

Fundamentals of Software Testing*

(A.Y. 2022/2023) – Duration: 1h30m
January 15th, 2025

Exercise 1.

Consider the following program written to establish the rate to apply to the owner of a bank account¹.

```
1 package BankAccount;
2
3 public class AccountRate {
4
5     public static double establishRate(Account account) {
6         double initialRate = 0.02;
7         double finalRate = 0;
8         double rateCoefficient = account.balance * 0.5;
9         if (account.rating == OwnerRating.platinum ||
10             ((account.rating == OwnerRating.platinum) && (rateCoefficient >= 500)) ||
11             rateCoefficient >= 4000)
12             finalRate = initialRate + 0.02;
13         else
14             if (account.rating == OwnerRating.standard && rateCoefficient >= 1000)
15                 finalRate = initialRate + 0.01;
16         return finalRate;
17     }
18 }
```

Apply a test derivation strategy that intends to spot all the possible mistakes (boolean, relational and expressions) related to the conditions of the program. As any good programmer, in case you judge it useful, you can revise the conditional statements to make them simpler and easier to understand. Obviously the behaviour should not change.

16 points

Exercise 2.

Consider the program in the previous exercise

- Compute the condition/decision coverage for the test suite you derived for Exercise 1 (in case you did not derive any test suite yet, manually derive one with at least 5 tests, and then start from such test suite).
- Derive a data-flow graph for the program above. Some suggestions on how to perform the task:
 - Use line numbers to define the blocks in the data flow.
 - consider the variable in the class **Account** as four different variables.
- Provide an assessment for the all-uses coverage criteria

16 points

*For the QAIS module you should solve Exercise 1 in 45 minutes

¹At the end of the document you find all the complementary classes needed to better understand the behaviour of the program

Supporting Java classes

```
1
2 //OwnerRating.java
3 package BankAccount;
4
5 public enum OwnerRating {
6     platinum,
7     premium,
8     standard,
9     risky
10 }
11
12 //Account.java
13 package BankAccount;
14
15 public class Account {
16     //fields made public to reduce code verbosity
17     public double ID;
18     public String accountHolderName;
19     public OwnerRating rating;
20     public double balance;
21
22     public Account(double id,String ahn,OwnerRating or,double balance) {
23         this.ID = id;
24         this.accountHolderName = ahn;
25         this.rating = or;
26         this.balance = balance;
27     }
28 }
```