Lab 2 Report

Name: 呂聖文
Student ID:108598007
Date:2020/4/17

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) 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	2 h	4/9
2	Learn **ISP** and JUnit	1 day	4/9
3	Design test cases for the selected methods	2days	4/10,4/11
4	Implement test cases	5days	4/12~4/16

5	Perform tests	1 h	4/17
6	Complete Lab2 report	1days	4/17

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.

No.	Class	Method	Test Objective	Inputs	Expected Outputs
1	Base32	encodeBase32(long i, int length)	10 進制轉 32 進制(i<0,length<0)	i:-75324 length:-4	"-29jw"
2	Base32		10 進制轉 32 進制(i<0,length=0)	i:-75324 length:0	"-29jw"
3	Base32		10 進制轉 32 進制(i<0,length>0)	i:-75324 length:4	"-29jw"
4	Base32		10 進制轉 32 進制(i=0,length<0)	i:0 length:-4	"0"
5	Base32		10 進制轉 32 進制(i=0,length=0)	i:0 length:0	"0"
6	Base32		10 進制轉 32 進制(i=0,length>0)	i:0 length:4	"0000"
7	Base32		10 進制轉 32 進制(i>0,length<0)	i:75324 length:-4	"29jw
8	Base32		10 進制轉 32 進制(i>0,length=0)	i:75324 length:0	"29jw"
9	Base32		10 進制轉 32 進制(i>0,length>0)	i:75324 length:4	"29jw"
10	Base32	encodeBase32(long i)	10 進制轉 32 進制(i<0)	i:-75324	"0000000029jw"
11	Base32		10 進制轉 32 進制(i=0)	i:0	"000000000000"
12	Base32		10 進制轉 32 進制(i>0)	i:75324	"-0000000029jw"

13	Base32	decodeBase32(String hash)	32 進制轉 10 進制(hash>0)	hash:"29jw"	75324
14	Base32		32 進制轉 10 進制(hash=0)	hash:"0"	0
15	Base32		32 進制轉 10 進制(hash<0)	hash:"-29jw"	-75324
16	Base32	getCharIndex(char ch)	搜尋字元為陣列第幾 index	ch:"b"	10
17	Base32		搜尋非陣列中字元是否正常拋出例外	ch:"i"	IllegalArgumentException
18	Base32	padLeftWithZerosToLength(String s,int length)	測試補 0 功能(s<0,i<0)	s:"-29jw",i:-5	"-29jw"
19	Base32		測試補 0 功能(s<0,i=0)	s:"-29jw",i:0	"-29jw"
20	Base32		測試補 0 功能(s<0,i>0)	s:"-29jw",i:-5	"-29jw"
21	Base32		測試補 0 功能(s=0,i<0)	s:"0",i:-5	"0"
22	Base32		測試補 0 功能(s=0,i=0)	s:"0",i:0	"0"
23	Base32		測試補 0 功能(s=0,i>0)	s:"0",i:5	"00000"
24	Base32		測試補 0 功能(s>0,i<0)	s:"29jw",i:-5	"29jw"
25	Base32		測試補 0 功能(s>0,i=0)	s:"29jw",i:0	"29jw"
26	Base32		測試補 0 功能(s>0,i>0)	s:"29jw",i:5	"029jw"
27	CoverageLongs	getRatio()	測試參數 ratio 與回傳 ratio 是否同一數字(ratio<0)	ratio:-3.14	-3.14
28	CoverageLongs		測試參數 ratio 與回傳 ratio 是否同一數字(ratio=0)	ratio:0	0
29	CoverageLongs		測試參數 ratio 與回傳 ratio 是否同一數字(ratio>0)	ratio:3.14	3.14
30	CoverageLongs	getCount()	回傳初始化 set 數量 count(count<0)	count:-3	-3
31	CoverageLongs		回傳初始化 set 數量 count(count=0)	count:0	0
32	CoverageLongs		回傳初始化 set 數量 count(count>0)	count:3	3
33	CoverageLongs	getHashLength()	取 array 第一個值&0x0f 後的值(<0)	arr[0]=-10	6
34	CoverageLongs		取 array 第一個值&0x0f 後的值(=0)	arr[0]=0	0

35	CoverageLongs	getHashLength()	取 array 第一個值&0x0f 後的值 (>0)	arr[0]=10	10
36	Coverage	getHashes()	回傳初始化之 set 位址 (set.size=0)	Set=[]	getHashes().size()==0
37	Coverage		回傳初始化之 set 位址 (set.size>0)	Set=[e1,e2,e3,e4]	getHashes().size()==4
38	Coverage	getRatio()	測試參數 ratio 與回傳 ratio 是否同一數字 (ratio<0)	ratio:-3.14	-3.14
39	Coverage		測試參數 ratio 與回傳 ratio 是否同一數字 (ratio=0)	ratio:0	0
40	Coverage		測試參數 ratio 與回傳 ratio 是否同一數字 (ratio>0)	ratio:3.14	3.14
41	Coverage	getHashLength()	陣列第一元素長度為何 len(set[0])=0	Set=[]	0
42	Coverage		陣列第一元素長度為何 len(set[0])>0	Set[0]="number1"	7
43	GeoHash	adjacentHash(String hash, Direction direction)	hash 的 direction 方位區域為何 (邊界,top)	hash:"zz" direction:top	"gb"
44	GeoHash		hash 的 direction 方位區域為何 (邊界,bottom)	hash:"zz" direction: bottom	"zy"
45	GeoHash		hash 的 direction 方位區域為何 (邊界,left)	hash:"zz" direction: left	"zx"
46	GeoHash		hash 的 direction 方位區域為何 (邊界,right)	hash:"zz" direction: right	"bp"
47	GeoHash		hash 的 direction 方位區域為何 (非邊界,top)	hash:"ts" direction:top	"tt"
48	GeoHash		hash 的 direction 方位區域為何 (非邊界,bottom)	hash:"ts" direction: bottom	"te"
49	GeoHash		hash 的 direction 方位區域為何 (非邊界,left)	hash:"ts" direction: left	"tk"
50	GeoHash		hash 的 direction 方位區域為何 (非邊界,right)	hash:"ts" direction: right	"tu"
51	GeoHash	adjacentHash(String hash, Direction direction, int steps)	hash 的 direction 方位 steps 格區域為何 (邊界,top,steps<0)	hash:"zz" direction:top steps:-5	"zf"

52	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,bottom,steps<0)	hash:"zz" direction:bottom steps:-5	"gu"
53	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,left,steps<0)	hash:"zz" direction:left steps:-5	"cp"
54	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,right,steps<0)	hash:"zz" direction:right steps:-5	"yx"
55	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,top,steps=0)	hash:"zz" direction:top steps:0	"zz"
56	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,bottom,steps=0)	hash:"zz" direction:bottom steps:0	"zz"
57	GeoHash	adjacentHash(String hash, Direction direction, int steps)	hash 的 direction 方位 steps 格區域為何(邊界,left,steps=0)	hash:"zz" direction:left steps:0	"zz"
58	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,right,steps=0)	hash:"zz" direction:right steps:0	"zz"
59	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,top,steps>0)	hash:"zz" direction:top steps:5	"gu"
60	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,bottom,steps>0)	hash:"zz" direction:bottom steps:5	"zf"
61	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,left,steps>0)	hash:"zz" direction:left steps:5	"yx"
62	GeoHash		hash 的 direction 方位 steps 格區域為何(邊界,right,steps>0)	hash:"zz" direction:right steps:5	"cp"

63	GeoHash	adjacentHash(String hash, Direction direction, int steps)	hash 的 direction 方位 steps 格區域為何 (非邊界,top,steps<0)	hash:"ts" direction:top steps:-5	"mx"
64	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,bottom,steps<0)	hash: ="ts" direction:bottom steps:-5	"v9"
65	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,left,steps<0)	hash:"ts" direction:left steps:-5	"wu"
66	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,right,steps<0)	hash:"ts" direction:right steps:-5	"sk"
67	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,top,steps=0)	hash:"ts" direction:top steps:0	"ts"
68	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,bottom,steps=0)	hash:"ts" direction:bottom steps:0	"ts"
69	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,left,steps=0)	hash:"ts" direction:left steps:0	"ts"
70	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,right,steps=0)	hash:"ts" direction:right steps:0	"ts"
71	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界,top,steps>0)	hash:"ts" direction:top steps:5	"v9
72	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界, bottom,steps>0)	hash:"ts" direction:bottom steps:5	"mx"
73	GeoHash		hash 的 direction 方位 steps 格區域為何 (非邊界, left,steps>0)	hash:"ts" direction:left steps:5	"sk"

74	GeoHash	adjacentHash(String hash, Direction direction, int steps)	hash 的 direction 方位 steps 格區域為何 (非邊界,right,steps>0)	hash:"ts" direction:right steps:5	"wu"
75	GeoHash	decodeHash(String geohash)	Geohash 轉為經緯度 (geohash<0)	Geohash:"-sb52"	getLat()==72.31201171875 getLon()==157.52197265625
76	GeoHash		Geohash 轉為經緯度 (geohash=0)	Geohash:"0"	getLat()==-67.5 getLon()==-157.5
77	GeoHash		Geohash 轉為經緯度 (geohash>0)	Geohash:"sb52"	getLat()==0.087890625 getLon()==38.49609375
78	GeoHash	neighbours(String hash)	測試 w 區域八方位	("w")	geohash.get(0)==”t” geohash.get(1)==”x” geohash.get(2)==”y” geohash.get(3)==”q” geohash.get(4)==”v” geohash.get(5)==”m” geohash.get(6)==”z” geohash.get(7)==”r”
79	GeoHash	right(String hash)	回傳右邊(東方)區域 geohash 編號 (邊界)	hash:"zz"	"bp"
80	GeoHash		回傳右邊(東方)區域 geohash 編號 (非邊界)	hash:"ts"	"tu"
81	GeoHash	left(String hash)	回傳左邊(西方)區域 geohash 編號 (邊界)	hash:"zz"	"zx"
82	GeoHash		回傳左邊(西方)區域 geohash 編號 (非邊界)	hash:"ts"	"tk"
83	GeoHash	bottom(String hash)	回傳下方(南方)區域 geohash 編號 (邊界)	hash:"zz"	"zy"

84	GeoHash		回傳下方(南方)區域 geohash 編號(非邊界)	hash:"ts"	"te"
85	GeoHash	top(String hash)	回傳上方(北方)區域 geohash 編號(邊界)	hash:"zz"	"gb"
86	GeoHash		回傳上方(北方)區域 geohash 編號(非邊界)	hash:"ts"	"tt"
87	Geomem	add(double lat, double lon, long time, T t)	加入一個紀錄資訊包含經緯度、時間、物件	lat:-20 lon:150 time:5 t:"E2"(String)	getLat()==-20 getLon()==150 getTime()==5 getValue()=="E2"
88	Info	toString()	顯示 info 資訊	lat:3.123 lon:4.123 time:10 value:5 Option:"NTUT"	"Info [lat=3.123, lon=4.123, time=10, value=5, id=Optional.of(NTUT)]"

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).

No.	Test method	Source code
1	`encodeBase32_T1()`	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java
2	`encodeBase32_T2()`	
3	`encodeBase32_T3()`	
4	`encodeBase32_T4()`	
5	`encodeBase32_T5()`	
6	`encodeBase32_T6()`	
7	`encodeBase32_T7()`	
8	`encodeBase32_T8()`	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/c
9	`encodeBase32_T9()`	

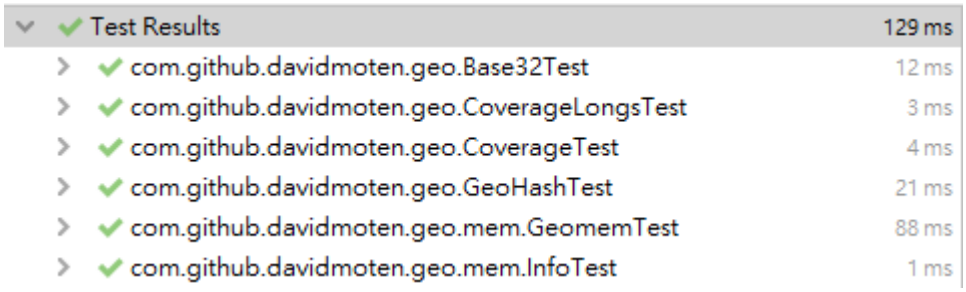
10	encodeBase32_noLength_positive_T1()	om/github/davidmoten/geo/Base32Test.java
11	encodeBase32_noLength_zero_T2()	
12	encodeBase32_noLength_negative_T3()	
13	decodeBase32_positive_T1()	
14	decodeBase32_zero_T2()	
15	decodeBase32_negative_T3()	
16	getCharIndex_exception_T1()	
17	getCharIndex_exception_T2()	
18	getCharIndex_padLeftWithZerosToLength_T1()	
19	getCharIndex_padLeftWithZerosToLength_T2()	
20	getCharIndex_padLeftWithZerosToLength_T3()	
21	getCharIndex_padLeftWithZerosToLength_T4()	
22	getCharIndex_padLeftWithZerosToLength_T5()	
23	getCharIndex_padLeftWithZerosToLength_T6()	
24	getCharIndex_padLeftWithZerosToLength_T7()	
25	getCharIndex_padLeftWithZerosToLength_T8()	
26	getCharIndex_padLeftWithZerosToLength_T9()	
27	getRatio_T1()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/CoverageLongsTest.java
28	getRatio_T2()	
29	getRatio_T3()	
30	getCount_T1()	
31	getCount_T2()	
32	getCount_T3()	
33	getHashLength_T1()	
34	getHashLength_T2()	
35	getHashLength_T3()	
36	getHashes_T1()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/CoverageTest.java
37	getHashes_T2()	
38	getRatio_T1()	
39	getRatio_T2()	
40	getRatio_T3()	
41	getHashLength_T1()	
42	getHashLength_T2()	
43	adjacentHash_T1()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java
44	adjacentHash_T2()	
45	adjacentHash_T3()	
46	adjacentHash_T4()	

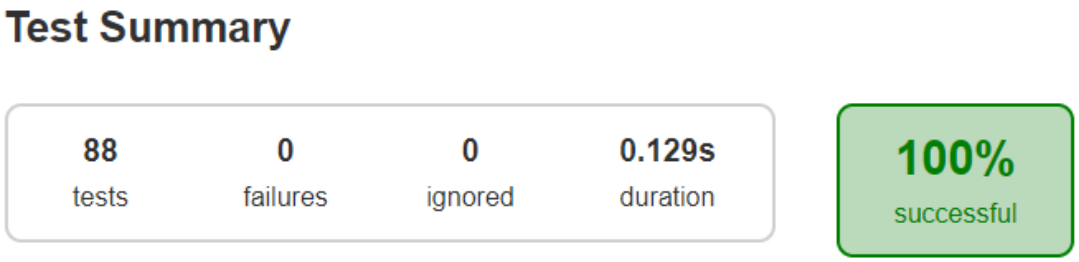
47	adjacentHash_T5()	
48	adjacentHash_T6()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java
49	adjacentHash_T7()	
50	adjacentHash_T8()	
51	adjacentHash_steps_T1()	
52	adjacentHash_steps_T2()	
53	adjacentHash_steps_T3()	
54	adjacentHash_steps_T4()	
55	adjacentHash_steps_T5()	
56	adjacentHash_steps_T6()	
57	adjacentHash_steps_T7()	
58	adjacentHash_steps_T8()	
59	adjacentHash_steps_T9()	
60	adjacentHash_steps_T10()	
61	adjacentHash_steps_T11()	
62	adjacentHash_steps_T12()	
63	adjacentHash_steps_T13()	
64	adjacentHash_steps_T14()	
65	adjacentHash_steps_T15()	
66	adjacentHash_steps_T16()	
67	adjacentHash_steps_T17()	
68	adjacentHash_steps_T18()	
69	adjacentHash_steps_T19()	
70	adjacentHash_steps_T20()	
71	adjacentHash_steps_T21()	
72	adjacentHash_steps_T22()	
73	adjacentHash_steps_T23()	
74	adjacentHash_steps_T24()	
75	decodeHash_T1()	
76	decodeHash_T2()	
77	decodeHash_T3()	
78	right_T1()	
79	right_T2()	
80	left_T1()	
81	left_T2()	
82	top_T1()	
83	top_T2()	

84	bottom_T1()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java
85	bottom_T2()	
86	neighbours()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java
87	find_OneItemInRange()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/mem/GeomemTest.java
88	testToString()	https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/mem/InfoTest.java

5 Test Results

5.1 JUnit test result snapshot



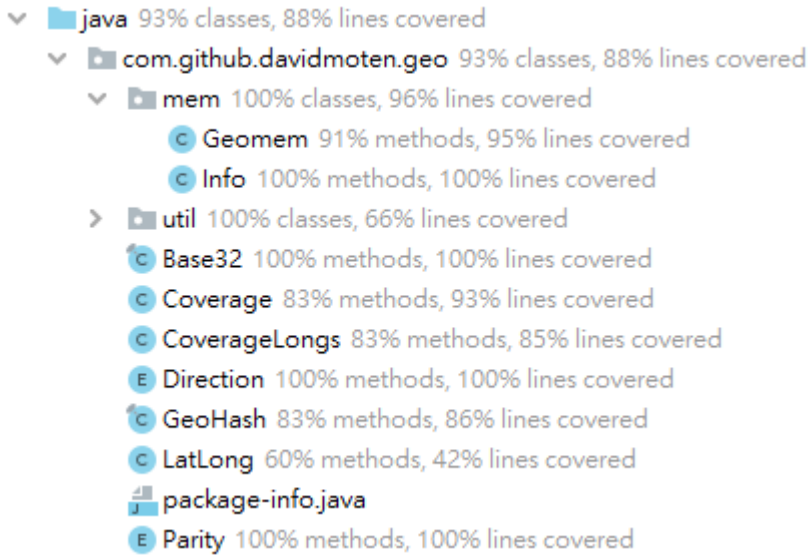


Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	86	0	0	0.040s	100%
com.github.davidmoten.geo.mem	2	0	0	0.089s	100%

5.2 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		85%		76%	42	149	45	348	11	68	0	10
com.github.davidmoten.geo.util		36%		50%	2	4	2	6	0	2	0	1
com.github.davidmoten.geo.mem		96%		65%	8	30	2	62	1	20	0	3
Total	301 of 2,327	87%	47 of 186	74%	52	183	49	416	12	90	0	14

5.3 CI result snapshot (3 iterations for CI)

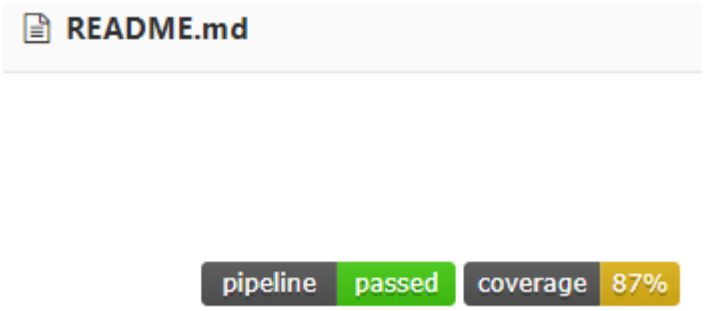
- CI#1

- CI#2

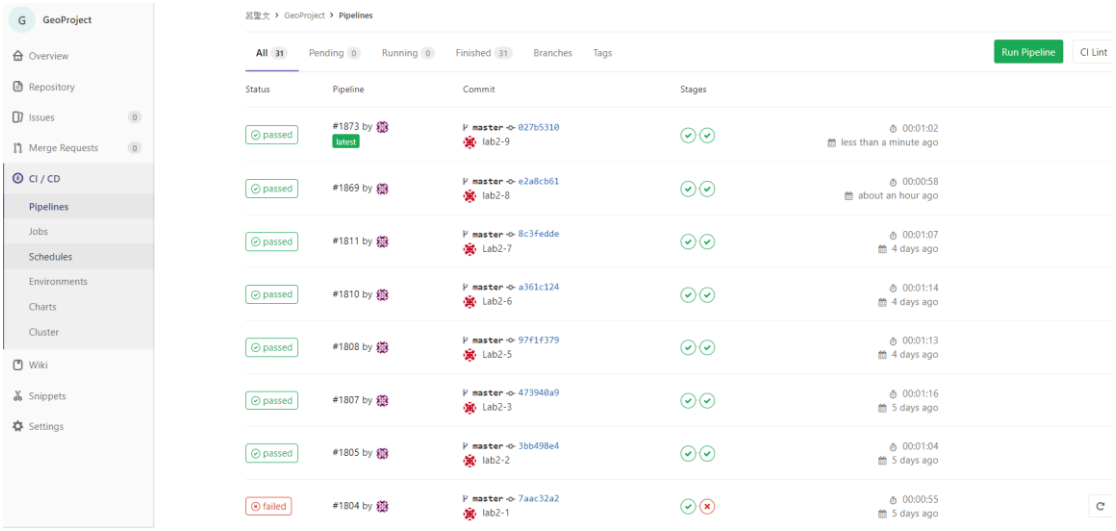
- CI#3

- CI#4

- CI#5



- CI Pipeline



6 Summary

In Lab 2, **88 test cases have been designed and implemented using JUnit and the ISP technique**. The test is conducted in 5 CI and **the execution results of the 15 test methods are all passed. The total statement coverage of the test is 87%.** Thus, the test requirements described in Section 1 are satisfied.

因上次作業我寫到高達 84%，且內容稍亂，因此這次我重新開始撰寫，並使用 ISP 方式進行測試。ISP 詳細內容已付在 excel 檔案中。

這次與 Lab1 最大不同為本次使用 ISP 的方式進行測試，使用 ISP 之前

必須先進行規劃，規劃如何切割該方法進行測試，對於切割又是需要時間思考，怎麼切才能達到有效結果，這也是我在這次作業中花最多時間的部分，過程中也有遇到不知該如何切割才能有效測試，我自己也不是非常確定我的方式是否正確，但主要能測試該功能的運作是否正常，雖然本次作業花費非常多時間但其中也學到很多，經過本次作業我才真正了解測試的麻煩性以及需要極大的耐心才能將測試做到好。