**BCPR301 – Advanced Programming**

**Assessment 4 Marking Sheet for Coder**

Student Name Sini Gao

# 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:

https://github.com/echo127/TIGR-Assignment-2.git

# 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)

**12 marks**

Bad Smell:

* Long Method- Category: Bloater
* Feature Envy- Category: Couplers
* Duplicate Code- Category: Dispensables
  1. The location of each bad smell identified (3 marks)
* Bad smells: Long Method

TigrParser.py line 22-71

* Bad smells: Feature Envy

TigrParser.py line 62 self.drawer.\_\_getattribute\_\_

* Bad smells: Duplicate Code

all the error handling in TigrParser.py and TigrReader.py

* 1. The reasons why you think that the ones you identify are bad smells in a concise fashion (3 marks)
* Bad smells: Long Method

1. the method parse has 51 lines, which cannot fit in one screen.

2. the method code will be hard to understand and maintained by other developers.

3. against the single responsibility principle.

* Bad smells: Feature Envy

break encapsulation of drawer

* Bad smells: Duplicate Code

1. The error handling is all over the place

2. It's difficult to read and maintain the code

* 1. Brief description of the refactoring strategies/ approaches you are going to use to remove each bad smell (3 marks)

• **Bad smells: Long Method**

Strategy: break the long method into small methods, to make the long method short

• **Bad smells: Feature Envy**

Strategy: make a method in class TigrParser to use instead of using the using Drawers

private method

• **Bad smells: Duplicate Code**

Strategy: Extract all the Error handling into a class ErrorHandle

1. 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)

* **Worst Smell Long Method**

because it’s a really long method and has some other bad smell codes inside as well in it, hard to understand and maintain

* **Worst Smell Feature Envy**

break encapsulation of drawer

* **Worst Smell Duplicate Code**

1. Duplication makes it hard to change the code
2. Hurts the readability
3. Version control via an online repository (N marks)

3 marks

Each bad smell codes followed more than 3 commits to added on GitHub

1. Modification to remove the worst smell and PEP8 validation (2 \* N marks)

6 marks

1. Testing and effective evaluations on your refactored code in a concise fashion (N marks)

Each modification was tested and then validated by PyCharm then commit to GitHub