**Lab 1 Report**

Name 陳宇宏

Student ID 110598067

Date 2022/3/1

1. **Test Plan**
   1. **Test requirements**

The Lab 1 requires to (1) select **15 methods** from **6 classes** of the SUT (GeoProject), (2) design Unit test cases based on the experience or intuition for the selected methods, (3) develop test scripts to implement the test cases, (4) execute the test script on the selected methods, and (5) report the test results.

In particular, based on the statement coverage criterion, the **test requirements** for Lab 1 are to design test casesfor 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 40%*”.

* 1. **Strategy**

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

1. select those public methods that are easy to understand and have primitive types of input and output parameters (if possible).
2. set the objective of the minimum statement coverage to be 50% initially and (if necessary) adjust the objective based on the time available.
3. learn the necessary skills and tools as soon as possible.
4. design the test cases for those selected methods by considering
   1. the possible **valid values** and **combinations** of the input parameters.
   2. the **boundary values** of the input parameters.
   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 | 1 | 3/10 |
| 2 | Learn JUnit | 0.5 | 3/11 |
| 3 | Design test cases for the selected methods | 2 | 3/12 |
| 4 | Implement test cases | 2 | 3/12 |
| 5 | Perform test | 1 | 3/12 |
| 6 | Complete Lab1 report | 1 | 3/12 |

* 1. **Success criteria**

All test cases designed for the selected methods must pass (or "90% of all test cases must pass) and *the statement coverage should have achieved at least 40%.*

1. **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):String | Test if it returns the base 32 encoding of the given length from a Long geohash. | 75324, 4 | “29jw” |
| 2 | Base32 | +encodeBase32(long i):String | Test if returns the base 32 encoding of length GeoHash.MAX\_HASH\_LENGTH from a Long geohash. | 15324 | “000000000fyw” |
| 3 | Base32 | +decodeBase32(String): long | Test if it returns the conversion of a base32 geohash to a long. | "29jw" | 75324 |
| 4 | GeoHash | +encodeHash(double,double):String | Test if it returns a geohash of length MAX\_HASH\_LENGTH (12) for the given WGS84 point (latitude,longitude). | 25.033821717782278, 121.56459135583758 | "wsqqqm28s695" |
| 5 | GeoHash | +encodeHash(double,double,int):String | Test if it returns a geohash of given length for the given WGS84 point (latitude,longitude). | 25.033821717782278, 121.56459135583758,8 | “wsqqqm28” |
| 6 | GeoHash | + encodeHash(LatLong):String | Test if it returns a geohash of of length MAX\_HASH\_LENGTH (12) for the given WGS84 point. | new LatLong(25.033821717782278, 121.56459135583758) | "wsqqqm28s695" |
| 7 | GeoHash | + encodeHash(LatLong,int):String | Test if it returns a geohash of given length for the given WGS84 point. | new LatLong(25.033821717782278, 121.56459135583758),4 | "wsqq" |
| 8 | GeoHash | +hashContains(String, double, double):boolean | Test if it returns true if and only if the bounding box corresponding to the hash contains the given lat and long. | "wsqqqm28s",25.0338077545166, 121.564621925354 | true |
| "wsqqqm28s",25.03385066986084, 121.564621925354 | true |
| "wsqqqm28s",25.03385066986084, 121.56457901000977 | true |
| "wsqqqm28s",25.0338077545166, 121.56457901000977 | true |
| "wsqqqm28s",25.0338077545165, 121.564621925354 | false |
| "wsqqqm28s",25.03385066986085, 121.564621925354 | false |
| "wsqqqm28s",25.03385066986084, 122.56457901000977 | false |
| "wsqqqm28s",25.0338077545166, 120.56457901000977 | false |
| 9 | GeoHash | +adjacentHash (String, Direction) :String | Test if it returns the adjacent hash in given Direction. | "wsqqhp8hk1mn",Direction.TOP | "wsqqhp8hk1mp" |
| 10 | GeoHash | +adjacentHash (String, Direction,int) :String | Test if it returns the adjacent hash N steps in the given Direction. | "wsqqhp8hk1mn",Direction.TOP,3 | “wsqqhp8hk1t1” |
| 11 | GeoHash | +right(String):String | Test if it returns the adjacent hash to the right (east). | "wsqqhjwyuvdu" | "wsqqhjwyuveh" |
| 12 | GeoHash | +bottom(String):String | Test if it returns the adjacent hash to the bottom (south). | "wsqqhjwyuvdu" | "wsqqhjwyuvdg" |
| 13 | GeoHash | +decodeHash(String):String | Test if it returns a latitude,longitude pair as the centre of the given geohash. | "wsqqhjwyuvdu" | new LatLong(24.99233888, 121.47432117) |
| 14 | GeoHash | +hashLengthToCoverBoundingBox(double, double, double, double):int | Test if it returns the maximum length of hash that covers the bounding box. | 24.67693180092837, 121.73868842362948,22.09087140332405, 120.75488854089994 | 1 |
| 36.633950257465244, 138.20176404334265,-1.2383111850902806, 103.60559538049397 | 0 |
| 29.240501036359017, 107.95600702691077,29.230257104690256, 107.92737197784561 | 3 |
| 15 | GeoHash | +neighbours(String hash): List<String> | Test if it returns a list of the 8 surrounding hashes for a given hash in order left,right,top,bottom,left-top,left-bottom,right-top,right-bottom. | "wsqqhjwyuvdu" | Arrays.asList("wsqqhjwyuvds", "wsqqhjwyuveh", "wsqqhjwyuvdv", "wsqqhjwyuvdg", "wsqqhjwyuvdt", "wsqqhjwyuvde", "wsqqhjwyuvej", "wsqqhjwyuve5") |

1. **Test Implementation**

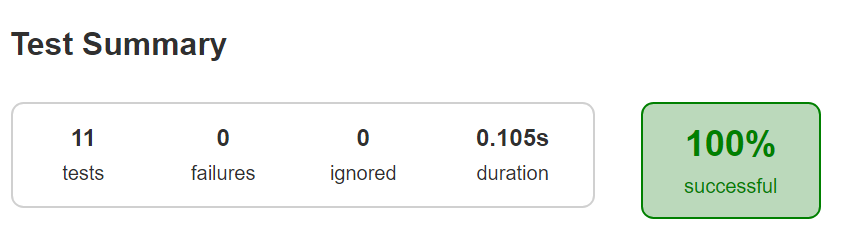
The design of test cases specified in Section 2 was implemented using JUnit 4. The test scripts of 3 selected test cases are given below. The rest of test script implementations can be found in the [link](https://github.com) (or JUnit files).

|  |  |  |
| --- | --- | --- |
| **No.** | **Test method** | **Source code** |
| 1 | +encodeBase32(long i,  int length):String |  |
| 2 | +encodeBase32(long i):String |  |
| 3 | +decodeBase32(String): long |  |
| 4 | +encodeHash(double,double):String |  |
| 5 | +encodeHash(double,double,int):String |  |
| 6 | + encodeHash(LatLong):String |  |
| 7 | + encodeHash(LatLong,int):String |  |
| 8 | +hashContains(String, double, double):boolean |  |
| 9 | +adjacentHash (String, Direction) :String |  |
| 10 | +adjacentHash (String, Direction,int) :String |  |
| 11 | +right(String):String |  |
| 12 | +bottom(String):String |  |
| 13 | +decodeHash(String):String |  |
| 14 | +hashLengthToCoverBoundingBox(double, double, double, double):int |  |
| 15 | +neighbours(String hash): List<String> |  |

1. **Test Results**
   1. **JUnit test result snapshot**

**一張含有 文字 的圖片

自動產生的描述**

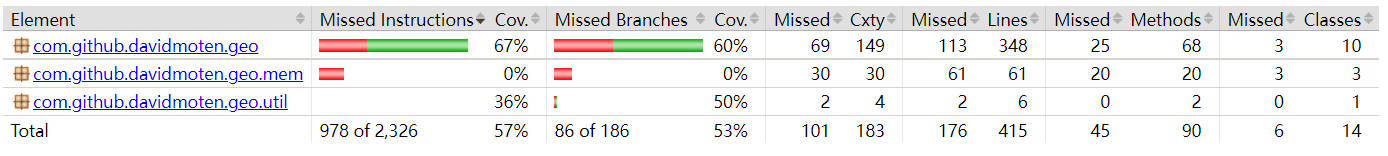
**   
一張含有 文字 的圖片

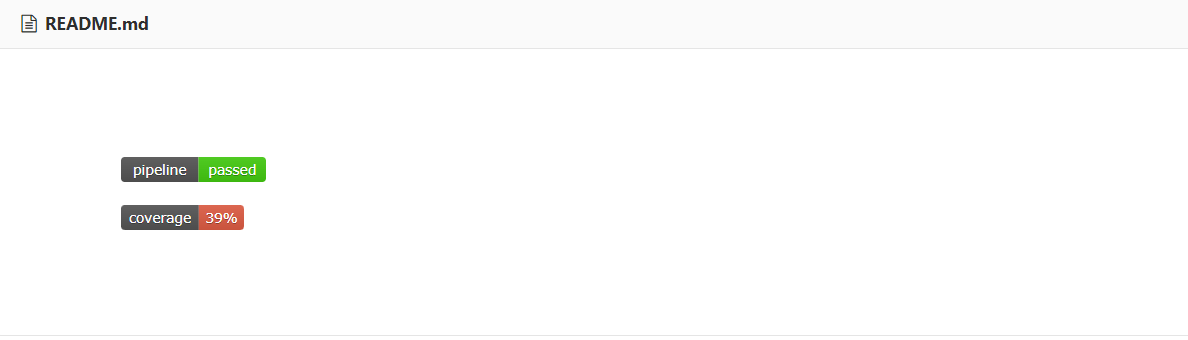
自動產生的描述**

* 1. **Code coverage snapshot**
* Coverage of each selected method

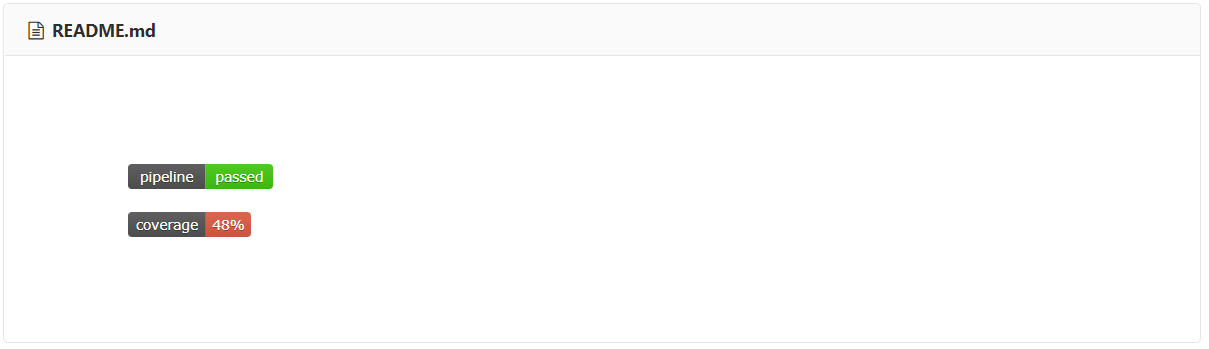
**一張含有 文字 的圖片

自動產生的描述**

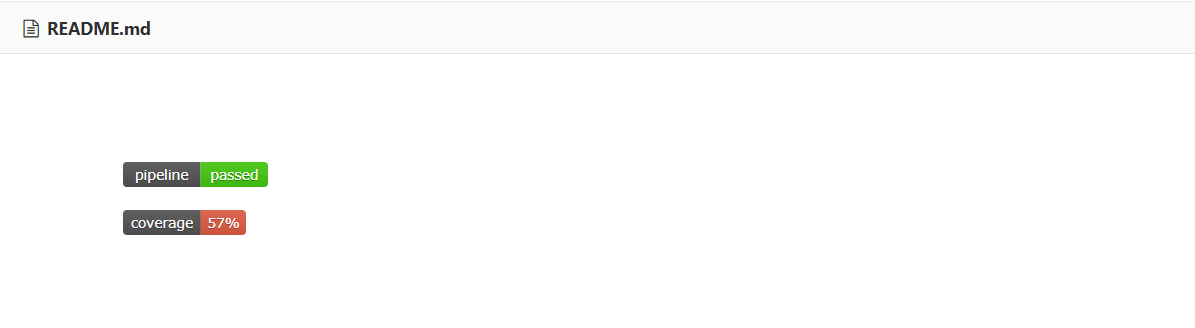
* Total coverage
  1. **CI result snapshot (3 iterations for CI)**
* CI#1

****

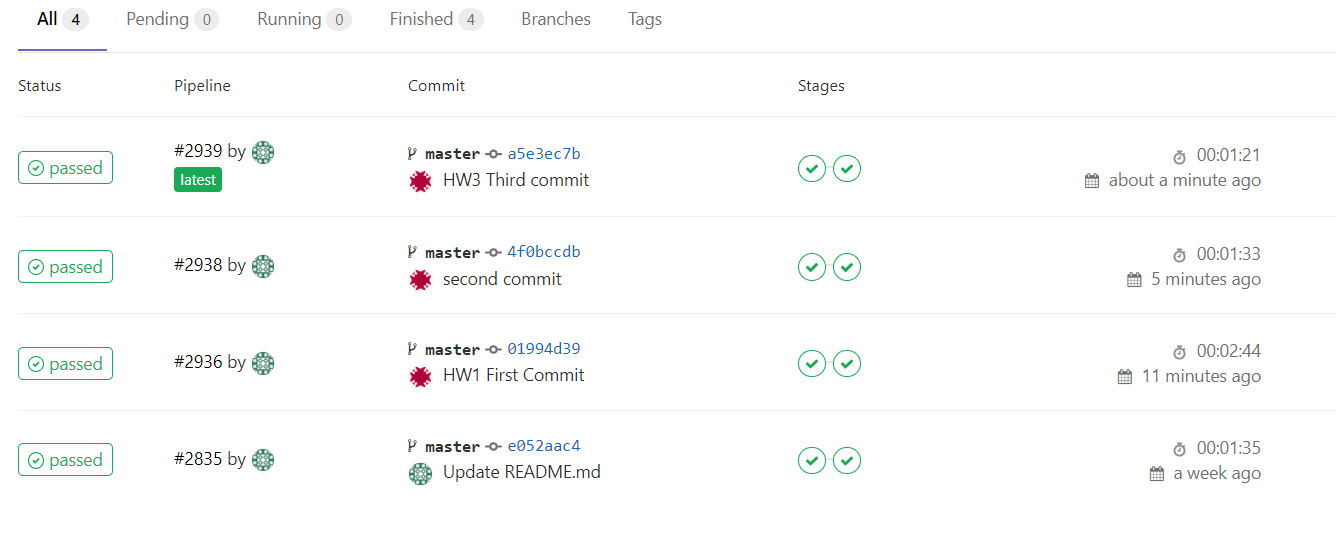
* CI#2



* CI#3

****

* CI Pipeline

****

1. **Summary**

In Lab 1, **15 test cases have been designed and implemented using JUnit**. The test is conducted in 3 CI and **the execution results of the 15 test methods are all passed**. **The total statement coverage of the test is 57%.** Thus, the test requirements described in Section 1 are satisfied.

為了能更完整的測出這個程式，在測試之前，也學習了GeoProject的使用方法和背景知識。這次作業完成而3次的CI，總共測15個methods，且**total statement coverage**為**57%，**符合此次作業要求。對於+hashContains(String, double, double):boolean，也增加了邊界測試。