

1. Describe the 5 key activities in an object-oriented design process.

- (a) Define the context and modes of use of the system
- (b) Design the system architecture
- (c) Identify the principal system objects
- (d) Develop design models
- (e) Specify object interfaces

2. Consider software evolution?

(a) What are the differences between refactor and reengineering?

Reengineering takes place after a system has been maintained for some time and maintenance costs are increasing. You use automated tools to process and reengineer a legacy system to create a new system that is more maintainable.

Refactoring is a continuous process of improvement throughout the development and evolution process. It is intended to avoid the structure and code degradation that increases the costs and difficulties of maintaining a system.

(b) Describe the tree types of software maintenances?

(1) Maintenance to repair software faults :

甲、Changing a system to correct deficiencies in the way meets its requirements.

乙、Corrective maintenance

(2) Maintenance to adapt software to a different operating environment :

甲、Changing a system so that it operates in a different environment (computer, OS, etc.) from its initial implementation.

乙、Adaptive maintenance

(3) Maintenance to add to or modify the system's functionality :

甲、Modifying the system to satisfy new requirements.

乙、Perfective maintenance

(c) What are bad smells (or code smells)? How will you relate bad smells and preventative maintenance?

甲、Duplicate code : The same or very similar code may be included at different places in a program. This can be removed and implemented as a single method or function that is called as required.

乙、Long methods : If a method is too long, it should be redesigned as a number of shorter methods.

丙、Switch (case) statements : These often involve duplication, where the switch depends on the type of a value. The switch statements may be scattered around a program. In object-oriented languages, you can often use polymorphism to achieve the same thing.

丁、Data clumping : Data clumps occur when the same group of data items (fields in classes, parameters in methods) re-occur in several places in a program. These can often be replaced with an object that encapsulates all of the data.

戊、Speculative generality : This occurs when developers include generality in a program in case it is required in the future. This can often simply be removed.

3. Consider a program that takes a numerical score (ranged from 0 to 100) and transfers the score to a letter grade A(score \geq 90), B(80 \leq score $<$ 90), C(70 \leq score $<$ 80), D(60 \leq score $<$ 70), or F(score $<$ 60); otherwise X(score $<$ 0 or score $>$ 100).

- (a) Apply the equivalence partitioning testing technique to design test cases for testing the program.

Score range	grade	Test case
score $>$ 100	X	Score = 101
score \geq 90	A	Score = 95
80 \leq score $<$ 90	B	Score = 85
70 \leq score $<$ 80	C	Score = 75
60 \leq score $<$ 70	D	Score = 65
score $<$ 60	F	Score = 55
score $<$ 0	X	Score = -5

- (b) Based on your answers in (a), design additional test cases by applying the boundary value analysis testing technique.

Score range	Test case	期望 output
score $>$ 100	Score = 101	X
	Score = 100	A
	Score = 99	A
score \geq 90	Score = 91	A
	Score = 90	A
	Score = 89	B

80<=score<90	Score = 91	A
	Score = 90	A
	Score = 89	B
	Score = 81	B
	Score = 80	B
	Score = 79	C
70<=score<80	Score = 81	B
	Score = 80	B
	Score = 79	C
	Score = 71	C
	Score = 70	C
	Score = 69	D
60<=score<70	Score = 71	C
	Score = 70	C
	Score = 69	D
	Score = 61	D
	Score = 60	D
	Score = 59	F
score<60	Score = 61	D
	Score = 60	D
	Score = 59	F
score<0	Score = 1	F
	Score = 0	F
	Score = -1	X

4. Consider the following program.

```

1  public static char letterGrade(int score) {
2      char grade;
3      if (score < 0 || score > 100)
4          grade = 'X';
5      else if (score >= 90 && score <= 100)
6          grade = 'A';
7      else if (score >= 80 && score < 90)
8          grade = 'B';
9      else if (score >= 70 && score < 80)
10         grade = 'C';
11     else if (score >= 60 && score < 70)
12         grade = 'D';
13     else
14         grade = 'F';
15     return grade;
16 }
```

- (a) Implement your test cases in problem 4(a) using JUnit. Show the JUnit source code of your test cases and the screen snapshots of the execution results of the test cases (including code coverage).

```

public static char letterGrade(int score) {
    char grade;
    if (score < 0 || score > 100) {
        grade = 'X';
    } else if (score >= 90 && score <= 100) {
        grade = 'A';
    } else if (score >= 80 && score < 90) {
        grade = 'B';
    } else if (score >= 70 && score < 80) {
        grade = 'C';
    } else if (score >= 60 && score < 70) {
        grade = 'D';
    } else {
        grade = 'F';
    }
    return grade;
}

```

```

public class letterGradeByPartitioningTest {
    @Test
    public void scoreIs101() {
        Assert.assertEquals('X', Grade.letterGrade(101));
    }
    @Test
    public void scoreIs95() {
        Assert.assertEquals('A', Grade.letterGrade(95));
    }
    @Test
    public void scoreIs85() {
        Assert.assertEquals('B', Grade.letterGrade(85));
    }
    @Test
    public void scoreIs75() {
        Assert.assertEquals('C', Grade.letterGrade(75));
    }
    @Test
    public void scoreIs65() {
        Assert.assertEquals('D', Grade.letterGrade(65));
    }
    @Test
    public void scoreIs55() {
        Assert.assertEquals('F', Grade.letterGrade(55));
    }
    @Test
    public void scoreIsMinus5() {
        Assert.assertEquals('X', Grade.letterGrade(-5));
    }
}

```

The screenshot shows an IDE with the following components:

- Package Explorer:** Shows the project structure with 'Partitioning' and 'letterGradeByPartitioningTest'.
- JUnit Console:** Displays the test results for 'letterGradeByPartitioningTest'. It shows 'Finished after 0.017 seconds', 'Runs: 7/7', 'Errors: 0', and 'Failures: 0'.
- Source Editor:** Displays the source code of 'letterGradeByPartitioningTest.java'.
- Problems/Console:** Shows the code coverage report for 'letterGradeByPartitioningTest (2017/12/29 上午 10:46:01)'.

Element	Coverage	Covered Instructi...	Missed Instructions	Total Instructions
SE_hw4	47.3 %	86	96	182
test	29.0 %	38	93	131
boundary	0.0 %	0	93	93
Partitioning	100.0 %	38	0	38
letterGradeByPartitioningTest.java	100.0 %	38	0	38
src	94.1 %	48	3	51

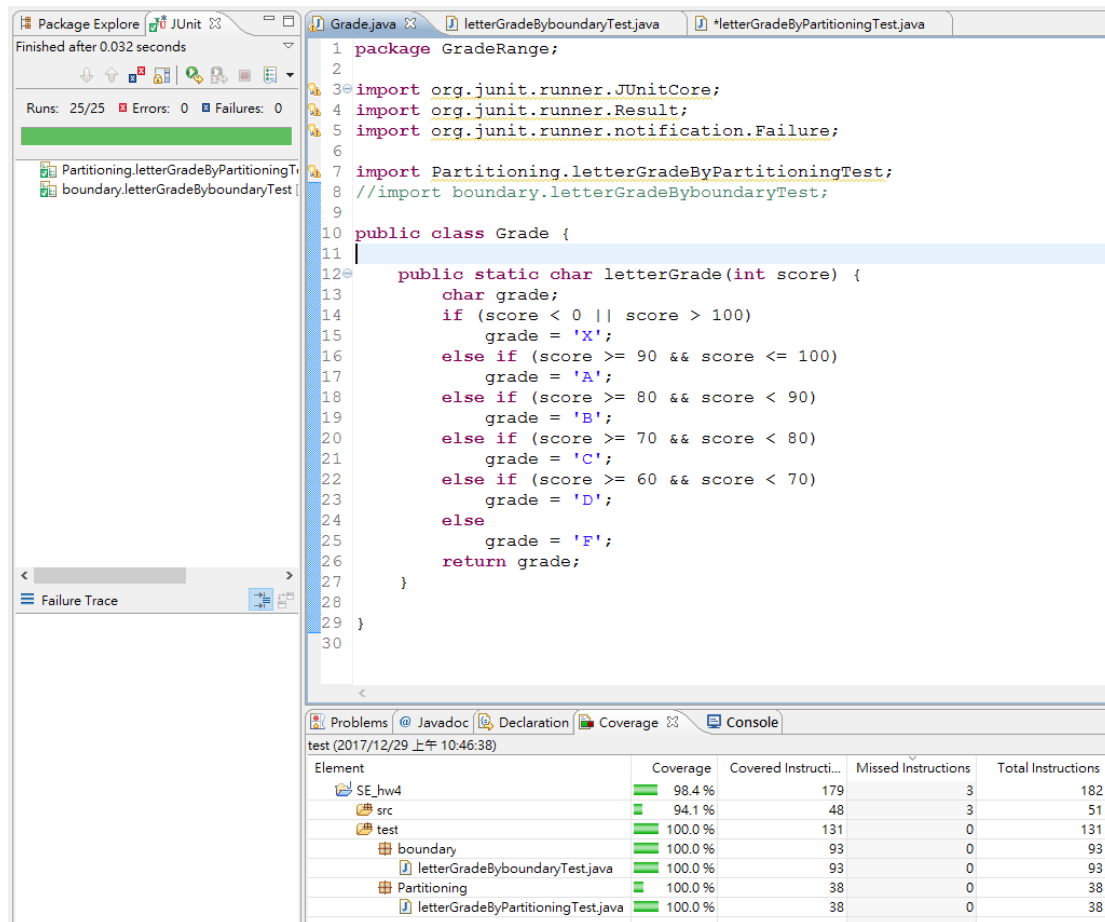
(b) Implement your test cases in problem 4(b) using JUnit. Show the JUnit source code of your test cases and the screen snapshots of the execution results of the test cases (including code coverage).

```
public class letterGradeByboundaryTest {
    @Test
    public void scoreIs101() {
        Assert.assertEquals('X', Grade.letterGrade(101));
    }
    @Test
    public void scoreIs100() {
        Assert.assertEquals('X', Grade.letterGrade(100));
    }
    @Test
    public void scoreIs99() {
        Assert.assertEquals('A', Grade.letterGrade(99));
    }
    @Test
    public void scoreIs91() {
        Assert.assertEquals('A', Grade.letterGrade(91));
    }
    @Test
    public void scoreIs90() {
        Assert.assertEquals('A', Grade.letterGrade(90));
    }
    @Test
    public void scoreIs89() {
        Assert.assertEquals('B', Grade.letterGrade(89));
    }
    @Test
    public void scoreIs81() {
        Assert.assertEquals('B', Grade.letterGrade(81));
    }
    @Test
    public void scoreIs80() {
        Assert.assertEquals('B', Grade.letterGrade(80));
    }
    @Test
    public void scoreIs79() {
        Assert.assertEquals('C', Grade.letterGrade(79));
    }
    @Test
    public void scoreIs71() {
        Assert.assertEquals('C', Grade.letterGrade(71));
    }
    @Test
    public void scoreIs70() {
        Assert.assertEquals('C', Grade.letterGrade(70));
    }
    @Test
    public void scoreIs69() {
        Assert.assertEquals('D', Grade.letterGrade(69));
    }
    @Test
    public void scoreIs61() {
        Assert.assertEquals('D', Grade.letterGrade(61));
    }
    @Test
    public void scoreIs60() {
        Assert.assertEquals('D', Grade.letterGrade(60));
    }
    @Test
    public void scoreIs59() {
        Assert.assertEquals('F', Grade.letterGrade(59));
    }
    @Test
    public void scoreIs1() {
        Assert.assertEquals('F', Grade.letterGrade(1));
    }
    @Test
    public void scoreIs0() {
        Assert.assertEquals('F', Grade.letterGrade(0));
    }
    @Test
    public void scoreIsMinus1() {
        Assert.assertEquals('X', Grade.letterGrade(-1));
    }
}
```

The screenshot shows an IDE with the JUnit test results and code coverage for the `letterGradeByboundaryTest` class. The test results panel on the left indicates that all tests passed successfully. The code coverage table at the bottom provides a detailed breakdown of the test results.

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
SE_hw4	77.5 %	141	41	182
test	71.0 %	93	38	131
Partitioning	0.0 %	0	38	38
boundary	100.0 %	93	0	93
letterGradeByboundaryTest.java	100.0 %	93	0	93
src	94.1 %	48	3	51

下圖為兩種測試方式都測的 code coverage



5. Illustrate the application of the JUnit and configuration management (CM) tools, such as subversion or Git (or GitHub), in software development. Note that you may integrate your IDE tool with your chosen CM tool, and you also need to create your own repository using the chosen CM tool.

- (a) Show the screen snapshots for using the CM tool to check in the source code of `letterGrade.java` and then check out the code to add a `main()` function so that the program can be executed and tested in console manually. After the manual testing is completed and the program is correct, commit the source code to the repository.

```

<terminated> Grade [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe
Enter a grade:-1
your letterGrade is : X
Enter a grade:50
your letterGrade is : F
Enter a grade:60
your letterGrade is : D
Enter a grade:70
your letterGrade is : C
Enter a grade:80
your letterGrade is : B
Enter a grade:90
your letterGrade is : A
Enter a grade:105
your letterGrade is : X
Enter a grade:exit
  
```

Package Explorer

- SE_hw4
 - src
 - GradeRange
 - Grade.java
 - LetterGrade.java
 - test
 - boundary
 - letterGradeByBoundaryTest.java
 - Partitioning
 - letterGradeByPartitioningTest.java
 - JRE System Library [JavaSE-1.6]
 - JUnit 4
 - 105598069_HW4.docx
 - 105598069_HW4.pdf
 - README.md

```

package GradeRange;

import java.io.IOException;

public class Grade {

    public static void main(String[] args) {

        // 給使用者輸入input
        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.print("Enter a grade:");
            String input = sc.nextLine();
            if (input.equals("exit")) {
                break;
            }
            int grade = Integer.parseInt(input);
            System.out.println("your letterGrade is : "
                + LetterGrade.letterGrade(grade));
        }

        sc.close();
    }
}

```

Element	Coverage	Covered Instructi...	Missed Instructions	Total Instructions
SE_hw4	35.5 %	81	147	228
test	0.0 %	0	141	141
src	93.1 %	81	6	87
GradeRange	93.1 %	81	6	87
Grade.java	91.7 %	33	3	36
LetterGrade.java	94.1 %	48	3	51

Commit to: master

Message:

1. 修改main() 可以手動輸入一筆grade值，印出你的letterGrade

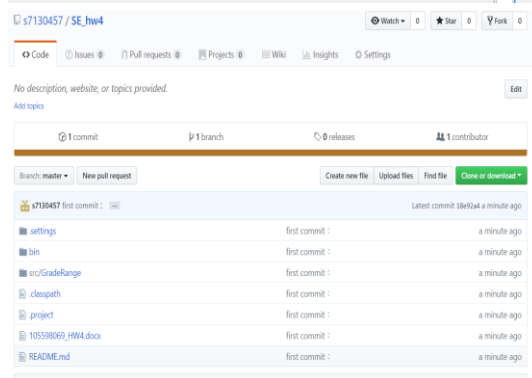
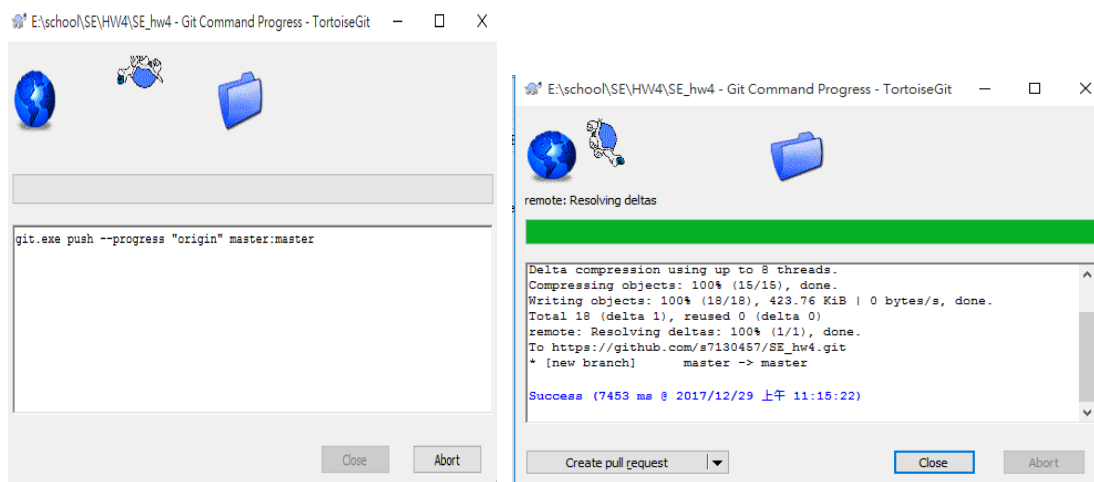
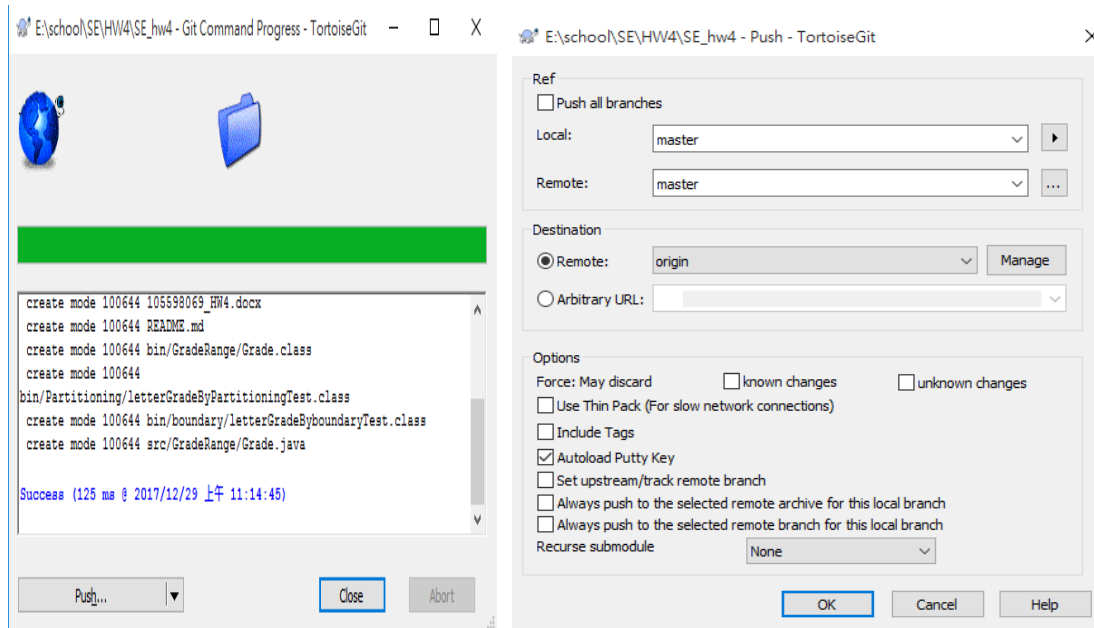
2. 修改read me

Changes made (double-click on file for diff):

Path	Extension	Status	Lines added	Lines removed
Not Versioned Files				
.classpath	.classpath	Unknown		
project	project	Unknown		
settings/org.eclipse.jdt.core.prefs	.prefs	Unknown		
105598069_HW4.docx	.docx	Unknown		
README.md	.md	Unknown		
bin/GradeRange/Grade.class	.class	Unknown		
bin/Partitioning/letterGradeByPartitioningTest.class	.class	Unknown		
bin/boundary/letterGradeByBoundaryTest.class	.class	Unknown		
src/GradeRange/Grade.java	.java	Unknown		

9 files selected, 9 files total

Commit



(b) Show the screen snapshots for using the CM and JUnit tools to check out your source code of letterGrade.java committed in 5(a) and add JUnit test cases to test the program automatically. After all the test cases are pass and the statement coverage is 100%, commit you source code and test cases to the repository.

Package Explorer

- SE_hw4
 - src
 - GradeRange
 - Grade.java
 - LetterGrade.java
 - test
 - boundary
 - letterGradeByBoundaryTest.java
 - Partitioning
 - letterGradeByPartitioningTest.java
 - JRE System Library [JavaSE-1.6]
 - JUnit 4
 - 105598069_HW4.docx
 - 105598069_HW4.pdf
 - README.md

```

package GradeRange;

import java.io.IOException;

public class Grade {
    public static void main(String[] args) {
        // 先跑全部的測試

        Result result = JUnitCore.runClasses(letterGradeByBoundaryTest.class,
            letterGradeByPartitioningTest.class);

        for (Failure failure : result.getFailures()) {
            System.out.println(failure.toString());
        }

        System.out.println(result.wasSuccessful());

        // 給使用者輸入input
        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.print("Enter a grade:");
            String input = sc.nextLine();
            if (input.equals("exit")) {
                break;
            }
            int grade = Integer.parseInt(input);
            System.out.println("your letterGrade is : "
                + LetterGrade.letterGrade(grade));
        }
        sc.close();
    }
}
  
```

Problems | Javadoc | Declaration | Console | Coverage

Grade (2017/12/29 下午 04:25:49)

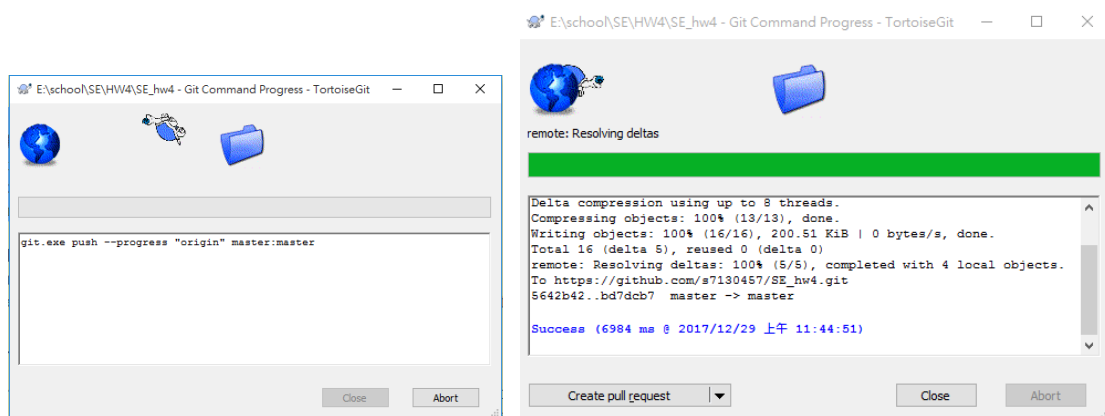
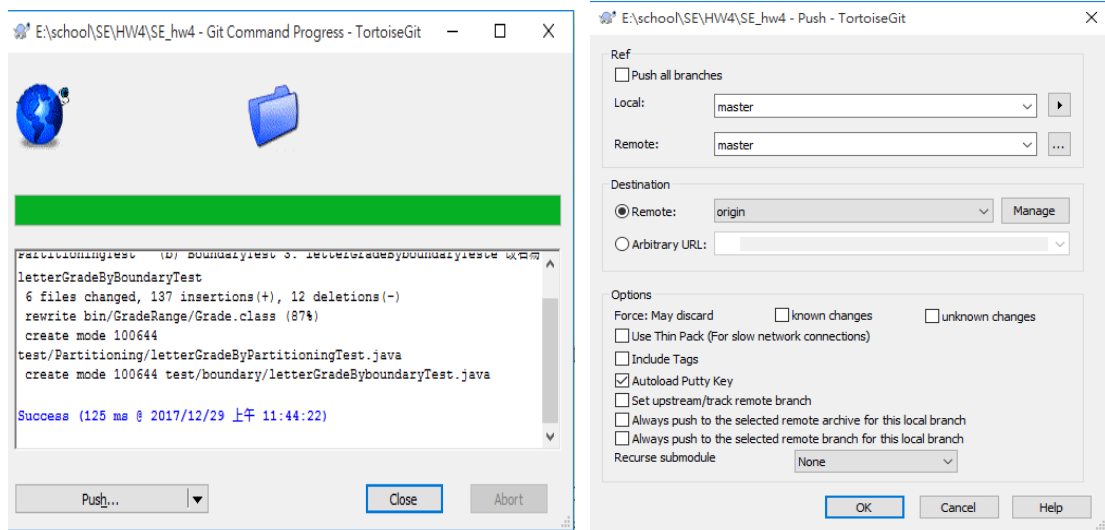
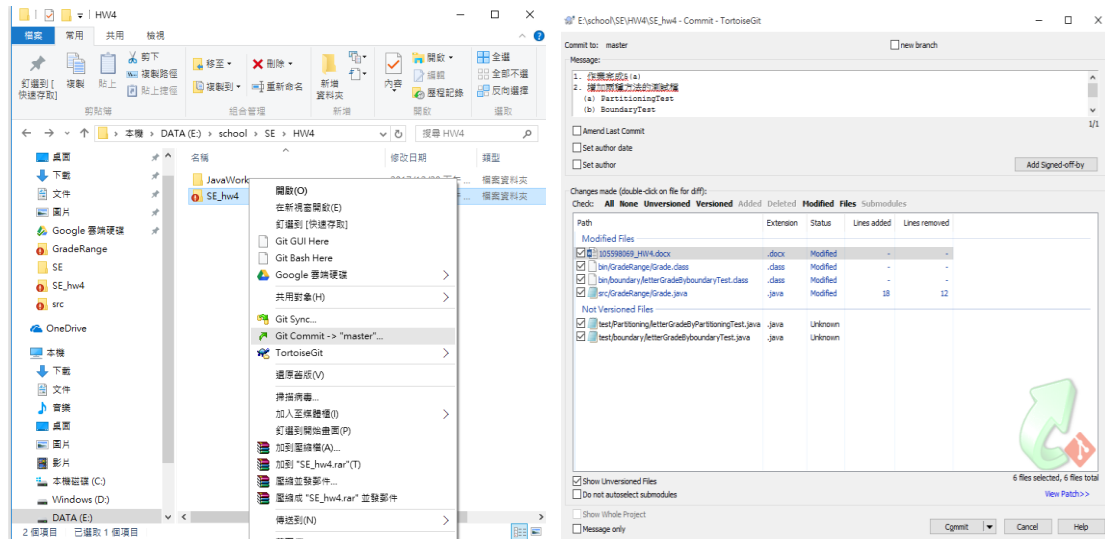
Element	Coverage	Covered Instructi...	Missed Instructions	Total Instructions
SE_hw4	95.8 %	249	11	260
src	90.8 %	108	11	119
GradeRange	90.8 %	108	11	119
Grade.java	83.8 %	57	11	68
LetterGrade.java	100.0 %	51	0	51
test	100.0 %	141	0	141
boundary	100.0 %	98	0	98
letterGradeByBoundaryTest.java	100.0 %	98	0	98
Partitioning	100.0 %	43	0	43
letterGradeByPartitioningTest.java	100.0 %	43	0	43

Problems | Javadoc | Declaration | Console | Coverage

<terminated> Grade [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (2017/12/29 下午 04:25:49)

```

Enter a grade:-1
your letterGrade is : X
Enter a grade:5
your letterGrade is : F
Enter a grade:60
your letterGrade is : D
Enter a grade:70
your letterGrade is : C
Enter a grade:80
your letterGrade is : B
Enter a grade:90
your letterGrade is : A
Enter a grade:105
your letterGrade is : X
Enter a grade:exit
  
```



s7130457 / SE_hw4

Watch 0

Star 0

Fork 0

Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings

No description, website, or topics provided.

Edit

Add topics

3 commits

1 branch

0 releases

1 contributor

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download

s7130457 1. 作業完成5(a) ...			Latest commit bd7dc7 a minute ago
.settings	first commit :	30 minutes ago	
bin	1. 作業完成5(a)	a minute ago	
src/GradeRange	1. 作業完成5(a)	a minute ago	
test	1. 作業完成5(a)	a minute ago	
.classpath	first commit :	30 minutes ago	
.project	first commit :	30 minutes ago	
105598069_HW4.docx	1. 作業完成5(a)	a minute ago	
README.md	first commit :	30 minutes ago	