

CSCI 3060U Course Project

Winter 2022

Phase #4 - Front End Unit Testing

April 6th, 2022

Analysis and Test Report

Group members:

Ben Tsoumagas 100751395

Thinh Le 100741899

Muaz Rehman 100553376

Cody Malcolm 100753739

Statement Coverage:

Method chosen:

Search.queryMaxPrice(Scanner)

Analysis of code according to statement coverage system:

*For simplicity, we will use the line number as the statement number.

Statement	maxPriceInput input	Test	maxPriceInput
76	0	T1	0
77	0		
79	0		
81	0		
83	0		
84	*	T2	*
87	0		
89	0		
90	100000	T3	100000
91	0		
92	-1	T4	-1
95	0		

Basic block analysis:

```
75 private int queryMaxPrice(Scanner scanner) {
76     LinkedHashMap<String, String> condition = new LinkedHashMap<>();
77     condition.put(IS_NUMBER, "Sorry, that input was not understood. The input should be numeric.");
78
79     String maxPriceInput = Utils.getInput(scanner, condition, prompt: "Please enter the maximum rental price...", allowWildcard: true);
80
81     int maxPrice;
82     // If user inputs wildcard
83     if (maxPriceInput.equals("*")) {
84         2 return 99999;
85     }
86     // Cast input to integer
87     maxPrice = Utils.priceToInt(Float.parseFloat(maxPriceInput));
88     // If max price out of bound, return closest permissible values
89     if (maxPrice > 99999) {
90         3 return 99999;
91     } else if (maxPrice < 0) {
92         4 return 0;
93     }
94
95     return maxPrice;
96 }
```

Unit tests:

Test	maxPriceInput	Expected return value	Actual return value	Test result
T1	0	0	0	Pass
T2	*	99999	99999	Pass
T3	100000	99999	99999	Pass
T4	-1	0	0	Pass

✓ Test Results	26 ms
✓ SearchTest	26 ms
✓ testQueryMaxPrice_T1()	23 ms
✓ testQueryMaxPrice_T2()	1 ms
✓ testQueryMaxPrice_T3()	1 ms
✓ testQueryMaxPrice_T4()	1 ms

Note that no failures were uncovered during the unit testing of this method.

Decision and Loop Coverage:

Method chosen:

Rental.queryUnit(Scanner, ArrayList<Unit>, ArrayList<Unit>, String)

Analysis of code according to decision coverage system:

These tests are specifically designed to work with the following input data similar to that used for the majority of the black box testing (the only change is unit 4 is rented). Specifically, the “units” ArrayList<Unit> is populated with Unit objects with the following data:

Unit	host	city	price	bedrooms	rented	id
1	User001	Toronto	30000	2	false	A0000001
2	User002	Quebec	60050	4	false	A0000002
3	User005	Toronto	20000	1	false	A0000003
4	User005	Calgary	20050	1	true	A0000004
5	User001	Oshawa	45050	3	false	A0000005
6	User001	Tobermory	9999	1	false	A0000006
7	User002	Tofino	25000	2	false	A0000007
8	User005	Toronto	35000	3	false	A0000008

Additionally, these tests assume that in the same user session the user has already listed a new rental. Specifically, the “sessionUnits” ArrayList<Unit> is populated with a Unit object with the following data:

Unit	host	city	price	bedrooms	rented	id
9	User001	Oshawa	15000	1	false	A0000009

Finally, these tests are based on the current user being User001. Specifically, the username is “User001”.

Decision	idToRent	Test	idToRent in order
1 true	A00000009	T1	A00000009, A00000002
1 false	A00000000	T2	A00000000, A00000002
2 true	A00000000		
2 false	A00000001	T3	A00000001, A00000002
3 true	A00000004	T4	A00000004, A00000002
3 false	A00000001		
4 true	A00000001		
4 false	A00000002	T5	A00000002

Decision identification analysis:

```

65 @ private Unit queryUnit(Scanner scanner, ArrayList<Unit> units, ArrayList<Unit> sessionUnits, String username) {
66     Unit unit;
67     String idToRent;
68
69     LinkedHashMap<String, String> condition = new LinkedHashMap<>();
70     condition.put(VALID_ID_FORMAT, "Sorry, that rental does not exist!");
71
72     while(true) {
73         idToRent = Utils.getInput(scanner, condition, prompt: "Please enter the rental ID...");
74         1 if(null != Unit.getUnit(sessionUnits, idToRent)) {
75             // If it is a new listing
76             System.out.println("Sorry, that rental is not available for rental yet!");
77         } else {
78             unit = Unit.getUnit(units, idToRent);
79             2 if (null == unit) {
80                 // Unit isn't found
81                 System.out.println("Sorry, that rental does not exist!");
82             3 } else if (unit.getRented()) {
83                 // Unit is already rented
84                 System.out.println("Sorry, that rental is not available!");
85             4 } else if (unit.getHost().equals(username)) {
86                 // Unit belongs to this user
87                 System.out.println("Sorry, you own this rental!");
88             } else {
89                 return unit;
90             }
91         }
92     }
93 }

```

Analysis of code according to loop coverage system:

*Note that T5 and T2 are redundant with tests identified in the decision coverage analysis

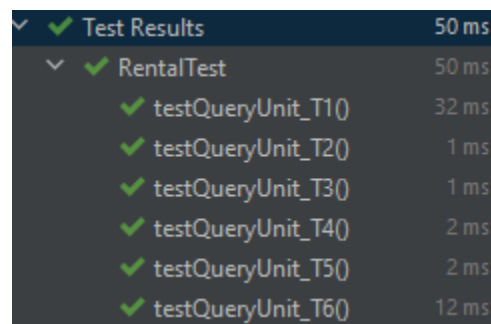
Loop body	idToRent inputs in order	Test
Zero times	n/a	
Once	A0000002	T5
Twice	A0000000, A0000002	T2
Many times	A0000009, A0000000, A0000001, A0000004, A0000002	T6

Loop analysis:

The loop always executes at least one time, so a test for zero times is not possible. The test executes until a valid available unit is selected. If the first unit selected is valid and available, the loop will execute only one time. If the first unit selected is valid but not available, the loop will execute a second time. If multiple selected units are valid but not available, the loop will execute many times.

Unit tests required for decision and loop coverage:

Test	Expected output	Return value	Test result
T1	"Please enter the rental ID..." "Sorry, that rental is not available for rental yet!" "Please enter the rental ID..."	<Unit identified by A0000002>	Pass
T2	"Please enter the rental ID..." "Sorry, that rental does not exist!" "Please enter the rental ID..."	<Unit identified by A0000002>	Pass
T3	"Please enter the rental ID..." "Sorry, you own this rental!" "Please enter the rental ID..."	<Unit identified by A0000002>	Pass
T4	"Please enter the rental ID..." "Sorry, that rental is not available!" "Please enter the rental ID..."	<Unit identified by A0000002>	Pass
T5	"Please enter the rental ID..."	<Unit identified by A0000002>	Pass
T6	"Please enter the rental ID..." "Sorry, that rental is not available for rental yet!" "Please enter the rental ID..." "Sorry, that rental does not exist!" "Please enter the rental ID..." "Sorry, you own this rental!" "Please enter the rental ID..." "Sorry, that rental is not available!" "Please enter the rental ID..."	<Unit identified by A0000002>	Pass



✓	✓ Test Results	50 ms
✓	✓ RentalTest	50 ms
✓	✓ testQueryUnit_T1()	32 ms
✓	✓ testQueryUnit_T2()	1 ms
✓	✓ testQueryUnit_T3()	1 ms
✓	✓ testQueryUnit_T4()	2 ms
✓	✓ testQueryUnit_T5()	2 ms
✓	✓ testQueryUnit_T6()	12 ms

Note that no failures were uncovered during the unit testing of this method.