Hektor Dahlberg 41685

Excel URL: https://docs.google.com/spreadsheets/d/1CIv--WrMmLPPu3RDF1trws-M_xEHDMH4gMuiWvtYYv4/edit?usp=sharing

Task 1:

Test created used assertEquals, did not have much thought behind the test created other to get 100% code coverage. The run also got 100% mutation score. Mutation score and code coverage below.

```
167     public BigDecimal calculateChange(final double price, final String insertedCoins) {
168
169         StringTokenizer st = new StringTokenizer(insertedCoins);
170
171 1       while (st.hasMoreElements()) {
172                 String coin = st.nextToken();
173
174 1               if (coin.equals("TC")) {
175 1                       amountPaid += Coin.TC.value;
176 1               } else if (coin.equals("FC")) {
177 1                               amountPaid += Coin.FC.value;
178 1                       } else if (coin.equals("OE")) {
179 1                               amountPaid += Coin.OE.value;
180 1                       } else if (coin.equals("TE")) {
181 1                               amountPaid += Coin.TE.value;
182                         } else {
183                         System.out.printf("Wrong coin type!" );
184                 }
185         }
186
187 2       return BigDecimal.valueOf(amountPaid - price).setScale(2, RoundingMode.FLOOR);
188     }
```

171	1. negated conditional → KILLED
174	1. negated conditional → KILLED
175	1. Replaced double addition with subtraction → KILLED
176	1. negated conditional → KILLED
177	1. Replaced double addition with subtraction → KILLED
178	1. negated conditional → KILLED
179	1. Replaced double addition with subtraction → KILLED
180	1. negated conditional → KILLED
181	1. Replaced double addition with subtraction → KILLED
187	1. Replaced double subtraction with addition → KILLED 2. replaced return value with null for vending_machine/VendingMachine::calculateChange

Task 2:

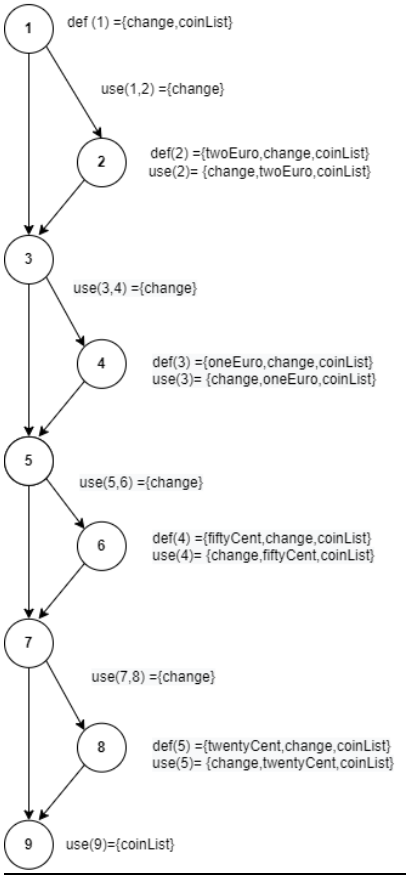
Test created according to the test paths with AssertEquals, no errors where found. Code coverage was 100% and the mutation score 19/25=76%

```
132     public int[] calculateReturningCoins(double change) {
133
134                 int[] coinList = new int[4];
135                 //number of coins corresponding to TE OE FC TC, respectively
136
137 3           if (change / Coin.TE.value >= 1) {
138 1                   int twoEuro = (int) (change / Coin.TE.value);
139 2                   change = change - (twoEuro * Coin.TE.value);
140                     coinList[0] = (int) twoEuro;
141             }
142 3           if (change / Coin.OE.value >= 1) {
143 1                   int oneEuro = (int) (change / Coin.OE.value);
144 2                   change = change - (oneEuro * Coin.OE.value);
145                     coinList[1] = (int) oneEuro;
146             }
147 3           if (change / Coin.FC.value >= 1) {
148 1                       int fiftyCent = (int) (change / Coin.FC.value);
149 2                   change = change - (fiftyCent * Coin.FC.value);
150                     coinList[2] = (int) fiftyCent;
151             }
152 3           if (change / Coin.TC.value >= 1) {
153 1                   int twentyCent = (int) (change / Coin.TC.value);
154 2                   change = change - (twentyCent * Coin.TC.value);
155                     coinList[3] = (int) twentyCent;
156             }
157 1       return coinList;
158     }
```

137	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → SURVIVED 3. negated conditional → KILLED
138	1. Replaced double division with multiplication → KILLED
139	1. Replaced double multiplication with division → KILLED 2. Replaced double subtraction with addition → KILLED
142	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → SURVIVED 3. negated conditional → KILLED
143	1. Replaced double division with multiplication → SURVIVED
144	1. Replaced double multiplication with division → SURVIVED 2. Replaced double subtraction with addition → KILLED
147	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → KILLED 3. negated conditional → KILLED
148	1. Replaced double division with multiplication → KILLED
149	1. Replaced double multiplication with division → KILLED 2. Replaced double subtraction with addition → KILLED
152	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → KILLED 3. negated conditional → KILLED
153	1. Replaced double division with multiplication → KILLED
154	1. Replaced double multiplication with division → SURVIVED 2. Replaced double subtraction with addition → SURVIVED
157	1. replaced return value with null for vending_machine/VendingMachine::c

Hektor Dahlberg 41685

TASK 2	Code coverage=100%		Mutation Coverage= 19/25=76%				
	Test path in graph	Input	Expected Output	NC	EC	EPC	PC
T1	[1,2,3,4,5,6,7,8,9]	3.7	1,1,1,1	9/9 = 100 %	7/12	7/16	4/8
T2	[1,3,4,5,7,9]	1.0	0,1,0,0	6/9	5/12, tot = 11/12	4/12 tot=9/16	4/8 tot=6/8
T3	[1,3,5,7,8,9]	0,2	0,0,0,1	6/9	5/12 tot=12/12	4/12 tot=11/16	4/8 tot=8/8
T4	[1,2,3,5,7,9]	2.0	1,0,0,0	6/9	5/12 tot=12/12	4/12 tot=15/16	4/8 tot=8/8
T5	[1,2,3,5,6,7,9]	2.5	1,0,1,0	7/9	6/12 tot=12/12	5/12 tot=16/16	4/8 tot=8/8



List all predicates and their reachability conditions

Predicate	Line	Predicate	Condition
1	137	change / Coin.TE.value >= 1	TRUE
2	142	change / Coin.OE.value >= 1	TRUE
3	147	change / Coin.FC.value >= 1	TRUE
4	152	change / Coin.TC.value >= 1	TRUE

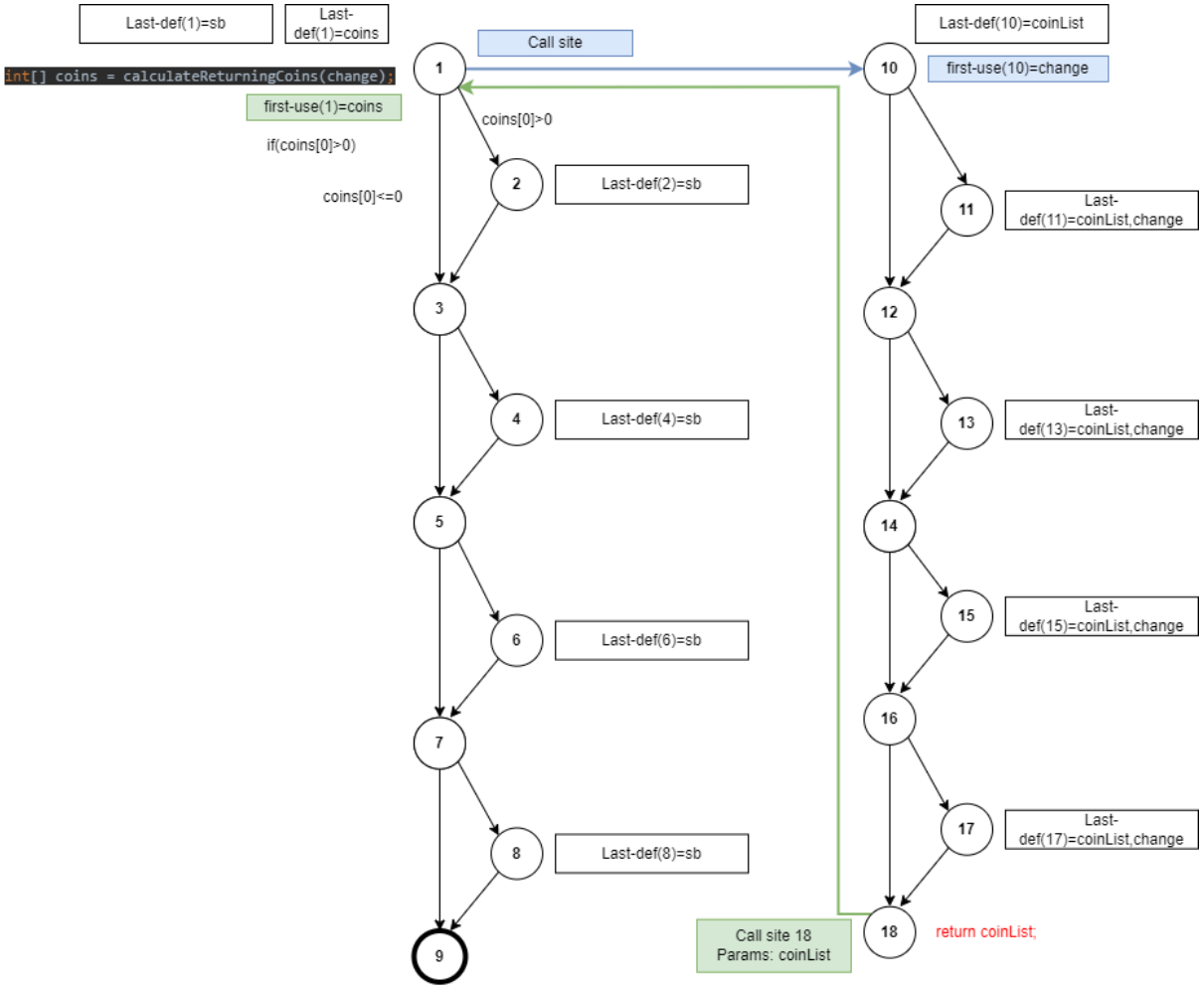
Select the last predicate in the method and list test requirements for predicate coverage.

a= change, b=coin.TC.value, c=1	a/b>=c	TRUE
a= change, b=coin.TC.value, c=1	a/b<=c	FALSE

Hektor Dahlberg 41685

Task 3:

The tests were designed according to the paths and tested that the string returned was the as the expected output. No errors where found. Mutation score was 79% and code coverage 100%.



```
106     public String displayReturningCoins(double change) {
107         // displays the change
108 
109         int[] coins = calculateReturningCoins(change);
110 
111         StringBuilder sb = new StringBuilder();
112 
113 2       if (coins[0] > 0)
114                 sb.append("\t" + coins[0] + " x 2 Euro"); sb.append(nlc);
115 2       if (coins[1] > 0)
116                 sb.append("\t" + coins[1] + " x 1 Euro"); sb.append(nlc);
117 2       if (coins[2] > 0)
118                 sb.append("\t" + coins[2] + " x 50 Cent"); sb.append(nlc);
119 2       if (coins[3] > 0)
120                 sb.append("\t" + coins[3] + " x 20 Cent"); sb.append(nlc);
121 
122 1       return sb.toString();
123 
124     }

132     public int[] calculateReturningCoins(double change) {
133 
134             int[] coinList = new int[4];
135             //number of coins corresponding to TE OE FC TC, respectively
136 
137 3           if (change / Coin.TE.value >= 1) {
138 1                   int twoEuro = (int) (change / Coin.TE.value);
139 2                   change = change - (twoEuro * Coin.TE.value);
140                     coinList[0] = (int) twoEuro;
141             }
142 3           if (change / Coin.OE.value >= 1) {
143 1                   int oneEuro = (int) (change / Coin.OE.value);
144 2                   change = change - (oneEuro * Coin.OE.value);
145                     coinList[1] = (int) oneEuro;
146             }
147 3           if (change / Coin.FC.value >= 1) {
148 1                       int fiftyCent = (int) (change / Coin.FC.value);
149 2                   change = change - (fiftyCent * Coin.FC.value);
150                     coinList[2] = (int) fiftyCent;
151             }
152 3           if (change / Coin.TC.value >= 1) {
153 1                   int twentyCent = (int) (change / Coin.TC.value);
154 2                   change = change - (twentyCent * Coin.TC.value);
155                     coinList[3] = (int) twentyCent;
156             }
157 1       return coinList;
158     }
```

113	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
115	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
117	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
119	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
122	1. replaced return value with "" for vending_machine/VendingMachine::displayReturningCoins → KILLED
137	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → SURVIVED 3. negated conditional → KILLED
138	1. Replaced double division with multiplication → KILLED
139	1. Replaced double multiplication with division → KILLED 2. Replaced double subtraction with addition → KILLED
142	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → SURVIVED 3. negated conditional → KILLED
143	1. Replaced double division with multiplication → SURVIVED
144	1. Replaced double multiplication with division → SURVIVED 2. Replaced double subtraction with addition → KILLED
147	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → KILLED 3. negated conditional → KILLED
148	1. Replaced double division with multiplication → KILLED
149	1. Replaced double multiplication with division → SURVIVED 2. Replaced double subtraction with addition → KILLED
152	1. changed conditional boundary → KILLED 2. Replaced double division with multiplication → KILLED 3. negated conditional → KILLED
153	1. Replaced double division with multiplication → KILLED
154	1. Replaced double multiplication with division → SURVIVED 2. Replaced double subtraction with addition → SURVIVED
157	1. replaced return value with null for vending_machine/VendingMachine::calculateReturningCoins → KILLED

TASK 3	Code coverage=100%	Mutation Coverage= 27/34=79%	

	Test path in graph	Input	Expected Output	All-Coupling-Defs	All-Coupling-Use
T1	[1,10]	0.0	""	1->10 [1,10]	1->10 [1,10]
T2	[11,12,14,16,18,1]	2.0	"1 x 2 Euro"	-11->1 [11,12,14,16,18,1]	-11->1 [11,12,14,16,18,1]
T3	[13,14,16,18,1]	1.0	"1 x 1 Euro"	-13->1 [13,14,16,18,1]	-13->1 [13,14,16,18,1]
T4	[15,16,18,1]	0.5	"1 x 0.5 Cent"	-15->1 [15,16,18,1]	-15->1 [15,16,18,1]
T5	[17,18,1]	0.2	"1 x 0.2 Cent"	-17->1 [17,18,1]	-17->1 [17,18,1]