DUE DATE: NOVEMBER 18, 2020 AT 11:59PM

Instructions:

- You must complete the "**Blanket Honesty Declaration**" checklist on the course website before you can submit any assignment.
- Only submit the **java files**. Do ***not*** submit any other files, unless otherwise instructed.
- To submit the assignment, upload the specified files to the **Assignment 3** folder on the course website.
- Assignments must follow the **programming standards** document published on UMLearn. You will lose marks for not following these standards.
- After the due date and time, assignments may be submitted but will be subject to a late penalty. Please see the ROASS document published on UMLearn for the course policy for late submissions.
- If you make multiple submissions, only the **most recent version** will be marked.
- These assignments are your chance to learn the material for the exams. Code your assignments independently. We use software to compare all submitted assignments to each other, and **pursue academic dishonesty vigorously**.
- Your Java programs must compile and run upon download, without requiring any modifications.
- Automated tests will be used to grade the output of your assignment. If you do not follow precisely the guidelines detailed below, these automated tests can fail, and you will lose marks.
- **Using both Arrays and/or ArrayLists is allowed in this assignment.**

Assignment overview

You will implement seven classes which form the basis for a simple game simulation. A **World** will contain several **Clans**. A **Clan** will contain a number of **Citizens**. There will be two types of **Citizens**: **Civilians** and **Fighters**. There will be two types of **Fighters**: **Archers** and **Barbarians**. The simulation starts with several clans in the world, each containing random citizens. Repeatedly, one clan will attack another one. The civilians are not affected (unlike in real life), but the fighters damage or kill each other. When a clan loses all its fighters, it is removed from the world. The attacks continue until only one clan is left.

In this assignment you will practice how to implement concepts such as defining hierarchies of classes, abstract classes, and polymorphism. Please keep all of your methods short and simple. Make sure to call methods already created when appropriate, instead of duplicating code for no reason. Unless specified otherwise, all instance variables must be **`private`**, and all methods should be **`public`**. Also, unless specified otherwise, there should be ***no* println** statements in your classes. Objects usually do not print anything; they only return **String** values from methods. Make sure the outputs of the methods, including toString methods, in your classes, are exactly the same as the outputs explained in the instructions.

Instructions

A java file, TestA3.java, will be given to you which contains a main program which will control the overall simulation. You must write all of the following classes for this game.

1: The **`Citizen`** class:

Create an abstract Citizen class. Since it will not be possible to create an object of this class, its methods will be very simple. There are no instance variables. This class exists only to allow polymorphism to occur later.

- Implement a **String toString()** method which returns "**Citizen**".
- Implement **randomCitizen()** method**,** that is a **static** method which will return a random **Citizen**. It should return a Civilian, Archer, or Barbarian, at random. (You will need to implement those classes first.)

2-The **`Civilian`** class:

Create a **Civilian** class as a subclass of the **Citizen** class.

- Implement a **String toString()** method which returns "**Civilian**".
- This class must have no instance variables, and no other methods.

3-The **`Fighter`** class:

Create an **abstract Fighter** class as a subclass of the **Citizen** class.

- Fighters have two private **int** instance variables: the “**currentHealth**”, and the **“currentAttack”**. When fighters attack each other, each one's attack power is used to decrease the other one's health. When a fighter's health reaches 0, it dies.
- Create a custom constructor with two parameters which initializes both the health and the attack power. The first input parameter of your constructor should be currentHealth and the second one should be currentAttack.
- Implement a **String toString()** method which returns **"Fighter"** followed by the values of health and attack power.
  - **`Fighter: health=100 attack=15`**
  - **`Fighter: health=116 attack=10`**
- Provide accessor (get) methods only if needed by other classes.
- Do not provide any mutator (set) methods. The instance variables must be strictly private! These values are changed only by the following methods.
- Write a method **decreaseHp(int)** which will reduce the health of this fighter by the given amount. Health should not go below 0.
- Write a method **boolean isDead()** which will detect when the fighter is dead (health is zero).
- Write a method **void gainPower()** which does nothing. But subclasses of this class will use this method, and so the superclass must also have one to allow polymorphism.
- Write a method **void gainPower(int)** which increases the attack power by the given amount. Survivors of a fight will gain attack power, and this can be useful in the subclasses to override the gainPower() method.
- Note that health only goes down, never up, and attack power only goes up, never down. "What doesn't kill you makes you stronger."

4-The **`Archer`** class:

Create an **Archer** class as a subclass of the **Fighter** class.

- Archers always start with 100 health and 15 attack power. Provide a constructor with no parameters which does this.
- Implement a **String toString()** method which returns the String produced by the **Fighter** superclass, with **"(Archer)"** added to the end.
  - **`Fighter: health=100 attack=15 (Archer)`**
- Write a method **void gainPower()** which adds 10 to the archer's attack power.

5-The **`Barbarian`** class:

Create a **Barbarian** class as a subclass of the **Fighter** class.

- Barbarians always start with 200 health and 20 attack power. Provide a constructor with no parameters which does this.
- Implement a **String toString()** method which returns the String produced by the **Fighter** superclass, with **"(Barbarian)"** added to the end.
  - **`Fighter: health=145 attack=110 (Barbarian)`**
- Write a **method void gainPower()** which adds 15 to the barbarian’s attack power.

6-The **`Clan`** class:

Create a Clan class.

- A clan must have a name, and a list of **Citizens**.

- Provide a constructor with the header **public Clan(String clanName, int clanSize**) which will create a clan with the given name, and the given number of citizens. The number of citizens will never change later, although some of them (the fighters) may become dead. The constructor should create and store the correct number of random citizens, using the **randomCitizen** method of the Citizen class.
- Since a clan will be deleted from the game when it has no fighters left, it is important that the clan has at least one to start with. Make sure the first citizen is an Archer or Barbarian.
- Provide a **getName()** accessor method.
- Write a private **int totalAttackPower()** method which will find the total attack power of all Fighters in this clan who are not dead (sum of all attack power of all Fighters).
- Write a private **int numFightersAlive()** method which will count the number of Fighters in this clan who are not dead.
- Write a method with the header **public void attack(Clan others)** which will cause two clans to fight each other (this clan, and the others). When clan A attacks clan B, the maximum possible damage to members of clan B is the total attack power of clan A divided by the number of living fighters in clan B. The health of each living fighter in clan B should be reduced by a random value between 0 and this maximum value. This should be done in **both** directions as if both clans are attacking each other **at the same time** (this clan attacks the others, but the others attack this clan, too.)
- Write **a boolean isDead()** method which will return true if there are no living fighters in this clan.
- Write a **void gainPower()** method which will increase the attack power of every living fighter in this clan. This will be called (in the World class, below) to increase the power of the survivors of a fight. Don't increase the attack power of dead fighters!
- Provide the usual **String toString()** method, which will return a multi-line String showing the information on the whole clan:
  - The first line should give the clan's name.
  - The second line should give the clan's total attack power.
  - The remaining lines should list the citizens, one per line, using the Strings provided by their toString methods. Do not list dead fighters, only living ones.
  - Follow the example here:

```
Clan Raiders:
Total attack power: 440
Fighter: health=87 attack=110 (Barbarian)
Fighter: health=145 attack=110 (Barbarian)
Civilian
Fighter: health=116 attack=110 (Barbarian)
Fighter: health=134 attack=110 (Barbarian)
```

7-The `World` class:

Create a **World** class. The world will contain a list of all of the clans, and provide a few methods that will complete the functionality of the game.

- A **World** must have instance variable(s) that store a list of **up to 10 Clans**. The number of clans will change during the game.
- Provide a constructor (no parameters) that creates a world containing no clans.
- Provide a **void addClan(Clan c)** method to add a new clan to the world. You do not need to check to see if the maximum number of clans has been exceeded.
- Provide an **int getNumClans()** method which will return the current number of clans in the world.
- The most important method is **void attack(int clan1, int clan2)** which will cause the two clans to attack each other. (The clans are identified by index number, not by name.) It should print a message indicating that this is happening, including the names of the two clans. Surviving fighters in each clan should gain power. If either (or both) clans no longer have any living fighters, they should be deleted from the world. A message should also be printed when a clan is deleted. You may want to write additional private methods to support this method.

- Provide a **String toString()** method that will return a multi-line String listing the entire world, and all the clans in it. It should give the number of clans in the world, and then list each clan. Add some sort of dividing line to separate the clans and make them more visible. See the sample output that follows.

```
There are 3 clans in the world.
-----
Clan Settlers:
Total attack power: 65
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=100 attack=15 (Archer)
Civilian
Civilian
Fighter: health=100 attack=15 (Archer)
Fighter: health=100 attack=15 (Archer)
-----
Clan Raiders:
Total attack power: 80
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
Civilian
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
-----
Clan Hordes:
Total attack power: 75
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
Civilian
Civilian
Fighter: health=200 attack=20 (Barbarian)
Civilian
Fighter: health=100 attack=15 (Archer)
```

8-The `TestA3` class:

This class is provided for you to test Assignment 3. It creates a world containing 3 clans, and then causes pairs of these clans to attack each other until only 1 (or perhaps 0) clans remain in the world.
See the Sample output on the last page. Your output should be similar to this, but it does not have to match it exactly. Every time the program runs, the output will be different, because the game is random. Choose a test run that is not too large, but shows that your program works as it should. Put the output from that test run into a text file, and hand it in with your assignment. Name the file YourNameA3output.txt.

Hand in

Submit only the .java files for your seven classes (Citizen, Civilian, Fighter, Archer, Barbarian, Clan and World). ***Do not submit .class or .java~ files!*** You do ***not*** need to submit the **TestA3.java `java`** files that were given to you. If your submitted code fails to compile and run, you will lose ***all*** of the marks for the test runs. The marker will ***not*** try to run anything else, and will ***not*** edit your files in any way. ***Make sure none of your files specify a package at the top, otherwise your code will not compile on the marker's computer!***

```
There are 3 clans in the world.
-----
Clan Settlers:
Total attack power: 65
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=100 attack=15 (Archer)
Civilian
Civilian
Fighter: health=100 attack=15 (Archer)
Fighter: health=100 attack=15 (Archer)
-----
Clan Raiders:
Total attack power: 80
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
Civilian
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
-----
Clan Hordes:
Total attack power: 75
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
Civilian
Civilian
Fighter: health=200 attack=20 (Barbarian)
Civilian
Fighter: health=100 attack=15 (Archer)

Clan Settlers attacks Raiders!

There are 3 clans in the world.
-----
Clan Settlers:
Total attack power: 110
Fighter: health=182 attack=35 (Barbarian)
Fighter: health=88 attack=25 (Archer)
Civilian
Civilian
Fighter: health=82 attack=25 (Archer)
Fighter: health=80 attack=25 (Archer)
-----
Clan Raiders:
Total attack power: 140
Fighter: health=195 attack=35 (Barbarian)
Fighter: health=197 attack=35 (Barbarian)
Civilian
Fighter: health=190 attack=35 (Barbarian)
Fighter: health=193 attack=35 (Barbarian)
-----
Clan Hordes:
Total attack power: 75
Fighter: health=200 attack=20 (Barbarian)
Fighter: health=200 attack=20 (Barbarian)
Civilian
Civilian
Fighter: health=200 attack=20 (Barbarian)
Civilian
Fighter: health=100 attack=15 (Archer)

Clan Hordes attacks Raiders!
```

```
There are 3 clans in the world.
-----
```

…More attacks happen…

```
Clan Raiders attacks Settlers!

Clan Settlers is wiped out!!

There are 2 clans in the world.
-----
Clan Raiders:
Total attack power: 440
Fighter: health=87 attack=110 (Barbarian)
Fighter: health=145 attack=110 (Barbarian)
Civilian
Fighter: health=116 attack=110 (Barbarian)
Fighter: health=134 attack=110 (Barbarian)
-----
Clan Hordes:
Total attack power: 285
Fighter: health=76 attack=95 (Barbarian)
Fighter: health=32 attack=95 (Barbarian)
Civilian
Civilian
Fighter: health=34 attack=95 (Barbarian)
Civilian
```

…More attacks happen…

```
Clan Raiders attacks Hordes!

Clan Hordes is wiped out!!

There are 1 clans in the world.
-----
Clan Raiders:
Total attack power: 465
Fighter: health=41 attack=155 (Barbarian)
Civilian
Fighter: health=31 attack=155 (Barbarian)
Fighter: health=58 attack=155 (Barbarian)

Game Over.
```