You need to write classes to complete a Java program that simulates the ecosystem of a savannah (of course an oversimplification of such a thing!) The main class, Savane, is provided. The classes are organized according to the diagram that you will find in the Savane.java file.

You must implement an Animal class that implements the Prey/Predator relationship. The methods of this relationship are described in the ProiePredateur interface that is provided. In this work, the preys are antelopes, and their specific behavior will have to be implemented in the Antilope class which extends the Animal class. Predators are lions and their specific behavior will need to be implemented in the Lion class which also extends the Animal class. The population (Population class) of the savannah (Savane class provided) consists of grass (Herbe class provided), a herd of antelope that eats grass and a herd of lions that eat grass. antelopes.

⚠ Do not modify the provided interfaces and classes.

The Savane class has a constructor that receives several arguments to initialize the simulation (see Savane.java). This builder creates the grass and the herds of animals, as well as the population that makes up the savannah. It also has the simule() method which initializes the savannah and executes a loop for the number of years of simulation desired and passed as an argument. This loop calls up methods for aging, hunting, and reproducing savannah, respectively. The class finally contains the stats method which allows to display certain data on the savannah.
The Population class stores grass and animals. All animals (antelopes and lions) must be inserted into a single ArrayList:
```
private ArrayList<Animal> individus = new ArrayList<>();
```
The Population class must implement the methods *vieillir(), chasser() and reproduire().*

vieillir() **//to get old**
This method ages grass (already implemented) and animals. An animal ages by adding 1 year to its age and increasing its mass by multiplying it with its growth factor. If an animal exceeds its maximum age (50 years for a lion and 15 years for an antelope), it dies. An animal is born with age 0 and mass 10. The maturity age of a lion is 5 years while that of an antelope is 2 years. This detail is important for reproduction, since only mature females can procreate.

chasser() **//hunt**
This method causes animals to hunt for food. Lions hunt live antelopes and antelopes eat grass. The first method called in chasser() is melanger() which shuffles the order of animals in the list. In order for everyone's labs to return the same results, you need to implement melanger() like this:

```
public void melanger()      {Collections.shuffle(this.animaux, new Random(4));}
```

Chasser consists of making all the animals of the savannah eat, in the order of the ArrayList.

A lion that hunts must eat antelopes for the equivalent of twice its mass. To do this, it cycles through the antelopes in the population in the order of the ArrayList until it can pick up that mass of food. The hunted antelopes are killed. In each hunting season, only 20% of the antelopes alive at the start of the hunt are “huntable”. Once this number of antelopes have been killed, there are no more antelopes available to the lions. If a lion fails to find its mass of food, because there are no more antelopes to hunt, it dies.
To survive, an antelope must eat twice its mass of grass each year. This mass is taken from the available mass of grass, which is reduced accordingly. If there is not enough grass to feed an antelope, it dies.

reproduire(). //reproduce
All pairs of mature animals have one baby per year. It is assumed that there are as many males as females and that all animals mate faithfully! We are therefore witnessing the birth of a newborn per female. The number of females corresponds to half of the mature population.

Functioning
You don't have a main method to write or I/O to manage. The classes will be used by test programs, some of which for testing your classes will be provided. For the correction, your classes will also be tested with test programs that will remain unknown to you.

java code
Your Java code (your classes) must respect the best practices mentioned in class: camelCase for your identifiers, encapsulation, information hiding, cohesion, polymorphism, etc. All files are provided with the lab distribution. Some classes need to be completed: Animal.java, Antilope.java, Lion.java and Population.java.
⚠ Avoid accented characters in your codes. ⚠ Do not use packages.