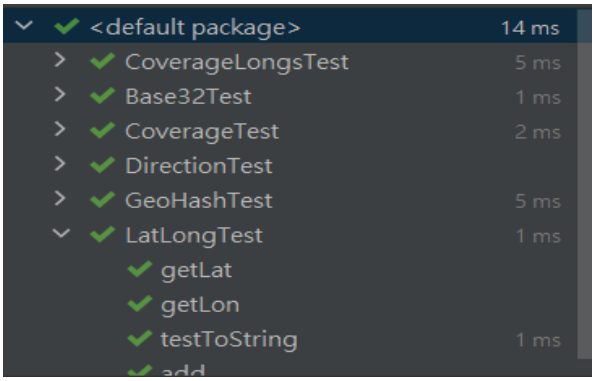
left()	no usages
	`@Test`
	`public void left() {`
	`  String leftstr = GeoHash.left( hash: "29jw");`
	`  assertEquals( expected: "29jq",leftstr);`
	`}`

bottom()	no usages
	`@Test`
	`public void bottom() {`
	`  String bottomstr = GeoHash.bottom( hash: "29jw");`
	`  assertEquals( expected: "29jt",bottomstr);`
	`}`

top()	`@Test`
	`public void top() {`
	`  String topstr = GeoHash.top( hash: "29jw");`
	`  assertEquals( expected: "29jx",topstr);`
	`}`

4 Test Result

4.1 JUnit test result snapshot



Test Summary

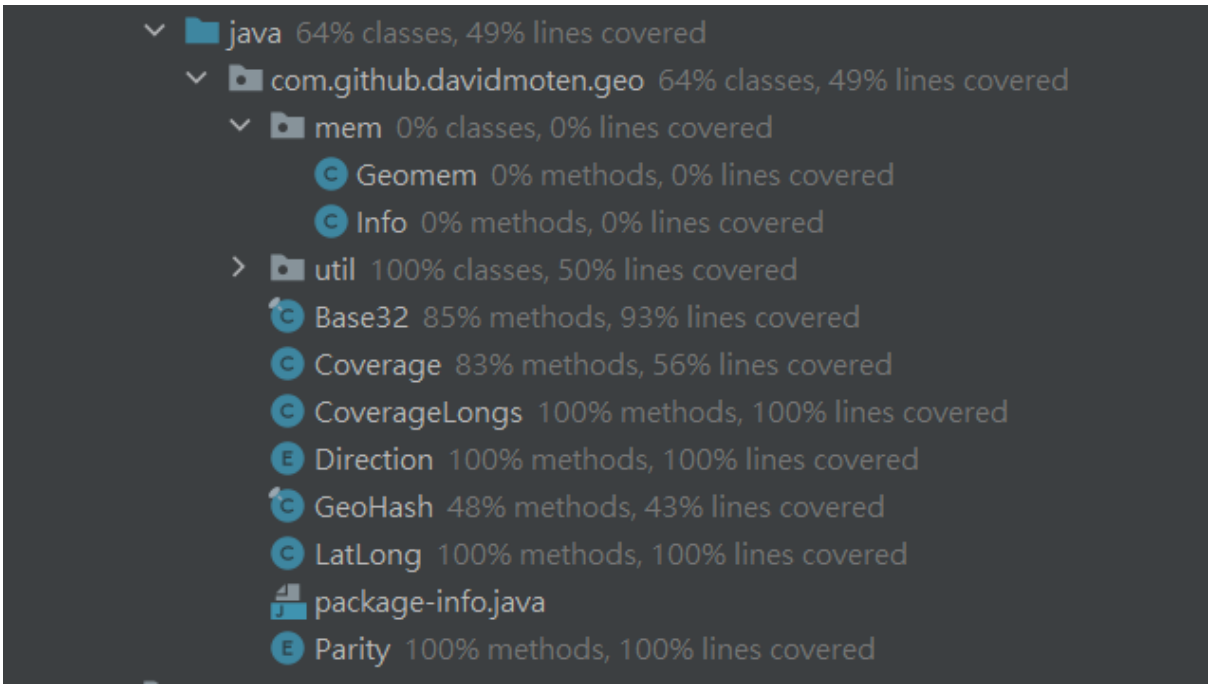
2	0	0	0.005s
tests	failures	ignored	duration

100% successful

Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	2	0	0	0.005s	100%

4.2 Code coverage snapshot



Total coverage

geo

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.github.davidmoten.geo.mem		0%		0%	30	30	61	61	20	20	3	3
com.github.davidmoten.geo		87%		75%	41	149	40	348	9	68	0	10
com.github.davidmoten.geo.util		36%		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

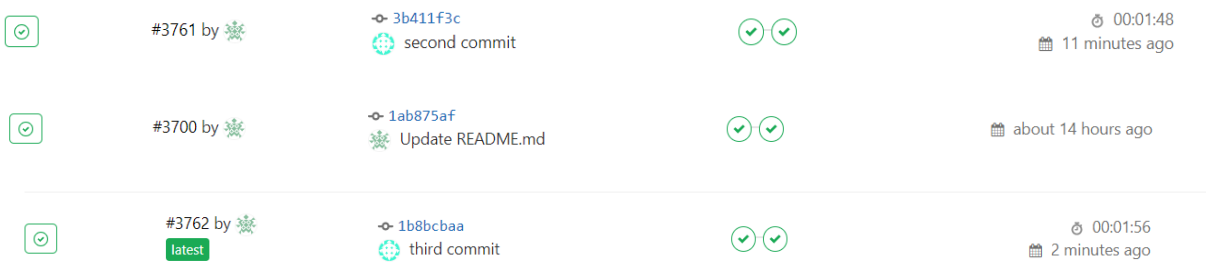
pipeline passed coverage 51%

README.md

pipeline passed coverage 52%

README.md

pipeline passed coverage 54%



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 21 test methods are all passed.The total statement coverage of test is 54%.