**CS3343 Software Engineering Practice**

**Semester A, 2019/20**

**Test One**

Total time allowed: 90 minutes

**Student Name: \_ LOW Zhi Hao \_**

**Student Number: \_ 54924670 \_**

**Tutorial Session: \_ LAB1 \_**

MARK:

|  |
| --- |
| **Download the application code and the test template at the following URL according to the instruction given to you:**  **Version 1: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V1.zip**  **Version 2: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V2.zip**  **Version 3: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V3.zip**  **Version 4: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V4.zip**  **Version 5: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V5.zip**  **Version 6: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V6.zip**  **Version 7: http://cs3343.cs.cityu.edu.hk/2019\_TestOne/CS3343\_Test1\_V7.zip** |

|  |
| --- |
| **Following the instruction to open the Eclipse for CS3343 to complete this test:**  The customized Eclipse is available on each CSLab machine (See the following picture) |
| **SUBMISSION PROCEDURE (SUBMIT THIS WORD DOCUMENT):**  **NOTE:**  **LAB1 refers to CS3343-L01 0900-1100 FRI**  **LAB2 refers to CS3343-L02 1300-1500 FRI**  **LAB3 refers to CS3343-LA1 1900-2100 FRI**  **IMPORTANT: Follow the three-step procedure below to submit your solution document.**  Step 1: **Save** your solution file in **your local machine.** Use **Save As** to change the authors of the document to yourself when saving.  Step 2: **Rename** your solution document as **your student number**. E.g., if your student name is 51234567, then save your document as “51234567.docx”.  Step 3: Submit your solution document on Canvas.  ***Attention:* YOU MUST keep one backup of your solution document**. It is required for the assessment in the case that your submission file in Canvas system has been corrupted.  **LAB1, LAB2 and LAB3 students:**   * **Canvas > 201909CS3343 > Assignments > Test1 LAB1 (9AM)**for **LAB1** students completed the test in the **Friday 09:00-10:50 session** * **Canvas > 201909CS3343 > Assignments > Test1 LAB2 (1PM)** for **LAB2** students completed the test in the **Friday 13:00-14:50 session** * **Canvas > 201909CS3343 > Assignments > Test1 LAB3 (7PM)** for **LAB3** students completed the test in the **Friday 19:00-20:50 session**     **Note the following:**   * Any file not within your designated tutorial session (according to your official time table) will not be considered. * For each student, if you have multiple submissions, only the latest one will be assessed. |

|  |
| --- |
| **How Will Your Answer Be Scored?**   * **For Questions 1, 2, 3 and 4, only those codes that can be run successfully may receive marks.** * **For Questions 1, 2 and 3, writing an excessive number of test cases will receive no mark.** * The Maximum mark is 100. |

1. **Read your downloaded code listing, and complete all of the following questions: (80%)**

Assuming that you are given a program version ***m*** (where *m* ∈ {1, 2, 3, 4, 5, 6, 7}).

(Suppose that your version number ***m*** is **1**. In this case, the following **BuyerV*m*** represents **BuyerV*1***; **CategoryV*m*** represents **CategoryV*1***; and **SalesItemV*m*** represents **SalesItemV*1***).

Two folders are in the downloaded code list: **salesV*m***and **testSalesV*m***. In the folder **salesV*m***, there are classes: **BuyerV*m***, **CategoryV*m***, and **SalesItemV*m***. In the folder **testSalesV*m***, there is only one test case file **TestSalesItemV*m*.java**. The class **SalesItemV*m*** will calculate the discount rate of a specified item via the function **getDiscountRate(CategoryV*m*, int)**. Read the code and write your test cases in the file **TestSalesItemV*m*.java** according to the following questions.

1. Write **six (6)** *unit* test cases for **SalesItemV*m*.getDiscountRate(CategoryV*m*, int)** such that each test case goes through *a unique path* without executing the statements labeled as **L1, L2** and **T1**. **(24%)**
2. Write **two (2)** *unit* test cases for **SalesItemV*m*.getDiscountRate(CategoryV*m*, int)** such that each test case goes through *a unique path with* executing the statementlabeled as **T1**, and yet the test case invokes your test stub of the **BuyerV*m*.isBirthdayThisWeek ()**. **(16%)**

* Note that you need to write your own test stub for these two test cases.

1. Write **two (2)** *unit* test cases for **SalesItemV*m*.getDiscountRate(CategoryV*m*, int)** such that each test case goes through *a unique path with* executing the statementlabeled as **T1**, and yet the test case invokes **BuyerV*m*.isBirthdayThisWeek ()**. **(8%)**

* Note that do not use your test stub for these two test cases.

1. There are Code Smells in function **getDiscountRate**(**CategoryV*m*, int**), perform code refactoring and your final code must keep the same interface as the function **getDiscountRate**(**CategoryV*m*, int**). You may consider using the “**Extract Method**” code refactoring strategy to accomplish this task. **(25%)**

* This question involves changes in Classes: **CategoryV*m*.java** and **SalesItemV*m*.java**
* Please copy/paste your entire class codes of **CategoryV*m*.java** and **SalesItemV*m*.java** to the solution box.

1. Capture the fully expanded **statement coverage screen** generated from your test cases after refactoring completed in Q4. The screenshot should clearly show the coverage measures of the classes: **BuyerV*m***, **CategoryV*m*** and **SalesItemV*m***. **(7%)**
2. **Critical Path Analysis (20%)**
3. Find the critical path for the following activities. Show the earliest event time and latest event time for each activity.

Critical Path Analysis of Computer Project

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Activity** | **Immediately Preceding Activities** | **Duration** |
| A | High Level Analysis |  | 1 |
| B | Select Hardware |  | 1 |
| C | Core Module Analysis | A | 2 |
| D | Install and Commission Hardware | A | 2 |
| E | Core Module Programming | B | 1 |
| F | Supporting Module Analysis | C | 2 |
| G | Supporting Models Programming | C | 2 |
| H | Dev and QA Management Reports | D | 1 |
| I | QA of Supporting Modules | G, H | 3 |
| J | Dev MIS | E | 3 |
| K | Core Module Training | I, F | 1 |
| L | Detailed Training | J, K | 1 |

**G(2)**

**A(1)**

**F(2)**

**B(1)**

**I(3)**

**K(1)**

**D(2)**

**E(1)**

**H(1)**

**C(2)**

**J(3)**

**L(1)**

− END OF QUESTIONS−

**Student Answer:**

***Attention:* Don’t paste the screenshot** for Q1−Q4. **Place in the editable code listing in the box provided.**

1. Copy and paste the test scripts for **Question 1** in the following box: (24%)

|  |
| --- |
| @Test  **public** **void** testQ1\_1() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BEVERAGE***;  **double** result = salesItem.getDiscountRate(cat, 0);  *assertEquals*(-1.00, result, 0);  }  @Test  **public** **void** testQ1\_2() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BOOK***;  **double** result = salesItem.getDiscountRate(cat, 1);  *assertEquals*(0.90, result, 0);  }  @Test  **public** **void** testQ1\_3() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***FOOD***;  **double** result = salesItem.getDiscountRate(cat, 11);  *assertEquals*(0.90, result, 0);  }  @Test  **public** **void** testQ1\_4() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***FOOD***;  **double** result = salesItem.getDiscountRate(cat, 2);  *assertEquals*(1.0, result, 0);  }  @Test  **public** **void** testQ1\_5() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BEVERAGE***;  **double** result = salesItem.getDiscountRate(cat, 4);  *assertEquals*(0.95, result, 0);  }  @Test  **public** **void** testQ1\_6() {  BuyerV1 buyer = **new** BuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***OTHER***;  **double** result = salesItem.getDiscountRate(cat, 4);  *assertEquals*(1.00, result, 0);  } |

1. Copy and paste the test scripts for **Question 2** in the following box: (16%)

|  |
| --- |
| // Add your answer to the two test cases for Q2  @Test  **public** **void** testQ2\_1() {  **class** StubBuyerV1 **extends** BuyerV1{  //Overwrite the class functions  **public** **boolean** isBirthdayThisWeek (){  **return** **true**;  }  }  BuyerV1 buyer = **new** StubBuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BOOK***;  **double** result = salesItem.getDiscountRate(cat, 6);  *assertEquals*(0.50\*0.90, result, 0.000001);  }  @Test  **public** **void** testQ2\_2() {  **class** StubBuyerV1 **extends** BuyerV1{  //Overwrite the class functions  **public** **boolean** isBirthdayThisWeek (){  **return** **true**;  }  }  BuyerV1 buyer = **new** StubBuyerV1();  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BOOK***;  **double** result = salesItem.getDiscountRate(cat, 4);  *assertEquals*(0.85\*0.90, result, 0.000001);  } |

1. Copy and paste the test scripts for **Question 3** in the following box: (8%)

|  |
| --- |
| // Add your answer to the two test cases for Q3  @Test  **public** **void** testQ3\_1() {  BuyerV1 buyer = **new** BuyerV1("Jackson", 12, 10);  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BOOK***;  **double** result = salesItem.getDiscountRate(cat, 6);  *assertEquals*(0.50\*0.90, result, 0.000001);  }  @Test  **public** **void** testQ3\_2() {  BuyerV1 buyer = **new** BuyerV1("Jackson", 12, 10);  SalesItemV1 salesItem = **new** SalesItemV1(buyer);  CategoryV1 cat = CategoryV1.***BOOK***;  **double** result = salesItem.getDiscountRate(cat, 4);  *assertEquals*(0.85\*0.90, result, 0.000001);  } |

1. Copy and paste the **refactored code scripts** for **Question 4** in the following box: (25%)

|  |
| --- |
| **package** salesV1;  **public** **enum** CategoryV1 {  ***BOOK***,  ***FOOD***,  ***BEVERAGE***,  ***OTHER***;  **public** **double** getRate(**int** item\_quan) {  **double** rate;  **if** (**this** == CategoryV1.***BOOK***) {  rate = 0.90;  } **else** **if** (**this** == CategoryV1.***FOOD***) {  **if** (item\_quan > 10) {  rate = 0.90;  } **else** **if** (item\_quan > 3) {  rate = 0.95; // This statement is labeled as L1  } **else** {  rate = 1.0;  }  } **else** **if** (**this** == CategoryV1.***BEVERAGE***) {  **if** (item\_quan > 6) {  rate = 0.85; // This statement is labeled as L2  } **else** {  rate = 0.95;  }  } **else** {  rate = 1.00;  }  **return** rate;  }  } |

1. Copy and paste the **screenshot** of statement coverage (**AFTER Refactoring)** in the following box: (7%)

|  |
| --- |
|  |

1. Show the **earliest event time** and **latest event time**, and the critical path activities for **Question 6** in the follow box: (20%)
2. EET and LET: Please place your solution directly on the boxes provided on the bubbles.
3. For Critical Path:

Option 1: You can directly illustrate by highlighting the lines on the diagram below.

Option 2: You can write the critical path sequence as A🡪B🡪C🡪 … for example.

 0

 0

 3

 4

 5

5

 2

 6

 8

8

 9

 9

 10

 10

 3

3

 1

 5

 1

 1

**G(2)**

**A(1)**

**B(1)**

**I(3)**

**K(1)**

**D(2)**

**E(1)**

**H(1)**

**C(2)**

**J(3)**

**L(1)**

**F(2)**

1 -> 2 -> 4 -> 7 -> 8 -> 9 -> 10

− END OF ANSWERS −