FIT2099 Assignment 3

Recommendation for engine code

By: Chong Chiu Gin (28842022) & Chin Wen Yuan (29975239)

Good Practice of engine code

1) Reduce dependencies as much as possible

    - all the engine code provided by the teaching team are packed in a package, it provides an encapsulation boundary and make it easier to access and use the classes that are provided.
    - it can also reduce excess import statement as we only need to import the whole package to use and implement the classes within the package.

2) Don't repeat yourselves

    - each piece of engine code gives different usage and there are no unnecessary duplicate codes in the given engine code.
    - inheritance is used and it greatly reduce the chances of getting duplicate codes as new classes that has similar functionality can just use those from the parent class and there is no need to have excessive imports from various class.

3) Classes should be responsible of their own properties

    - we can see that from the given engine code, that all of the attributes in the classes are either declared as protected or private. This can prevent other classes from accessing the attributes directly and it minimize dependencies. Attributes then can be kept in a tightest scope possible so the chances of error across classes can be reduced.
    - classes and their functions within have meaningful names which highly increased the readability of the codes.
    - all the attribute of classes has meaningful variable names and it greatly contribute to the readability of the codes and the uses of that attribute.

Bad Practice of engine code

- Does not follow fail fast design
  The engines codes given does not give clear error message when error is faced. The engine codes return null when the functions failed it is not clear that where does the error come from and where should i focus on mitigating the error.

- Recommendation:
  use assertion to catch errors, so that the boundaries of the function are clearer. Other than that, make the debugging process so much easier when it passed down to another person or even when you are debugging your own codes.

- Advantage:

  this fix is good because if the code is passed down to others, they will be able understand what the code is designed to do and the limitation of that function if they are able to build something better base of the given code. This could also lessen the amount of comments and documentations needed in the code as the person can simply know the boundaries of that function based on the assertion condition or the exception message in try and except clause. If are able to go to a further extend, it will even better to make customize exception for common error across the classes such as UnableToMoveException() to indicate that it's an invalid move other than return false or null.