# Lab 3 Report

Name:朱峻平
Student ID:107598058
Date:2019/4/24

## 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 and 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 ISP technique.

In particular, based on the statement and branch coverage criteria, the **test requirements** for Lab 3 are to design test cases *with* **basis path and/or graph coverage** for each selected method so that "*each statement and branch of the method will be covered by at least one test case* and *the both minimum* ***statement*** *(node) and* ***branch*** *(edge) coverage are 90%, respectively (greater than Lab 2)*".

### 1.2 Test Strategy

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

(1) 選較有變化或程式較複雜的 method 去畫 CFG，以及設計測試盡量達到最大 statement coverage 以及 branch coverage。
(2) 從 Lab1 和 Lab2 中挑選三個 method 以及再從新的未測 method 中挑選三個。
(3) statement coverage 以及 branch coverage 達到都 90%以上

### 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 | 3 | 4/22 |
| 2 | Learn **ISP** and JUnit | 2 | 4/22 |
| 3 | Design test cases for the selected methods | 12 | 4/23 |
| 4 | Implement test cases | 5 | 4/24 |
| 5 | Perform tests | 1 | 4/24 |
| *n* | Complete Lab3 report | 3 | 4/24 |

### 1.4 Design Approach

依據 CFG 去觀察有哪些 Branch 以及去添加測試案例，設想所以可能狀況去 Cover 住所有可能的分支，再去 Assert 期待值以及可能的例外狀況，盡可能提高 coverage。

### 1.5 Success criteria

所有測試案例都通過，statement coverage 以及 branch coverage 都要達到 90%以上。

## 2 Test Design

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

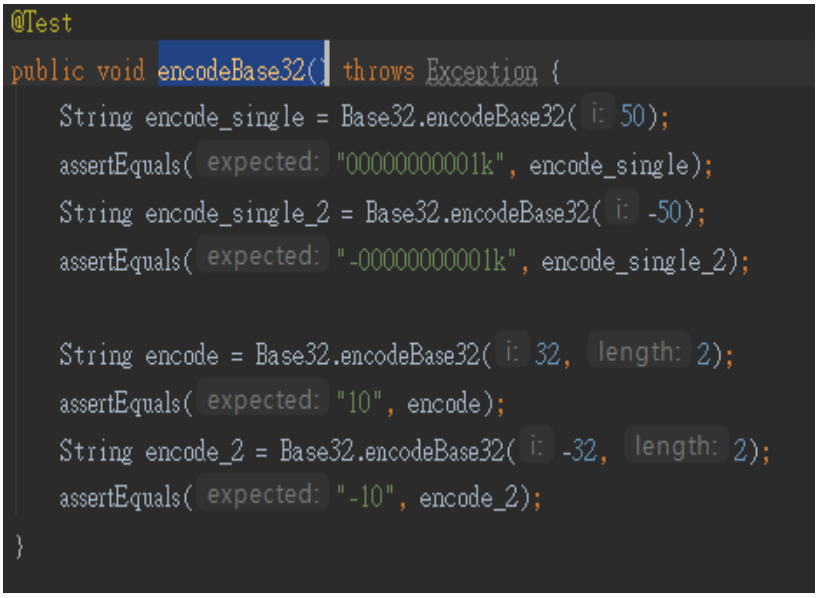
| No. | Class | Method | CFG | Basis Path | Inputs | Expected Outputs |
|---|---|---|---|---|---|---|
| 1 | Base32 | encodeBase32() | 請參考 Excel | | T1: {i = 32, lenth=2}<br>T2={i = -32, lenth=2} | 1. 10<br>2. -10 |
| 2 | Base32 | decodeBase32() | | | T1: {hash= "-1"}<br>T2={hash= ""} | 1. -1<br>2. 0 |
| 3 | GeoHash | adjacentHash() | | | T1 = { hash="TEST" direction= Direction.BOTTOM steps=7}<br>T2 = { hash="TEST" direction= Direction.TOP steps=-7}<br>T3 = { hash="TEST" direction= Direction.LEFT steps=0} | 1.tekw<br>2.tekw<br>3.TEST |
| 4 | Base32 | getCharIndex() | | | T1: {ch= 'b'}<br>T2={ch= 'a'} | 1.10<br>2. IllegalArgumentException |
| 5 | Geomem | Find() | | | T1 = { topLeftLat = 5 topLeftLon=0 bottomRightLat=1 bottomRightLon=5 start=0 finish=5}<br>(有事先進行 add 資料) | 1. New Info<Double,Double> (1.0,2.0,3.0,4.0,689.0)<br>2. IllegalArgumentException |

| | | | | | |
|---|---|---|---|---|---|
| | | | | T2 = { topLeftLat = 5<br>topLeftLon=0<br>bottomRightLat=1<br>bottomRightLon=5<br>start=0<br>finish=5}<br>(沒有事先進行 add 資料) | |
| 6 | GeoHash | fromLongToString() | | T1: {hash = 13}<br>T2={hash = 0}<br>T3={hash = 2} | 1.00000<br>2.00000<br>3.00 |

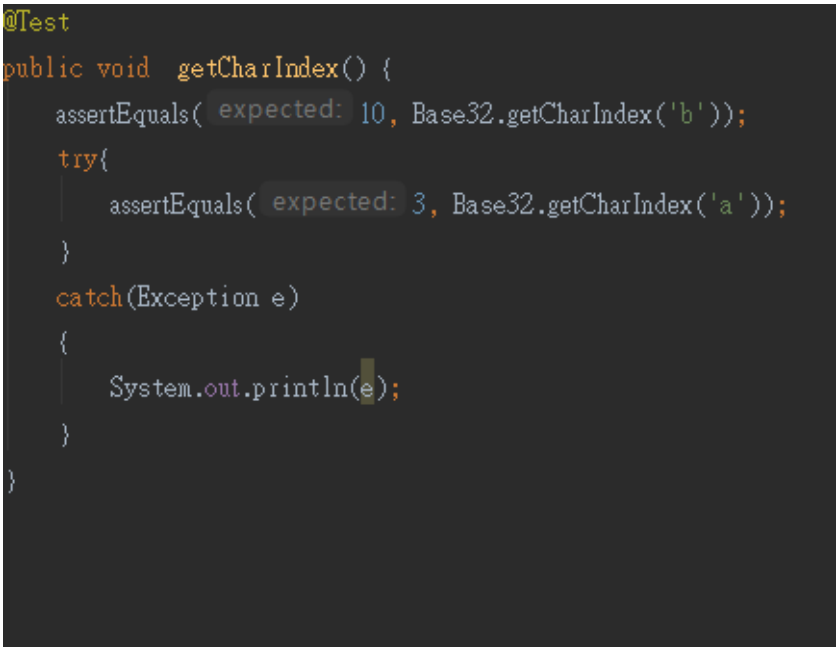
The details of the design are given below:
The Excel file of test cases… ＜－請點我

## 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). ＜－請點我

| No. | Test method | Source test code |
|---|---|---|
| 1 | encodeBase32() |  |

```
@Test
public void encodeBase32() throws Exception {
    String encode_single = Base32.encodeBase32( i: 50);
    assertEquals( expected: "00000000001k", encode_single);
    String encode_single_2 = Base32.encodeBase32( i: -50);
    assertEquals( expected: "-00000000001k", encode_single_2);

    String encode = Base32.encodeBase32( i: 32, length: 2);
    assertEquals( expected: "10", encode);
    String encode_2 = Base32.encodeBase32( i: -32, length: 2);
    assertEquals( expected: "-10", encode_2);
}
```

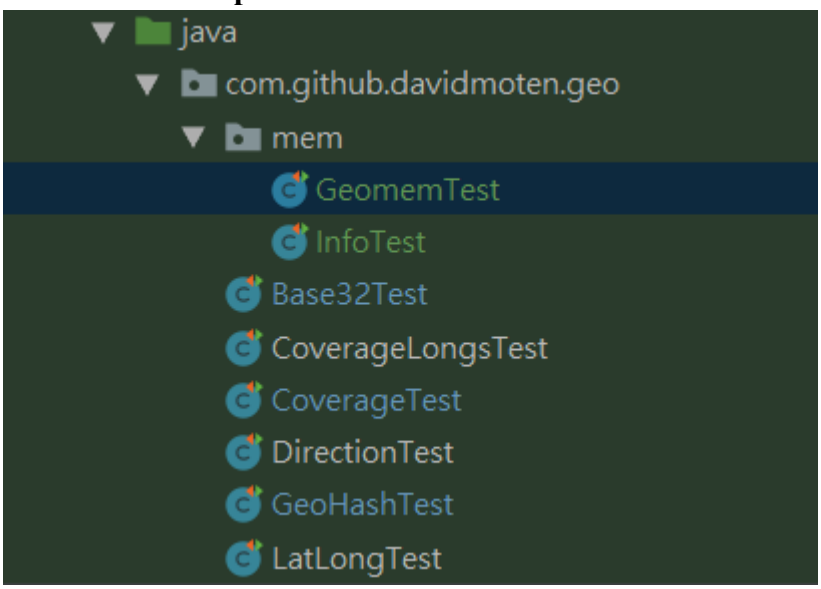
| | | |
|---|---|---|
| 2 | decodeBase32() |  |
| 3 | getCharIndex() |  |

```
public void decodeBase32()   {
        long ans = Base32.decodeBase32("bcde");
        assertEquals(339341, ans);

    long gp = Base32.decodeBase32( hash: "-1");
    assertEquals( expected: -1, gp);

    long temp = Base32.decodeBase32( hash: "");
    assertEquals( expected: 0, temp);

    try{
        long ans_2 = Base32.decodeBase32( hash: "a");
    }
    catch (Exception e){
        System.out.println(e);
    }
    try{
        long ans_3 = Base32.decodeBase32( hash: "-a");
    }
    catch (Exception e){
        System.out.println(e);
    }

}
```

```
@Test
public void  getCharIndex() {
    assertEquals( expected: 10, Base32.getCharIndex('b'));
    try{
        assertEquals( expected: 3, Base32.getCharIndex('a'));
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
```

# 4 Test Results

## 4.1 JUnit test result snapshot




**Test Summary**

| 43 | 0 | 0 | 0.285s |
|---|---|---|---|
| tests | failures | ignored | duration |

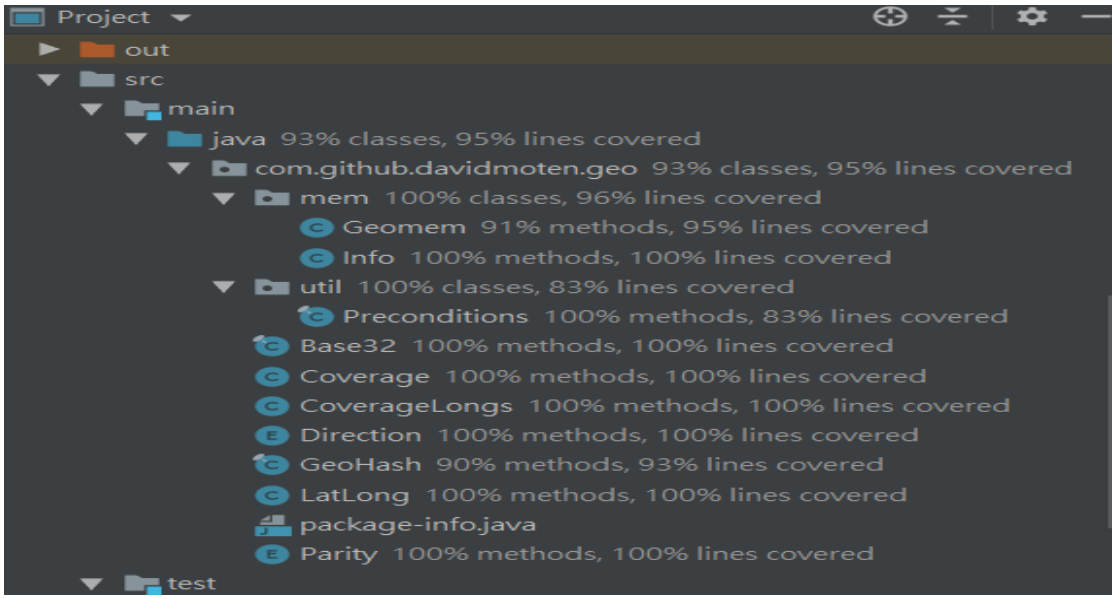
100% successful

Packages | Classes

| Package | Tests | Failures | Ignored | Duration | Success rate |
|---|---|---|---|---|---|
| com.github.davidmoten.geo | 31 | 0 | 0 | 0.209s | 100% |
| com.github.davidmoten.geo.mem | 12 | 0 | 0 | 0.076s | 100% |

## 4.2 Code coverage snapshot

- Coverage of each selected method



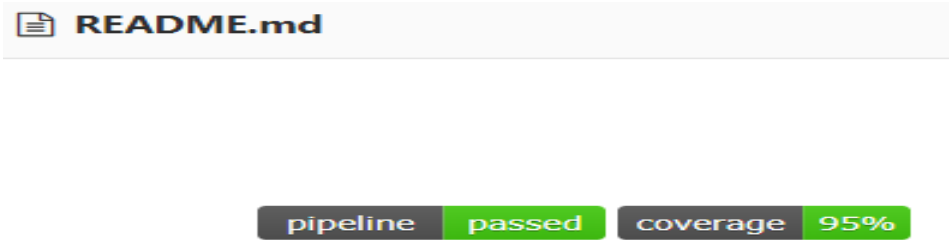
- Total coverage

**geo**

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty | Missed | Lines | Missed | Methods | Missed | Classes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| com.github.davidmoten.geo | | 97% | | 93% | 14 | 149 | 8 | 348 | 3 | 68 | 0 | 10 |
| com.github.davidmoten.geo.mem | | 97% | | 80% | 5 | 30 | 2 | 61 | 1 | 20 | 0 | 3 |
| com.github.davidmoten.geo.util | | 68% | | 75% | 1 | 4 | 1 | 6 | 0 | 2 | 0 | 1 |
| Total | 61 of 2,326 | 97% | 16 of 186 | 91% | 20 | 183 | 11 | 415 | 4 | 90 | 0 | 14 |

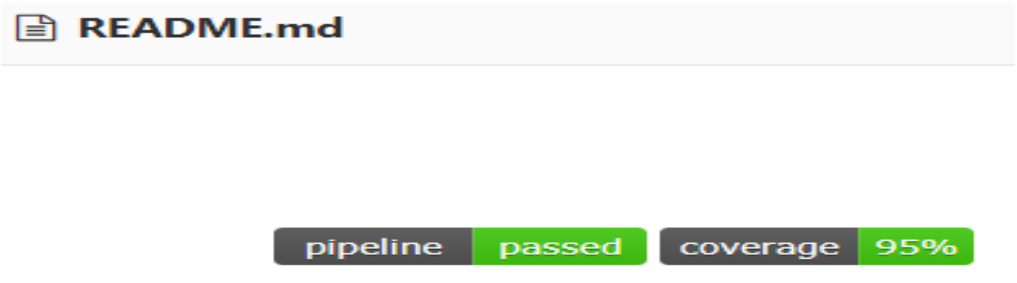
## 4.3 CI result snapshot (3 iterations for CI)

- CI#1

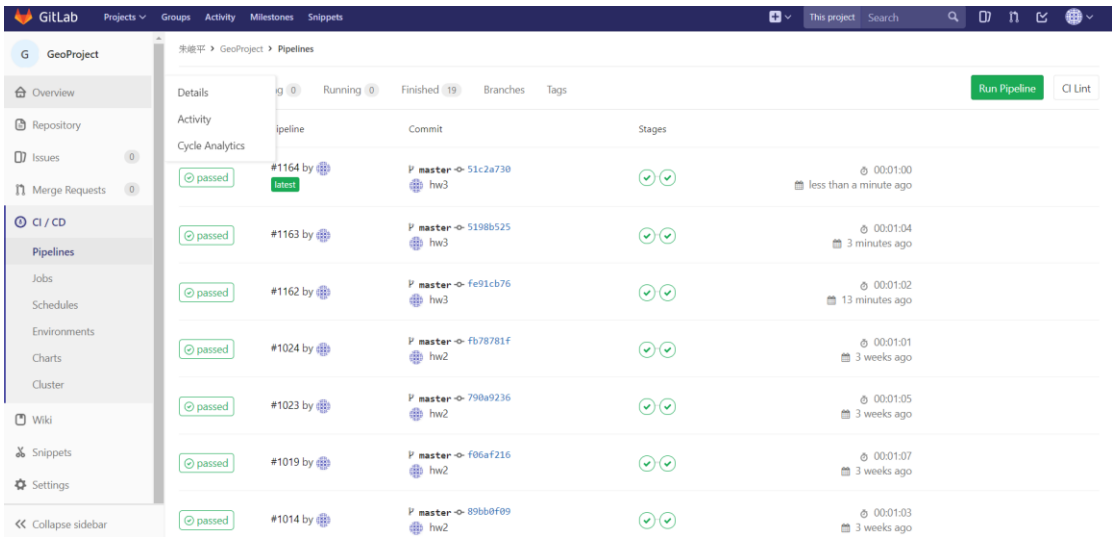



- CI#2


- CI#3




- CI Pipeline




## 5 Summary

已經對 GeoProject 使用 ISP 以及 CFG 方式去進行測試，Statement Coverage 達到 95%以及 Branch Coverage 達到 91%，經過 lab3 的練習我更了解如何畫 CFG 圖以及更有效率的對程式進行測試，已經不是盲測了，也更加了解 statemeent coverage 和 Branch Coverage 的意義以及對程式開罰人員能帶來的好處，經過這次作業讓我受益良多。