HW Set #4: Cairn RPG

Objective: Work with a partner to use version control properly while implementing the classes that will be needed in a new role-playing game.

1. Get a partner!

2. Set up your environment:

    a. Create a new git repository.

    b. Add the group members, and your professor (Username: bpetcaughCU), as collaborators

    c. Create an empty Intellij Project in the repo, and title it 'CairnRPG_initials_initials'

        i. Example: 'CairnRPG_BP_JS'

3. Complete the following tasks (group members should decide early on which tasks will be completed by each member):

    a. Team member #1:

        i. Implement the World, Character, and Item classes

        ii. Perform Unit testing on all methods for Hero, Enemy, and Boss

    b. Team member #2:

        i. Implement the Hero, Enemy, and Boss classes

        ii. Perform Unit testing on all methods for World, Character, and Item

4. Grades will be assigned based on how accurate the methods match the given UML and descriptions. A grade will also be based on the team's proper usage of GIT and GITHUB. There should be several commits over the course of a week, from different days.

World

Constructor: This should not be passed any parameters. Set all attributes to whatever default values you would like.

EXAMPLE: To create a new world, I should be able to write → new World()

Item

Constructor: REQUIRE parameters for all attributes in the class.

EXAMPLE: To create a new item, I should be able to write → new Item("Potion", 20)

Info(): Provide all information about the class in a SINGLE string *

Character

Constructor: This should be passed all 4 attributes to be set with the given parameters.

EXAMPLE: To create a new character, I should be able to write →

new Character("Maximus", 100.0, 10.0, true)

runAway(): Characters should be given a random probability to successfully run away from the encounter. Return true/false based on whether or not it was successful.

Info(): *

Hero

Constructor: This should be passed the 3 parameters needed to call super() to set the Character attributes (minus the 'name' – you can manually store in the name of the Hero you would like to use for your game). Nothing else should be passed to the constructor. Set the following attributes as default values:

Level: 1

Experience: 0

Money: 0.0

Then add one Potion to the inventory. (i.e . new Item('Potion', 20))

EXAMPLE: To create a new hero, I should be able to write →

new Hero(100.0, 10.0, true)

Fight(Enemy): Randomly decide between using a basic attack that utilizes the attackPower attribute, OR a Special Attack that does double/triple the damage from the basic attack. Print to the console how much damage was done. Make sure in your print statement to use the enemy name, and to have a different message whether or not the Special Attack was triggered. There should also be a small chance to MISS your target. Return true/false based on whether the attack landed.

useItem(int): Look through the inventory for the first item that can be found. Use that item by adding to your health the amount of healingPower it has. Print to the console how much you were healed, the character name, and the name of the item. (i.e A 'Potion' healed Johnny by 15 HP.)

levelUp(): Check how much experience the Hero has, if it is over 100, raise the level by 1, and reset the experience to 0. Increase both the health and attackPower by 10%. This should include a console message about what just happened.

addToInventory(Item): Add the item to the first empty spot in the inventory, return False if the inventory is full.

showInventory(): Return a string displaying all items in the inventory.

Info(): *

Enemy

Constructor #1: This should be passed the 3 parameters needed to call super() to set the Character attributes (minus the 'name' – you can manually store in the name of the Enemy you would like to use for your game). Nothing else should be passed to the constructor. Set the following attributes as default values:

magicPower: 30

EXAMPLE: To create a new enemy, I should be able to write →

new Enemy(100.0, 10.0, true)

Constructor #2: This is required so that the Boss will work later on. Just copy the following:

```
Enemy(String n, int h, int ap, boolean ia) {
    super(n, h, ap, ia);
}
```

fight(): Same as Hero. Feel free to change the randomness or message to customize it for an enemy. Instead of a special attack, they should have a random chance to use a magical attack if they have enough magic points to use it.

Info(): *

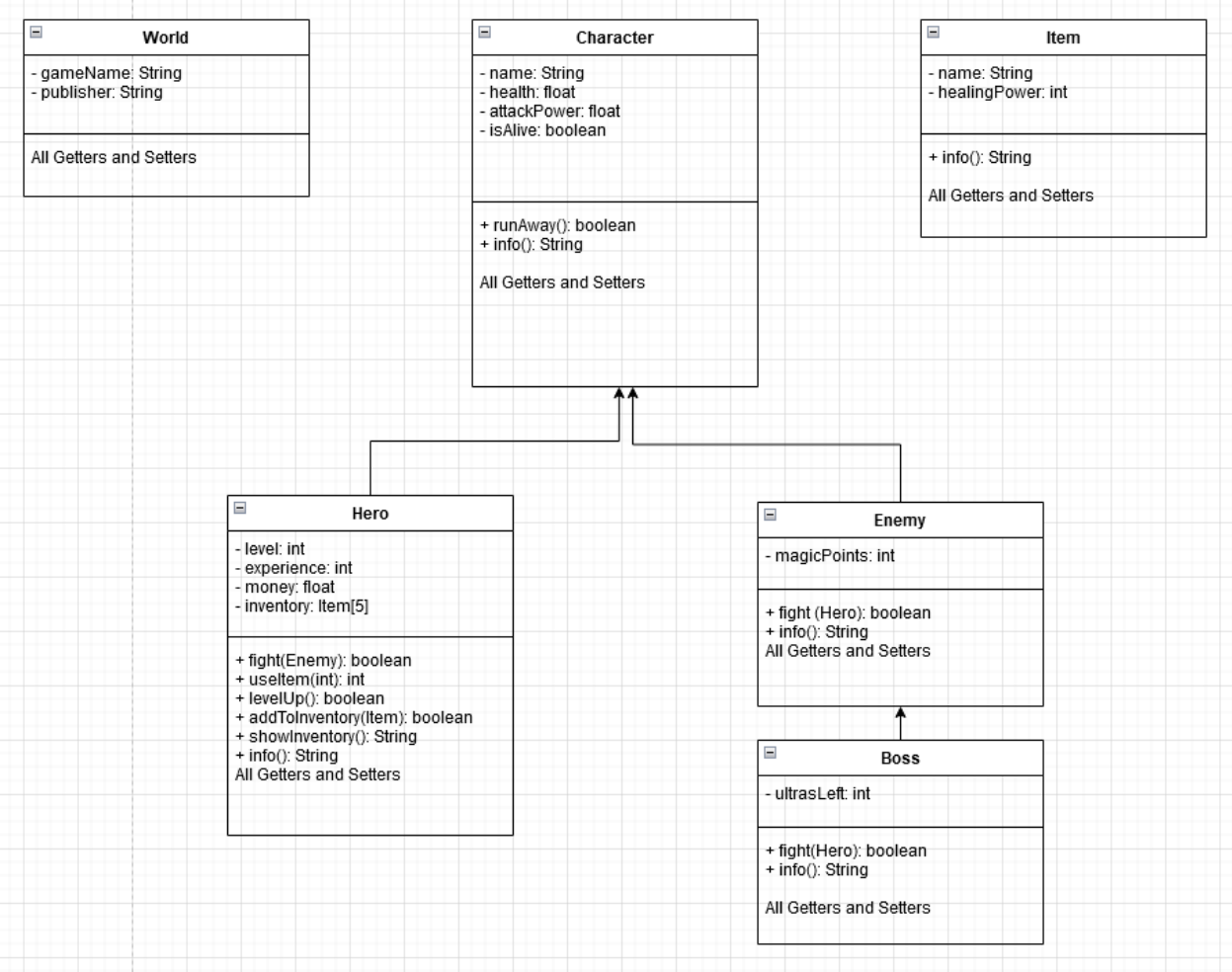
Boss

Constructor: Copy the following (add in your own Boss name to replace mine).

```
Boss(int h, int ap, boolean ia) {
    super( n: "Super Boss Man", h, ap, ia);
    this.ultrasLeft = 3;
}
```

fight(): Same as Hero/Enemy. Feel free to change the randomness or message to customize it for an enemy. Instead of a special attack, they should have a random chance to use an Ultra attack, which uses up one single ultrasLeft. If the boss has 0 of these, they cannot cast that attack.

Info(): *


World
- gameName: String
- publisher: String
All Getters and Setters
Character
- name: String
- health: float
- attackPower: float
- isAlive: boolean
+ runAway(): boolean
+ info(): String
All Getters and Setters
Item
- name: String
- healingPower: int
+ info(): String
All Getters and Setters
Hero
- level: int
- experience: int
- money: float
- inventory: Item[5]
+ fight(Enemy): boolean
+ useItem(int): int
+ levelUp(): boolean
+ addToInventory(Item): boolean
+ showInventory(): String
+ info(): String
All Getters and Setters
Enemy
- magicPoints: int
+ fight (Hero): boolean
+ info(): String
All Getters and Setters
Boss
- ultrasLeft: int
+ fight(Hero): boolean
+ info(): String
All Getters and Setters