

## プログラミングII

### 課題レポート3: リファクタリングを通してユニットテストとバージョン管理に慣れよう

編集日:2022-XX-XX

報告者:e225717 高嶺拓矢

協力者:なし

#### ステップ1: コードの準備

リポジトリは以下の場所です [https://github.com/medicine-t/prog2\\_rep3](https://github.com/medicine-t/prog2_rep3)

#### ステップ2: 死亡した後で攻撃できてしまう件をどうにかしたい

enemyが死亡しているのに攻撃できてしまうことについては、Enemyの持つdead:booleanのフラグを確認すれば良い。そのため、if文を通して生死判定を行うようにした。

```
$ git log -p -1
commit 1e5b2fd153897bba98810d726ded7ef5a4b07503
Author: Takuya Takamine <e225717@ie.u-ryukyu.ac.jp>
Date:   Tue Nov 1 09:26:45 2022 +0900

    report3 step2

diff --git
a/Task/report3/app/src/main/java/jp/ac/uryukyu/ie/e225717/Enemy.java
b/Task/report3/app/src/main/java/jp/ac/uryukyu/ie/e225717/Enemy.java
index ee86a90..b65660c 100644
--- a/Task/report3/app/src/main/java/jp/ac/uryukyu/ie/e225717/Enemy.java
+++ b/Task/report3/app/src/main/java/jp/ac/uryukyu/ie/e225717/Enemy.java
@@ -36,6 +36,9 @@ public class Enemy {
     * @param hero 攻撃対象
     */
    public void attack(Hero hero) {
+        if (dead) {
+            return;
+        }
        int damage = (int) (Math.random() * attack);
        System.out.printf("%sの攻撃! %sに%dのダメージを与えた!!\n", name,
            hero.name, damage);
        hero.wounded(damage);
```

#### ステップ3: カプセル化しよう。

##### 気づいた点

HeroとEnemyで、フィールド変数/メソッドがほとんど共通なため、継承などを通してまとめてかければ楽だろうと感じた。

(おまけ)ステップ4: アクセサのJavaDocを書こう。

下記画像はEnemyClassのJavadoc、特にメゾット部分のスクリーンショット。

メソッドの概要		
すべてのメソッド	インスタンス・メソッド	concreteメソッド
修飾子とタイプ	メソッド	説明
void	attack(Hero hero)	Heroへ攻撃するメソッド。
int	getAttack()	インスタンスの持つattackをintとして返す
int	getHitPoint()	インスタンスの持つHitPointをintとして返す
String <sup>🔗</sup>	getName()	インスタンスの持つname変数をStringとして返す。
boolean	isDead()	インスタンスの持つdeadフラグの状態をbooleanで返す
void	setAttack(int attack)	インスタンスのattackの値を設定する
void	setDead(boolean dead)	インスタンスのdeadフラグの状態を設定する
void	setHitPoint(int hitPoint)	インスタンスのhitPoint変数の値を設定する
void	setName(String <sup>🔗</sup> name)	インスタンスのname変数の値を設定する。
void	wounded(int damage)	自身へ攻撃されたときのダメージ処理をするメソッド。
クラスから継承されたメソッド java.lang.Object <sup>🔗</sup>		
clone <sup>🔗</sup> , equals <sup>🔗</sup> , finalize <sup>🔗</sup> , getClass <sup>🔗</sup> , hashCode <sup>🔗</sup> , notify <sup>🔗</sup> , notifyAll <sup>🔗</sup> , toString <sup>🔗</sup> , wait <sup>🔗</sup> , wait <sup>🔗</sup> , wait <sup>🔗</sup>		