Vietnam National University, Ho Chi Minh City
Ho Chi Minh City University of Technology
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Viet Nam

Subject: Advanced Programming

Assignment 1
Let the Battle Commence

Requirement

You are given a class diagram as presented in the Appendix. There are three parts.

Part A consists of initial classes, most of which are abstract classes or interfaces. Initial classes will be given to you as well as bytecode form (*class* files).

Part B consists fixed classes that you need to implement, which include the following classes.

- `Knight` and `Warrior`: Normally, when battle's `ground` is regular numbber, the `CombatScore` of a `Knight` or `Warrior` is based on his `WP` and `baseHP`. If `WP` is `1`, the `CombatScore` is the `baseHP`. Otherwise, the `CombatScore` is `baseHP / 10`.
    - There are 2 special cases of `ground` number. If `ground` is a *prime* number, the `CombatScore` of `Warriors` will be double of their `baseHP`. If `ground` is a *square* number, the `CombatScore` of `Knights` will be double of their `baseHP`.
- `Paladin`: When a `Paladin` does battle, his `CombatScore` would be *triple* of his `baseHP` on any `ground`. Since `Paladin` is from Heaven, his `CombatScore` can exceed 999. Especially, if `baseHP` of a `Paladin` is a Fibonacci number F_n with `n > 2`, his `CombatScore` will be `1000 + n`.
    - For example, if a `Paladin` has `baseHP` is 34 (=F_9), his `CombatScore` will be 1009.
- `DeathEater`: A `Death Eater` is a monster, who has no `HP` but only `MP` being a complex number C = (*real*, *imaginary*). When combatting, `CombatScore` of a Death Eater would be $\sqrt{real^2 + imaginary^2}$

Part C consists classes which you are free to change.

<u>Note: The game setting (i.e the `TeamMaker` class) will be changed based on test-cases. You do not need to implement this class.</u>

Vietnam National University, Ho Chi Minh City
Ho Chi Minh City University of Technology
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Viet Nam

Appendix

Initial Class Diagram

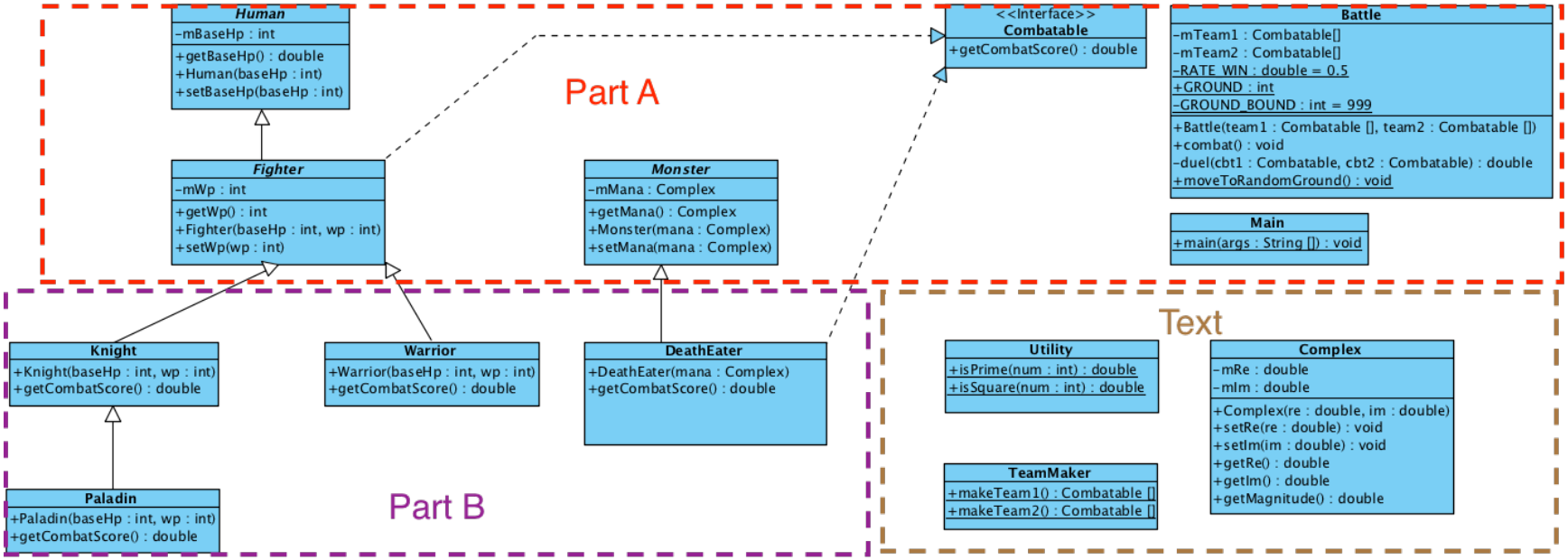

Figure 1. Initial class diagram

Part A includes following classes in bytecode form.

- The interface `Combatable.class`
- 3 abstract classes, which are `Human.class`, `Fighter.class`, `Monster.class`
- `Main.class` contains the `main` method.
- `Battle.class` contains the `Battle` class which simulates a battle, calculates the fighting results.

Part B includes classes that you must create by yourself. These classes are fixed as the diagram in Figure 1.

Part C is the "free-to-change" part as long as there is a `Complex` class.
Note that both `TeamMaker.java` and `Complex.java` are given with dummy code. You can change the source to adapt the requirement. The `TeamMaker.java` will also be replaced when we are grading your work.

Submission

Your initial code is provided as below structure.

```
/
└──── Initial Code
    ├──── class
    │    ├──── Battle.class
    │    ├──── Combatable.class
    │    ├──── Fighter.class
    │    ├──── Human.class
    │    ├──── Main.class
    │    └──── Monster.class
    ├──── source
    │    ├──── DeathEater.java
    │    ├──── Knight.java
    │    ├──── Paladin.java
    │    └──── Warrior.java
    └──── util
         ├──── Complex.java
         └──── TeamMaker.java
         └──── Utility.java
```

Vietnam National University, Ho Chi Minh City
Ho Chi Minh City University of Technology
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Viet Nam

You will complete the assignment individually and submit your code as a ZIP file via LMS. When you submit, rename the folder `Initial Code` to the following format `YourName_StudentID` (e.g., `NguyenVanA_1952968`)

Note: All classes must be placed in *default (root)* package.