CSE 110: Assignment 2

Monsters come and go from the zoo over the years. Monsters each have a name, a number of legs, a number of eyes and they may fly (or not). Monsters also have a favorite food. Their food has a name and a typical serving size measured in kilograms. The zoo's keepers need to keep track of how much each monster has been fed during its lifetime. Sometimes monsters get in fights and they might lose an eye or a limb. This information needs to be maintained in the zoo's database. The zoo-keeper can add monsters by only mentioning their name or with all attributes

Problem

We provide two classes **TestZoo** and **TestMonster**. Now write the source codes of class **Zoo**, class **Monster** and class **Food** so that the given classes work.

Step 1: Implement Food

First we need a class to model foods. Start by creating a class called *Food*. Copy and paste the skeleton below. This class defines methods to get the name of a food, find the serving size of the food. However, the skeleton that we provide is missing the implementations of the methods. Fill in the body of the methods with the appropriate code.

Note:

1. While implementing constructor method, remember each food has a name and a typical serving size measured in kilograms
2. serving size can never be negative

Step 2: Implement Monster

Next we need to build the class that will represent each monster. Create a class called **Monster**. We provide a class **TestMonster** that creates a monster named *cookieMonster*, then performs some operations on the monster. However, all the methods and member variables of class **Monster** are missing. You will need to define and implement the missing methods and member variables of class **Monster**. Read the main method of class **TestMonster** and look at the compile errors to figure out what methods are missing.

Note:

1. While implementing constructor method, remember the zoo-keeper can add monsters by only mentioning their name or with all attributes (with name, number of legs, number of eyes, favourite food, they may fly or not)

2. Number of limbs or eyes can never be negative

3. You will find the *eat()* method to have two versions

Step 3: Implementing Zoo

Create a class called **Zoo**. Copy and paste the skeleton below. This class defines methods to add a single monster or an array of monsters, get the number of monsters to the zoo. However, the skeleton that we provide is missing the implementations of the methods. Fill in the body of the methods with the appropriate code.

Note:

1. number of monsters can not surpass the capacity of zoo

Instructions

1. **DO NOT PANIC**. It may seem a bit lengthy at first. Once you follow the steps sequentially you will find it easier.

2. You should get a small part working at a time. Start with **Step** 1 first. To check whether you are doing it right or wrong take class **TestMonster** and comment out all the code inside the *main* method.

3. Write the constructor for class **Food**. Then uncomment the first line of main method of class **TestMonster**. If you have written your constructor correctly the code should run without any error.

   Next implement method *getName()* and *getServingSize()* and uncomment the 2nd and 3rd line like before to check if they are correct or not.

4. Implement Step 2 and 3 in the same way as stated above.

5. DO NOT CHANGE THE SOURCE CODES **TestMonster.java** and **TestZoo.java**

Deadline

Deadline is set at **8 October, 2019 4:50 pm**.

Food.java

```
public class Food {
    private final String name;
    private double servingSize;

    //creates a new food
    public Food(String name, double servingSize) {
```

```
        //implement this method
    }
    //gets the name of the food
    public String getName() {
        //implement this method
    }
    //gets the serving size of the food
    public double getServingSize() {
        //implement this method
    }
}
```

Monster.java

```
public class Monster {
    //implement this class
}
```

Zoo.java

```
public class Zoo {
    private int capacity;
    private int numberOfMonsters;
    private Monster[] entriedMonsters;

    //creates a new zoo
    public Zoo(int capacity) {
        //implement this method
    }
    //adds an array of monsters to the zoo
    public void addAMonster(Monster[] monsters){
        //implement this method
    }
    //adds a monster to the zoo
    public void addAMonster(Monster monster){
        //implement this method
    }
    //returns the number of monsters in zoo
    public int getNumberOfMonsters() {
       //implement this method
    }
}
```

Output of TestMonster

cookie
3.0
2
1
3
4
5
cookie has 2 limbs and 1 eyes.

Output of TestZoo

5 monsters
Zoo is full
5 monsters