National College of Ireland

Software Quality and Testing

Semester 2,

CA3 SAMPLE

Notes:

- This is an **open book assessment.** Any resources used in answering the questions should be referenced in the submitted document, and listed in the *References* section.
- This is an **individual assessment.** All work submitted **should be your own** and should be carried out using only **concepts covered in this module**. Conferring with others **is not permitted**.
- The responses should include all the intermediate calculations for obtaining the answer.
- **YOU ARE NOT ALLOWED TO PUBLISH THIS ASSIGNMENT BRIEF OR A PART THEREOF ON ANY WEBSITES. YOU ARE NOT ALLOWED TO PUBLISH/SHARE YOUR SOLUTION WITH OTHERS.**
- You should submit your answers electronically in a .docx or .pdf document via the *CA3 Submission Link* available on the module's Moodle page. You are also required to submit all the .java files corresponding to the implementation of your unit tests and functionalities tested.

- **Attachments** Java source code for Circle and TextComparator instantiable classes are available on our Moodle page in the files *Circle.java* and *TextComparator.java*.

National College of Ireland

Software Quality and Testing
Continuous Assessment 3
Unit Testing

As part of the continuous assessment for the Software Quality and Testing module, you are required to answers the questions below as part of the lab session, and submit the answers electronically *in a .docx or .pdf document* via the *CA3 Submission Link* available on the module's Moodle page. You are also required to submit all the .java files corresponding to the implementation of your unit tests and functionalities tested.

Note: • You have to submit both a report with all the answers to the questions below and all the .java files corresponding to the implementation of your unit tests and functionalities tested. **This is a submission requirement.**

For the questions below you are required to implement and execute JUnit unit tests in NetBeans IDE.

Question 1 Let us consider an instantiable class that contains a compute method that calculates the area of a circle, and another compute method that calculates the perimeter of a circle given the circle radius. The corresponding source code is presented in Figure 1:

```
3
4  public class Circle {
5
6      public double calculateArea(double radius){
7          double PI = 3.14;
8          double area;
9          // compute the area of a circle
10         area = PI * radius * radius;
11         return area;
12     }
13
14     public double calculatePerimeter(double radius){
15         double PI = 3.14;
16         double perimeter;
17         // compute the perimter of a circle
18         perimeter = PI * 2 * radius;
19         return perimeter;
20     }
21 }
```

Figure 1 Source code for Question 1

a) Implement all the required unit tests to check if the compute methods work according to the expectations
b) Run/execute the unit tests implemented at item *a)* and provide in your report a screenshot of the NetBeans *Test Results Window* that includes a summary of the execution of the unit tests (namely, for each unit test, the name of the test method run and whether the test has passed or failed).

[40 marks]

Continues on next page

Question 2 Let us consider an instantiable class that contains a compute method. The compute method should work according to the following specification: the compute method should check if two pieces of text (i.e. Strings[1]) contain the same sequence of characters without taking into consideration how the text is written (namely, using upper case letters, or lower case letters, or a combination of both upper case and lower case letters). When the two pieces of text contain the same sequence of characters (ignoring the letter case) the method should return the value true. Otherwise, when the two pieces of text contain a different sequence of characters the method should return the value false. Figure 2 presents a possible implementation.

```
3
4   public class TextComparator {
5
6       public boolean compare(String aText, String anotherText){
7           boolean sameContent;
8           if (aText.equals(anotherText)){
9               sameContent = true;
10      } else {
11              sameContent = false;
12          }
13          return sameContent;
14      }
15
16  }
17
```

Figure 2 Source code for Question 2

a) Implement all the required unit tests in order to achieve full code coverage according to the branch coverage technique.

b) Run/execute the unit tests implemented at item *a)* and provide in your report a screenshot of the NetBeans *Test Results Window* that includes a summary of the execution of the unit tests (namely, for each unit test, the name of the test method run and whether the test has passed or failed).

c) Implement all the required unit tests in order to check whether the compute method works according to expectation, namely according to the specification provided in *Question 2*. In case that there is any incorrect code/logic in the compute method, your unit test(s) should uncover such defects/bugs.

d) Run/execute the unit tests implemented at item *c)* and provide in your report a screenshot of the NetBeans *Test Results Window* that includes a summary of the execution of the unit tests (namely, for each unit test, the name of the test method run and whether the test has passed or failed).

e) If there are any defects/bugs uncovered by the unit test(s) implemented at item *c)* and run/executed at item *d)*

   i. First, fix the implementation of the compute method, i.e. update the compute method with the correct logic/code to remove any bugs/defects uncovered by the unit test(s) implemented at item *c)*. Include the latest version of your compute method in the report.

   ii. Second, run/execute the unit tests implemented at item *c)* and provide in your report a screenshot of the *Test Results Window* that includes a summary of the execution of the unit tests (namely, for each unit test, the name of the test method run and whether the test has passed or failed).

[60 marks]

[1] https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html