**BCPR301 – Advanced Programming**

**Assessment 4 Marking Sheet for Tester/ Evaluator**

Student Name

# The compulsory (i.e., ZERO mark if not being provided):

1. You MUST supply a filled self-marking sheet to indicate how many marks you think you can get for each part based on the marking guide provided below.
2. A feature list and an interface diagram of the target source code

# Your repository link:

# Marking guide (max 9 \* N marks in total where N = 3):

1. Development of tests (3 \* N marks)
   1. Developing a set of testing code for all external behaviours of the target source code and all tests should be able to be run together by running a single .py file (1.5 \* N marks)
   2. PEP8 validation on the developed testing code and Version control via an online repository (0.5 \* N marks)
   3. Using coverage package to generate a HTML report to show the developed testing code has **100%** branch coverage for the target source code (N marks)
2. Using coverage package to generate a HTML report to check whether the same set of developed testing code still has 100% branch coverage for the refactored source code done by each coder in your group; Briefly explain the reasons if the same set of developed testing code cannot reach to 100% branch coverage for both target source code and refactored code and how to handle this problem accordingly. (2 \* N marks)
3. Using the marking sheet for coder to evaluate the Assignment 2 solution of each coder in your group. For each part listed in the marking guide for coder, as a tester, you need to indicate corresponding marks and reasons why such marks are given by you for that part in a concise fashion. Your marks will depend on the correctness of your evaluation. (4 \* N marks)

**BCPR301 – Advanced Programming**

**Assessment 4 Marking Sheet for Coder**

Student Name

# The compulsory (i.e., ZERO mark if not being provided):

1. You MUST supply a filled self-marking sheet to indicate how many marks you think you can get for each part based on the marking guide provided below.
2. A feature list and an interface diagram of the target source code

# Your repository link:

# Marking guide (max 9 \* N marks in total where N = 3):

1. Smell detection (4 \* N marks)
   1. Identification of N+ bad smells in the target source code. For the sake of learning, you need to identify **bad smells from different bad smell categories (excluding comments bad smell)** covered in our class (N marks)
   2. The location of each bad smell identified (N marks)
   3. The reasons why you think that the ones you identify are bad smells in a concise fashion (N marks)
   4. Brief description about the refactoring strategies/ approaches you are going to use to remove each bad smell (N marks)
2. Refactoring (5 \* N marks)

In order to remove the bad smells that you previously identified, you need to follow the refactoring process that we discussed in class sessions.

1. Identifying the worst smell and the reasons why it is the worst one (N marks)
2. Version control via an online repository (N marks)
3. Modification to remove the worst smell and PEP8 validation (2 \* N marks)
4. Testing and effectively evaluations on your refactored code in a concise fashion (N marks)