

# Fundamentals of Software Testing\*

(A.Y. 2022/2023) – Duration: 1h30m  
July 6<sup>th</sup>, 2023

## Exercise 1.

A company selling various type of goods on-line has asked to our company to develop a simple software components that permits to select the packaging to be used to dispatch ordered goods. The component receives the order made by a customer from an external system, and it delivers to the operator the correct package/s to use in order to prepare the shipping. The selection of the package is based on the following specification:

1. In case the order refers to electronic appliances the package should be made using a special box, named TypeS, which has a special structure to reduce the risk of damage of the content;
2. In case the order refers to porcelain, or similar material, the packaging to use is still TypeS but it is recommended to add a particular plastic component inside the carton during the packaging process;
3. In case the order includes clothes the package to use is constituted by a simple plastic bags
4. if the order refers to drinks the packaging is constituted by a box, named TypeW
5. For other goods (referred as “normal goods”) the package to use is the TypeN.
6. It is possible that the order includes multiple products. In such a case the following constraints have to be considered:
  - Drinks can be shipped only together with porcelain (and viceversa) and in this case the package to use is the TypeW where the special plastic has to be added in the slot allocated to the porcelain
  - Clothes, electronic products, and “normal goods” can be shipped together on a TypeS (in case an electronic product is included) of TypeN package, where the clothes are at first stored within a plastic bag
  - It is not possible to include both electronic appliances and porcelain in the same order

You have been appointed as manager of the testing process and you are asked to select a testing strategy to derive the test suite. For the sake of simplicity you should not worry in any way about the dimension of a box or on the possibility that you would need more than one box.

**16 points**

---

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

## Exercise 2.

Consider the following program implementing the specification defined at the previous exercise (for your convenience you find in the next page the supporting classes for the program below, in case you have difficulties to fully understand the code):

```
1 package packaging;
2 import java.util.Vector;
3 public class PackagingManager {
4     static Package packagingProposal(Order order) {
5         Package package2use = new Package();
6         Vector<OrderItem> orderItems = order.getOrderItems();
7         int i = 0;
8         while (i < orderItems.size()) {
9             switch (orderItems.get(i).getType()) {
10                 case Electronic:
11                     package2use.typeS = true;
12                     if (package2use.plasticFiller == true) {
13                         System.out.println("WARNING: Porcelain and Electronic Appliances in the same order");
14                         return null; }
15                     if (package2use.typeN == true)
16                         package2use.typeN = false;
17                     break;
18                 case Porcelain:
19                     if (package2use.typeS == true && package2use.plasticFiller == false) {
20                         System.out.println("WARNING: Porcelain and Electronic Appliances in the same order");
21                         return null; }
22                     package2use.plasticFiller = true;
23                     if (package2use.typeW != true)
24                         package2use.typeS = true;
25                     break;
26                 case Clothes:
27                     package2use.plasticBag = true;
28                     break;
29                 case Drinks:
30                     package2use.typeW = true;
31                     if (package2use.plasticFiller == true)
32                         package2use.typeS = false;
33                     break;
34                 case Other:
35                     if (package2use.typeS == true && package2use.plasticFiller == true)
36                         package2use.typeN = true;
37                     break;
38             }
39             i++;
40         }
41         return package2use;
42     }
43 }
```

- Compute the condition/decision coverage for the test case 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 `Package` as five different variables.
- Provide an assessment for the p-use coverage criteria
- Is there any bug in the code that you were able to spot?

16 points

## Supporting Java classes

```
1 // File: GoodType.java
2 package packaging;
3 enum GoodType {Electronic, Porcelain, Clothes, Drinks, Other}
4
5 // File: OrderItem.java
6 package packaging;
7 public class OrderItem {
8     private GoodType type;
9     public OrderItem(GoodType type) { this.type = type; }
10    public GoodType getType() { return this.type; }
11 }
12
13 // File: Order.java
14 package packaging;
15 import java.util.Vector;
16 public class Order {
17     private Vector<OrderItem> items;
18     public Order(Vector<OrderItem> items) { this.items = items; }
19     public Vector<OrderItem> getOrderItems() { return this.items; }
20 }
21
22 // File: Package.java
23 package packaging;
24 // For the sake of simplicity this class does not use encapsulation
25 public class Package {
26     public Boolean typeS, typeW, typeN, plasticBag, plasticFiller;
27     public Package() {
28         this.typeS = false;
29         this.typeW = false;
30         this.typeN = false;
31         this.plasticBag = false;
32         this.plasticFiller = false;
33     }
34 }
```