



Software Testing HW#3

Student Name:
Pouya Haji Mohammadi Gohari

SID:810102113

Date of deadline
Monday 15th July, 2024

Dept. of Computer Engineering

University of Tehran

Contents

1	Question 1	3
1.1	Library In Impl1	4
1.2	Library of Impl2	7
2	Question 2	8
2.1	IMPL1	8
2.2	IMPL2	10

1 Question 1

For this question we are going to implement some property based on quick check tool. Each test case will generate randomly based on the test's inputs. We have obtained multiple properties from the provided specification's link.

Based on the following properties achieved by the specification, we will write some tests for each library implemented in impl1 and impl2 directories with maven.

Properties:

1. Each professor can borrow at most 5 documents.
2. For students, this number is 2.
3. Each document can have multiple copies therefore can be borrowed from multiple persons.
4. If all copies of same document were borrowed, no one can borrow this document.
5. Due time of returning a book is 10.
6. Due time of returning a reference is 5.
7. Due time of returning a magazine is 2.
8. Members of library can extend this duration at most 2 times if they borrowed either book or reference.
9. Magazines can not be extended.
10. Can not extend a document in the same day that was borrowed.
11. Can not extend a document that due time is passed.
12. For example, if a member borrows a regular book on 1/9/1399, the return deadline will be 11/9/1399. This means that if the book is returned on that day, no penalty will be applied. If the member extends the borrowing period on or before 11/9/1399, the new return deadline will be 21/9/1399.
13. Penalty of passing due time has been reported in table 1:

Reference Book	For the first 3 days overdue, the fine is 5000 tomans per day and for subsequent days, it is 7000 tomans per day (e.g., for 5 days overdue, the total fine will be 29000 tomans).
Regular Book	For the first 7 days overdue, the fine is 2000 tomans per day and for days 8 to 21, it is 3000 tomans per day and for subsequent days, it is 5000 tomans per day.
Magazine	For magazines published before 1390, the fine is 2000 tomans per day and for those published after 1390, it is 3000 tomans per day.

Table 1: Fine Schedule

14. Adding Student/Prof Member: If either member name or sid were empty, then exception must be thrown. Even if either student or professor with the same name will be expected to throw some exception.

15. Adding Book/Magazine/Reference: Duplicate name for documents are not allowed. If name is empty, it consider to throw an exception. For magazine year and number must be positive number.
16. Time Pass: Days must be non-negative number.

Now let's implement some tests for two following libraries.

1.1 Library In Impl1

Note that for each test, we will report the test-case's name and what property will be checked by it. Scenarios of testing are as follow:

- Valid Book/Reference/Magazine: Following three test cases will manage if document constructor of corresponding document will handle correctly. (This will handle Add Book/Magazine/Reference property and also due times [5, 6, 7])

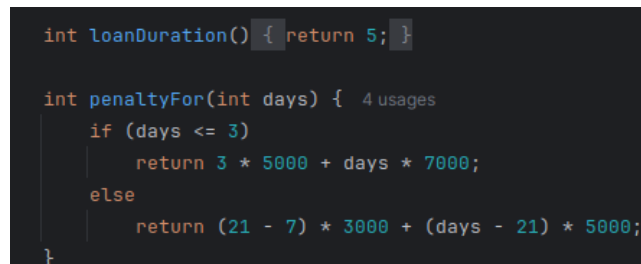
```
1 public void validBookTitle(String title);
2 public void validReference(String title);
3 public void validMagazine(String title, int yr, int num);
```

- Penalty Calculation Book/Reference/Magazine: We have implemented three test cases for checking the if each document will calculate the penalty correctly or not. (This is based on the table 1

```
1 public void testPenaltyCalculationBook(@InRange(min = "0", max = "35") int
   ↪ days);
2 public void testPenaltyCalculationReference(@InRange(min = "0", max = "35
   ↪ ") int days);
3 public void testPenaltyCalculationMagazine(@InRange(min = "-20", max = "
   ↪ 2000") int yr, int days){
```

Note that for not overflowing for large numbers of days, we consider to set a appropriate range. (It's like input partitioning the inputs if it would be correct for so many number in this range, therefore it would be correct for other values of days).

We have found a bug in penalty calculation for reference books. This bug is simply has been shown in the figure 1:



```
int loanDuration() { return 5; }

int penaltyFor(int days) { 4 usages
    if (days <= 3)
        return 3 * 5000 + days * 7000;
    else
        return (21 - 7) * 3000 + (days - 21) * 5000;
}
```

Figure 1: Bug Location

If late days is less than three, the penalty must be $5000 * \text{days}$ but it is obvious where it is not implemented correctly.

- Adding Student/Prof Member: In these test cases, we handle some exceptions defined in property(14) with following names:

```
1 public void testAddStudentMember(String studentId, String studentName);
2 public void testAddProfMember(String ProfName);
```

There is no bug for these two test cases.

- Borrow: For testing borrow method we have implemented two different test cases where property[1, 2, 4] are covered by first and property[3] are covered by second test.

```
1 public void testBurrow(String memberName, String documentTitle)
2 public void testBurrowMultipleMember(String memberName, String
    ↪ documentTitle);
```

There are no bugs for the borrow method in library.

- Extend: For testing this method, we utilize four test cases. First and second one will cover the property[10, 11] while third one will cover property[8] and the last one must cover the property[9].

```
1 public void testExtendBook(String memberName, String documentTitle, int
    ↪ copies, int days);
2 public void testExtendReference(String memberName, String documentTitle,
    ↪ int copies, int days);
3 public void testExtendMoreThanTwo(String memberName, String documentTitle
    ↪ , int copies, boolean bookReference);
4 public void testExtendMagazine(String memberName, String documentTitle,
    ↪ int copies, int yr, int num);
```

We have detected there would be no failure when extending a magazine. However, it was mentioned by specification that magazines can not be extended.

- Return Document: We just write one test to see if exception will throw correctly or not.

```
1 public void testReturnDocument(String memberName, String documentTitle);
```

No bug detected.

- Total Penalty for Book/Reference/Magazine: This test cases are different with those for first test cases(calculation penalty). Calculation penalty's test cases are for document classes while these tests are for library class. Corresponding to their due time of each document test cases can differ from one to another. These tests will again cover table 1.

```
1 public void testGetTotalPenaltyForBook(String memberName, @InRange(min = "
    ↪ 0", max = "50")int days, String title, int copies);
2 public void testGetTotalPenaltyForReference(String memberName, @InRange(
    ↪ min = "0", max = "50")int days, String title, int copies);
3 public void testGetTotalPenaltyForMagazine(String memberName, @InRange(min
    ↪ = "0", max = "50")int days, String title, int copies, int yr, int
    ↪ num);
```

As we know penalty for reference has a bug where we already saw it in the previous test cases.

- Available Titles: We have implemented a single test case for all possible variation of these where we consider to explain it all.

```
1 public void testAvailableTitles(Character c,@InRange(min = "0", max="3")
2     ↳ int length, String title, int copies){
3     assumeTrue(c!=null);
4     assumeTrue(!title.isEmpty());
5     assumeTrue(copies > 0);
6     Library library = new Library();
7     List<String> expected = new ArrayList<>();
8     try {
9         for(int i = 0; i < length; i++){
10             String repeatedC = String.join("", Collections.nCopies(i + 1,
11                 ↳ Character.toString(c)));
12             String temp = title + repeatedC;
13             library.addBook(temp, copies + i);
14             expected.add(temp);
15         }
16         List<String> actual = library.availableTitles();
17         assertEquals(length, actual.size());
18         boolean allSame = true;
19         for(int i = 0; i < length; i++){
20             if(!actual.contains(expected.get(i))){
21                 allSame = false;
22             }
23         }
24         assumeTrue(allSame);
25     }catch (InvalidArgumentEx | DuplicateDocumentEx e){
26         fail("Nothing must be thrown :)))) ");
27     }
```

Utilizing character c and string title, we can produce multiple document with temp. Temp string will simply concatenate i number of c to the title to have uniquely documents with different titles! Thus expected and actual list's size would be match. Also if all of the contents of expected list was in actual list, therefore the test will pass. This test case will pass also for variate number and string as input.

- Time Pass: For this method, we consider to check either if throws correct exception or not. The test's name is:

```
1 public void testTimePass(int days);
```

In summary, from 23 test cases implemented, 20 was passed and 3 failed. Two out of three failed tests is due to wrong implementation of calculation of penalty for reference class. Third failed test is caused by ignoring to throw an exception for extending magazine document.

1.2 Library of Impl2

The test cases are similar to previous sections with respect to this library.

Note that from 21 test cases just 11 passed while 10 were failed. Some of failed tests are categorized as follow:

1. Empty prof name was not handling correctly.
2. Empty title for Book/Reference/Magazine was not correct.
3. Extend on the same day for all documents was allowed whereas it must be forbidden.
4. Extend twice was allowed ignoring property 8/
5. Magazine can be extended where should not.
6. This library will calculate the penalty for book wrongly.

Note that some tests could not see all the possible for wrong implementation of empty string. Thus we consider to create second edition for some tests. (e.g. add book/reference/magazine/prof member)

The second library has so many more bugs than first implementation.

2 Question 2

2.1 IMPL1

We will evosuite tool for generating some basic tests for us.(It is useful for regression testing and we should complete some of them).We firstly install the evosuite from the provided link and will compile every class with maven.The binary files will executed in target classes.Then from following command we can generate some tests for specific class:

```
1 java -jar ~/evosuite/evosuite-1.2.0.jar -projectCP target/classes -prefix ir.  
   ↪ ramtung.impl1
```

Now from entering this command, we will wait till the evosuite will generate tests for each class in targets/class/impl1 and write it in evosuite-tests directory. Then we will copy them into src/test/java/ir/ramtung to the corresponding implementation.Since all tests will be designed in a way where all of them would pass(good for regression testing). We will change some incomplete tests where can detect failures.

Test Cases changing: From previous sections we have detected 2 main bugs where in Reference class the penalty was incorrectly implemented and extending magazine was doable but the property said is not possible.We will detect each test for each class one by one to detect more bugs.

Test 1: Negative day pass to penalty is not expected and must return 0 but we must change the following test:(test5 for book class)

```
1 @Test(timeout = 4000)  
2 public void test5() throws Throwable {  
3     Book book0 = new Book("c4|cRz10a");  
4     int int0 = book0.penaltyFor((-601));  
5     assertEquals((-1202000), int0);  
6 }
```

Change to:

```
1 @Test(timeout = 4000)  
2 public void test5() throws Throwable {  
3     Book book0 = new Book("c4|cRz10a");  
4     int int0 = book0.penaltyFor((-601));  
5     assertEquals(0, int0);  
6 }
```

Yet there were one bugs!Now lets proceed into Magazine class.

Test 2:The day if is negative number there would no penalty since negative numbers has no meaning to day.

```
1 @Test(timeout = 4000)  
2 public void test4() throws Throwable {  
3     Magazine magazine0 = new Magazine("hK]^W5PI)Jjk,zj", 5451, 5451);  
4     int int0 = magazine0.penaltyFor((-1));  
5     assertEquals((-3000), int0);  
6 }
```

Change to:


```

1  @Test(timeout = 4000)
2  public void test4() throws Throwable {
3      Magazine magazine0 = new Magazine("hK]^W5PI)Jjk,zj", 5451, 5451);
4      int int0 = magazine0.penaltyFor((-1));
5      assertEquals(0, int0);
6  }

```

Test 3:(Test0 for Reference class)This test is not correct since the penalty is calculated wrongly:

```

1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      Reference reference0 = new Reference("p<N(HTW");
4      int int0 = reference0.penaltyFor(3);
5      assertEquals(36000, int0);
6  }

```

Change to 15000 as expected:

```

1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      Reference reference0 = new Reference("p<N(HTW");
4      int int0 = reference0.penaltyFor(3);
5      assertEquals(15000, int0);
6  }

```

Test 4:(Test2 for Reference class)This test when 0 days has passed, expected to have no penalty.

```

1  @Test(timeout = 4000)
2  public void test2() throws Throwable {
3      Reference reference0 = new Reference("p<N(HTW");
4      int int0 = reference0.penaltyFor(0);
5      assertEquals(15000, int0);
6  }

```

Change to expected zero:

```

1  @Test(timeout = 4000)
2  public void test2() throws Throwable {
3      Reference reference0 = new Reference("p<N(HTW");
4      int int0 = reference0.penaltyFor(0);
5      assertEquals(15000, int0);
6  }

```

Test 5:(Test3 for Reference Class)This test for 5 days penalty should be 29000 toman but wrongly set:

```

1  @Test(timeout = 4000)
2  public void test3() throws Throwable {
3      Reference reference0 = new Reference((String) null);
4      int int0 = reference0.penaltyFor(5);
5      assertEquals((-38000), int0);
6  }

```

Change to expected 29000:

```
1  @Test(timeout = 4000)
2  public void test3() throws Throwable {
3      Reference reference0 = new Reference((String) null);
4      int int0 = reference0.penaltyFor(5);
5      assertEquals(0, int0);
6  }
```

From 85 test generated with evosuite only 5 was failed. **Attention:** If null string is not consider an empty therefore all remain tests seems to be true otherwise they are few tests that must throw and exception where create an object with null string.

All remaining tests seems to correct but need to investigate more.(Personally, I did not notice something special) Now lets see the buggy implementation(Second One).

2.2 IMPL2

Test 1:(Test 1 of Book) penalty for negative days must be zero(since it has no other meaning).

```
1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      Book book0 = new Book("goA&\"F&S", (-20));
4      int int0 = book0.calculatePenalty((-21));
5      assertEquals(3000, int0);
6      assertEquals(10, book0.getDay());
7  }
```

Change to:

```
1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      Book book0 = new Book("goA&\"F&S", (-20));
4      int int0 = book0.calculatePenalty((-21));
5      assertEquals(0, int0);
6      assertEquals(10, book0.getDay());
7  }
```

Test 2: (Test 1 for Book) copies must not be negative.(Also empty book title)

```
1  @Test(timeout = 4000)
2  public void test1() throws Throwable {
3      Book book0 = new Book("", (-1));
4      int int0 = book0.calculatePenalty((-7));
5      assertEquals(10, book0.getDay());
6      assertEquals(2000, int0);
7  }
```

Change to:

```
1  @Test(timeout = 4000)
2  public void test1() throws Throwable {
```

```

3      try {
4          Book book0 = new Book("", (-1));
5          fail("Expected to throw something");
6      } catch (Exception e){
7          verifyException("ir.ramtung.impl2.Book", e);
8      }
9  }

```

Also test 2 or 3 or 4 must be changed with same issue.(For Book Class)

Test 6(Test0 for Document) Must changed since the number is negative:

```

1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      Magazine magazine0 = new Magazine((String) null, 0, (-1434), 0);
4      magazine0.getTitle();
5      assertEquals(0, magazine0.getCopies());
6  }

```

Change to:

```

1  @Test(timeout = 4000)
2  public void test0() throws Throwable {
3      try {
4          Magazine magazine0 = new Magazine((String) null, 0, (-1434), 0);
5          fail("Expected to throw something");
6      } catch (Exception e){
7          verifyException("ir.ramtung.impl2.Magazine", e);
8      }
9  }

```

Test 7:(Test 3 for Document) Has already same issue with the previous test.

Test 8:(Test 4 for Document) Has same issue.

Test 9:(Test 5 for Document) Must not borrow twice since has no copy left:

```

1  @Test(timeout = 4000)
2  public void test5() throws Throwable {
3      Magazine magazine0 = new Magazine("w7", 1, 1, 1);
4      magazine0.borrowBook();
5      magazine0.borrowBook();
6      int int0 = magazine0.getCopies();
7      assertEquals((-1), int0);
8  }

```

Change to:

```

1  @Test(timeout = 4000)
2  public void test5() throws Throwable {
3      Magazine magazine0 = new Magazine("w7", 1, 1, 1);
4      magazine0.borrowBook();
5      try {
6          magazine0.borrowBook();

```

```

7         fail("Copies out of run exception");
8     }catch (Exception e){
9         verifyException("ir.ramtung.impl2.Magazine", e);
10    }
11 }

```

Test 10:(Test 01 for Library) Adding reference with negative copies must throw an exception.

```

1  @Test(timeout = 4000)
2  public void test01() throws Throwable {
3      Library library0 = new Library();
4      library0.addReference("EC/7bne", (-1553));
5      List<String> list0 = library0.availableTitles();
6      assertFalse(list0.contains("EC/7bne"));
7  }

```

Change to:

```

1  @Test(timeout = 4000)
2  public void test01() throws Throwable {
3      Library library0 = new Library();
4      try{
5          library0.addReference("EC/7bne", (-1553));
6          fail("Negative copies must throw and exception");
7      }catch (Exception e){
8          verifyException("ir.ramtung.impl2.Library", e);
9      }
10     List<String> list0 = library0.availableTitles();
11     assertFalse(list0.contains("EC/7bne"));
12 }

```

Test 11:(Test 03 for Library) Can not add reference with empty string.

From now on we will say just what tests having problem but won't change them since there are so many bugs exist in the code thus we consider to not change the tests since we can correct the code first then generate some test cases with evosuite.

Test 12: (Test 04 Library) Empty title detected but no exception handled.

Test 13: (Test 05 Library) Negative copies for adding a reference.

Test 14: (Test 08 Library) Number for magazine is zero without any exception.(Copies is zero) Also penalty is not calculate correctly since must expect 3665000.

Test 15: (Test 10 Library)Again negative copies detected without any exception.

Test 16: (Test 21 Library) No document is available in return document method and there is no exception for that.

Test 17: (Test 22 Library) Same as previous test.

Test 18: (Test 23 Library) Can not extend a nonexistence document!Even if exists can not extend on the same day.

Test 19: (Test 26 Library) Title is not empty why it should throw and exception? Instead negative copeis must be thrown.

We have already checked around 30 or 40 test cases and about 20 of them was wrong.Thus we will avoid to continue since if a developer face with this situation he would consider to change the code instead of changing all test cases.After correcting code he/she would again generate the test cases with evosuite tool.

For a code that can potentially has so many bug evosuite will generate bad tests as we can see in the library implemented in impl2. However if code has fewer bug, evosuite will help us to create a powerful oracle for testing as we sat for the library implemented in impl1.