

# Lab 2 Report

Name:朱峻平

Student ID:107598058

Date:2019/3/28

## 1 Test Plan

### 1.1 Test requirements

The Lab 2 requires to (1) select 15 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) 選較有變化或程式較複雜的 method 去設計 Test case 盡量達到最大 statement 覆蓋率。
- (2) 使用 ISP 設計 Test case 完成所有可能性的測試盡量達到該 method 最大 statement 覆蓋率。
- (3) Getter or Setter 不特別進行全數完整 ISP 設計只做基本測試。

### 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	3/25
2	Learn ISP and JUnit	2	3/25
3	Design test cases for the selected methods	12	3/26
4	Implement test cases	5	3/27
5	Perform tests	1	3/28
n	Complete Lab2 report	3	3/28

## 1.4 Design Approach

使用 ISP 設計 Test Case 需先辨別輸入參數可能的 partitions 和邊界值，針對 input 設計條件，並設計可能組合以及邊界值，最終根據這些組合，去設想測試資料並實作出來，最終將所有可能組合 cover 住，確保 method 測試完整性。

我在這次的 lab 都會先設想什麼樣情況此 method 會丟例外，去找參數可設計值，再設想測試條件，並經過組合，最終實作並列出每種組合的設計參數以及期待值。

## 1.5 Success criteria

所以用 ISP 設計的 Test case 要全數通過,statement 覆蓋率達 70%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.

詳細 Design 放在 master/LabReport/Lab2 與 lab2 報告在同層資料夾下。

The Excel file of test cases... [link](#)

## 3 Test Implementation

The design of test cases specified in Section 2 was implemented using JUnit

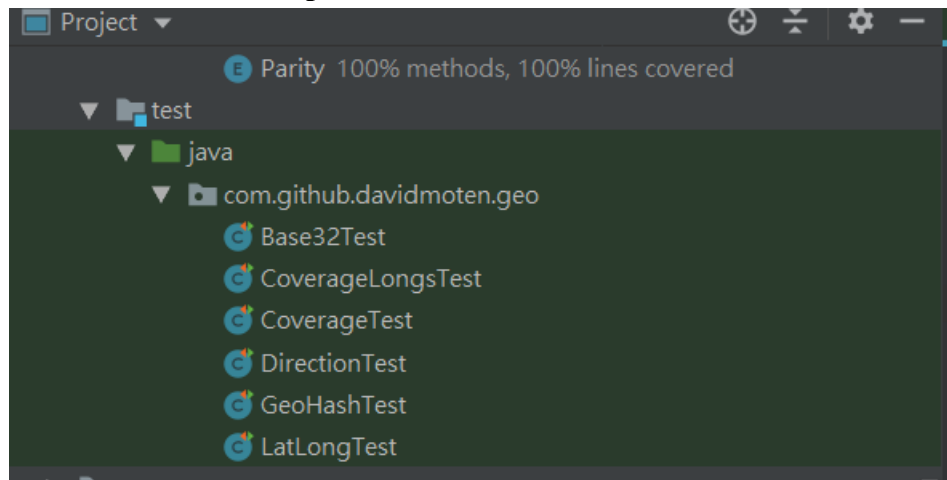
4. The test scripts of 3 selected test cases are given below. The rest of the test script implementations can be found in the [link](#) (or JUnit files).

No.	Test method	Source code
1	padLeftWithZerosToLength()	<pre>@Test public void padLeftWithZerosToLength() {     assertEquals("0000Test1", Base32.padLeftWithZerosToLength("Test1",9));     assertEquals("Test1", Base32.padLeftWithZerosToLength("Test1",-1));     assertEquals("000000000", Base32.padLeftWithZerosToLength("",9)); }</pre>
2	getCharIndex()	<pre>@Test public void getCharIndex() {</pre>

		<pre> assertEquals(0, Base32.getCharIndex('0'));  assertEquals(10, Base32.getCharIndex('b'));  try{     assertEquals(3, Base32.getCharIndex('a')); }  catch(Exception e)  {     System.out.println(e); }  } </pre>
3	decodeBase32()	<pre> @Test public void decodeBase32() {      long ans = Base32.decodeBase32("bcde");      assertEquals(339341, ans);      try{          long ans_2 = Base32.decodeBase32("a");      }      catch (Exception e){          System.out.println(e);      }      try{          long ans_3 = Base32.decodeBase32("-a");      }      catch (Exception e){          System.out.println(e);      }  } </pre>

## 4 Test Results

### 4.1 JUnit test result snapshot



#### Test Summary

31	0	0	0.126s
tests	failures	ignored	duration

**100%**  
successful

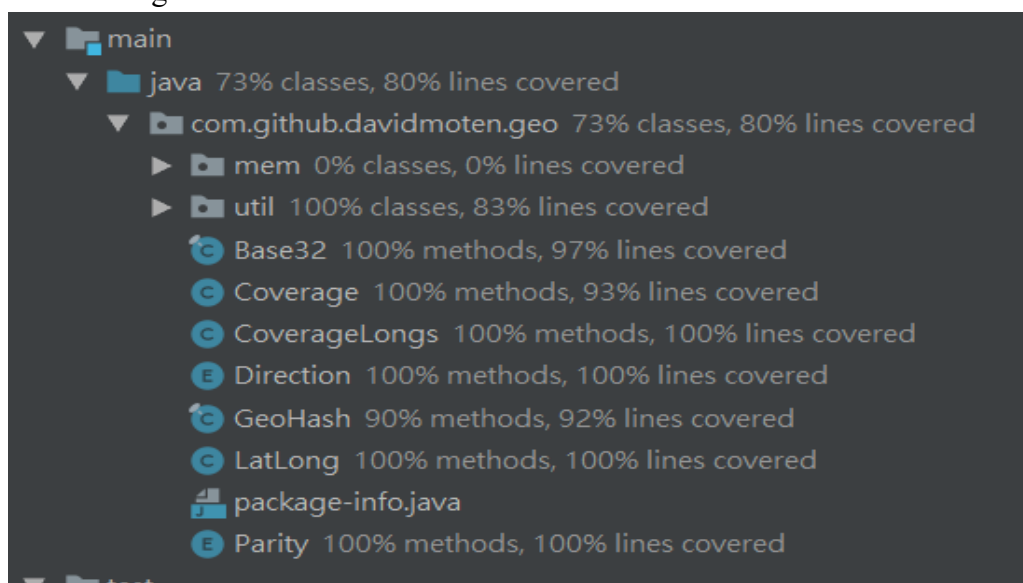
#### Packages

#### Classes

Package	Tests	Failures	Ignored	Duration	Success rate
<a href="#">com.github.davidmoten.geo</a>	31	0	0	0.126s	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
<a href="#">com.github.davidmoten.geo.mem</a>	<div></div>	0%	<div></div>	0%	30 30	61 61	20 20	3 3
<a href="#">com.github.davidmoten.geo</a>	<div></div>	96%	<div></div>	89%	19 149	11 348	3 68	0 10
<a href="#">com.github.davidmoten.geo.util</a>		68%	<div></div>	75%	1 4	1 6	0 2	0 1
Total	413 of 2,326	82%	38 of 186	79%	50 183	73 415	23 90	3 14

4.3 CI result snapshot (3 iterations for CI)

- CI#1

README.md

pipeline

passed

coverage

79%

- CI#2

README.md


pipeline

passed

coverage

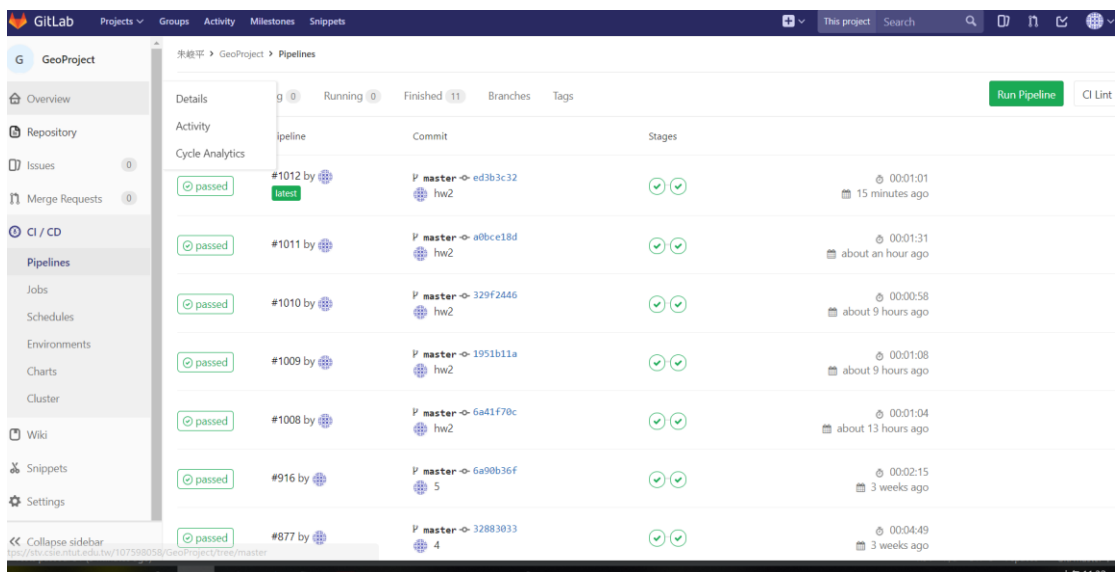
80%

- CI#3

 README.md

pipeline passed coverage 80%

- CI Pipeline



The screenshot shows the GitLab CI Pipeline interface for a project named 'GeoProject'. The left sidebar contains navigation links: Overview, Repository, Issues, Merge Requests, CI / CD (selected), Pipelines, Jobs, Schedules, Environments, Charts, Cluster, Wiki, Snippets, and Settings. The main content area displays the 'Pipelines' tab with a table of pipeline runs. The table has columns for 'Pipeline', 'Commit', and 'Stages'. Each row represents a pipeline run, showing its ID, the commit it was triggered by, the status (all 'passed'), and the time taken. The 'Stages' column shows two stages for each pipeline, both marked with green checkmarks. A 'Run Pipeline' button and a 'CI Lint' link are visible in the top right corner of the pipeline list.

Pipeline	Commit	Stages
#1012 by latest	P master -> ed3b3c32 hw2	00:01:01 15 minutes ago
#1011 by hw2	P master -> a0bce18d hw2	00:01:31 about an hour ago
#1010 by hw2	P master -> 329f2446 hw2	00:00:58 about 9 hours ago
#1009 by hw2	P master -> 1951b11a hw2	00:01:08 about 9 hours ago
#1008 by hw2	P master -> 6a41f70c hw2	00:01:04 about 13 hours ago
#916 by 5	P master -> 6a90b36f 5	00:02:15 3 weeks ago
#877 by 4	P master -> 32883033 4	00:04:49 3 weeks ago

## 5 Summary

在 Lab2 中我列出了 15 個我使用 ISP 設計，覺得具代表性的 **test case**，這次作業我已經習慣 JUnit 以及新的 IDE 開發工具，這次花了比較多時間在事前設計，列出所有可能，以及設想測試資料該如何設計才能達到最大 **statement coverage**，所以在撰寫測試上遵循自己的策略，沒有遇到太大的問題，最後測試覆蓋率達到 80%，之後會再研究程式碼補足我測試沒覆蓋到的部分，以上是我的心得。