CS265 Software Testing
Lab Week 04
Combinational Testing
(2.5% + 1.5% Bonus)

Important note: When building truth tables, we apply a particular notation with respect to the order of true and false cases. For a full set of combinations (i.e. in an initial truth table), false (F) is situated on the left, and true (T) on the right of the table. We organise the creation of truth tables along half-way splits for false-true between each cause (please review lecture slides for examples). This method has been repeatedly demonstrated in class and prevents errors and confusion. Please stick to this notation for ALL exercises in today's lab, in future labs and in the exam.

Exercise 1: Insurance

Consider a function `equipmentInsurance` that is used by an airline to automatically determine the amount of insurance customers must pay for their ticket. The following rules apply:

- Each customer can bring a maximum of two pieces of equipment
- They can bring one piece of sports equipment and one piece of musical equipment
- If they bring both sports and music equipment (one each max), the insurance is €20.
- If they only bring one piece of equipment, the insurance is €10.
- If they bring no equipment, then the insurance free is 5€.

The function has two boolean input parameters (sportsEquipment and musicEquipment) and one output (insurance).

Tasks:

- Using **Combinational Testing**, draw a truth table to find the rules that govern the causes and effects of this scenario. You DO NOT need to do Equivalence Partitioning or Boundary Value Analysis. You may use the template `CS265_Lab_Week04_Template` as provided on Moodle or hand-draw the tables.
- Design test data for the test cases (rules).
- Write suitable tests in TestNG based on your test data. The code for the function of this exercise is provided on Moodle as part of `CS265_Lab_Week04.java`.
- Complete some questions as part of this week's Moodle quiz. **You must complete these questions to receive points for this exercise.**

Exercise 2: Discount Rate

A function discountRate receives orders and calculates a discount rate. If an order is less than 50 units, there is no discount. However, if there is a cash payment and the order is placed in wholesale season, the discount rate is 4%. In case of either a cash payment or of an order being placed in the wholesale season, the discount rate is 2%. If the order quantity is at least 50 units, the discount rate is 4%. However, if there is a cash payment on delivery and/or the order is placed in the wholesale season, there is is an additional 2% discount.

Function signature:

```
double discountRate(int quantity, boolean cashpayment, boolean wholesale)
```

Specification:

Input:
quantity: int
cashPayment: boolean
wholesale: boolean

Output:
discount: double

If quantity < 50, no cashpayment, no wholesale: No discount or discount of 0%.
If quantity < 50, cashpayment, and wholesale: discount 4%
If quantity < 50, with only either cashpayment or wholesale: discount 2%
If quantity >= 50, no cashpayment, no wholesale: discount 4%
If quantity >= 50, cashpayment and/or wholesale: discount 6%

Tasks:

- Using **Combinational Testing**, draw a truth table to find the rules that govern the causes and effects of this scenario. You DO NOT need to do Equivalence Partitioning or Boundary Value Analysis. You may use the template `CS265_Lab_Week04_Template` or hand-draw the tables.
- Design test data for the test cases (rules).
- Write suitable tests in TestNG based on your test data. The code for the function of this exercise is provided on Moodle as part of `CS265_Lab_Week04.java`.
- Complete some questions as part of this week's Moodle quiz. **You must complete these questions to receive points for this exercise.**

Exercise 3: Moodle login

Consider the method `doMoodleLogon` which is provided to you via `CS265_Lab_Week04.java`. **You are required to develop a Truth Table or Decision Table AND corresponding test cases for this method based on the specification below.**

You DO NOT need to do Equivalence Partitioning or Boundary Value Analysis. This method is a very simplified version of a user logon to the Moodle system in the University.

The method takes three parameters:

1. **`String username`:** This is the email address of the user. A valid email has at least 4 characters (upper and lower case alphabetic characters, numbers 0 – 9 and the period character) then followed by @nuim.ie. No other string formation is valid.
2. **`String password`:** We shall pretend that this is the password corresponding to the username provided (this isn't how we do this in reality). A valid password is at least 5 characters long (upper and lower case alphabetic characters, numbers 0-9). Spaces are not allowed. No other type of password string is valid.
3. **`Boolean isStudent`:** If this is set to true, the user trying to logon is a student. If it is a member of staff trying to logon then this is false.

- If username and/or password are NULL or of zero length then the method returns "UNKNOWN"
- If username and password are VALID and isStudent is TRUE then the method returns "STUDENT PAGE"
- If username and password are VALID and isStudent is FALSE then the method returns "STAFF PAGE"
- If username and/or password are INVALID (due to incorrect formatting) then the method returns "UNKNOWN"

Tasks:

- Using **Combinational Testing**, draw a truth table to find the rules that govern the causes and effects of this scenario. You DO NOT need to do Equivalence Partitioning or Boundary Value Analysis. You may use the template `CS265_Lab_Week04_Template` or hand-draw the tables.
- Design test data for the test cases (rules).
- Write suitable tests in TestNG based on your test data. The code for the function of this exercise is provided on Moodle as part of `CS265_Lab_Week04.java`.
- Complete some questions as part of this week's Moodle quiz. **You must complete these questions to receive points for this exercise.**