
# Lab 8: OOP

## SUNY Korea - Francois Rameau

Spring 2023

# GitHub Classroom

## Lab 8 - OOP

# A little story

## In this lab we will code the story of the hare and the tortoise using OOP

### The Hare & the Tortoise

A Hare was making fun of the Tortoise one day for being so slow.

"Do you ever get anywhere?" he asked with a mocking laugh.

"Yes," replied the Tortoise, "and I get there sooner than you think. I'll run you a race and prove it."

The Hare was much amused at the idea of running a race with the Tortoise, but for the fun of the thing he agreed. So the Fox, who had consented to act as judge, marked the distance and started the runners off.

The Hare was soon far out of sight, and to make the Tortoise feel very deeply how ridiculous it was for him to try a race with a Hare, he lay down beside the course to take a nap until the Tortoise should catch up.

The Tortoise meanwhile kept going slowly but steadily, and, after a time, passed the place where the Hare was sleeping. But the Hare slept on very peacefully; and when at last he did wake up, the Tortoise was near the goal. The Hare now ran his swiftest, but he could not overtake the Tortoise in time.

*The race is not always to the swift.*

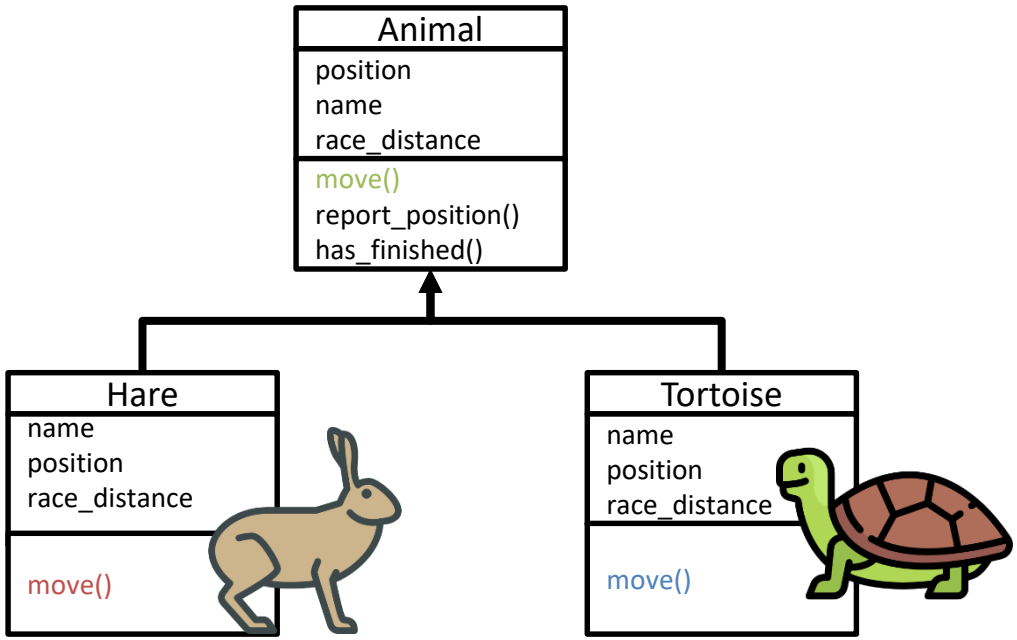
# Rules of the game

- **Today's task:**
  - Model a classic race between a hare and a tortoise using Python and object-oriented programming.
- **The race:**
  - The distance is set in kilometers.
  - Each day, the animals move a certain distance.
- **The competitors:**
  - The hare: Fast (5 to 10 km/day), but might not move at all (50% chance).
  - The tortoise: Slow (1 to 5 km/day), but always moves.
- **Our goals:**
  - Create classes for the animals, allowing them to move and report their positions.
  - Run the race, observing positions daily, until the finish line is reached.
  - Apply and understand Python classes, objects, and inheritance.

**Why no unit tests?**

This lab focuses on exploring and understanding object-oriented programming principles. Due to the randomness involved in the animal movements, traditional unit tests are not provided.

# The participants




We will create these 3 classes with inheritance

# Task 1: Create an Animal class

The Animal class is the parent class for our **Hare** and **Tortoise** classes. It provides the shared attributes and methods that all animals in this scenario have.

## Attributes

- **name**: The name of the animal.
- **position**: The current position of the animal in the race. Initialized at 0.
- **race_distance**: The total distance of the race. This is used to determine when the race is finished.

## Methods

- **`move()`**: This method is to be implemented in each subclass to determine how the animal moves. In the **Animal** class, it's left empty as a placeholder.
- **`report_position()`**: This method prints out the current position of the animal. You can print something like "The $name_of_object is at position $position_of_object"
- **`has_finished()`**: This method checks if the animal has reached or exceeded the race distance, and thus finished the race. Return a boolean

# Task 2: The hare

This is a subclass of Animal. It inherits all the attributes and methods of Animal, but also has its own unique behavior.

## Polymorphism

Rewrite the `move()` method to follow these behaviors:

- The hare moves faster (between 5 and 10 kilometers per day) but there's a 50% chance it doesn't move at all, simulating the hare's tendency to rest.
- Each day (each turn in our program), the hare will move a certain distance or rest, and its position will be updated accordingly.

Hints: for the resting time probability you can use random() and for the position increment you can get a random distance between 5 to 10 using randint()

# Task 2: The hare

Note: to inherit all the attributes from a superclass you can simply write

```
class Hare(Animal):
    def __init__(self, race_distance):
        super().__init__("Hare", race_distance)
```

Super() simply refers to the superclass (in this case the class animal) such that we can reuse its ___init___ method

If you do not want to use super():

```
class Hare(Animal):
    def __init__(self, race_distance):
        self.name = "Hare"
        self.position = 0
        self.race_distance = race_distance
```

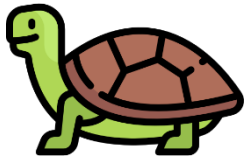
# Task 3: The tortoise

This is a subclass of Animal. It inherits all the attributes and methods of Animal, but also has its own unique behavior.

## Polymorphism

Rewrite the `move()` method to follow these behaviors:

- The tortoise moves slower (between 1 and 5 kilometers per day), but it moves every day, simulating the tortoise's steady pace.
- Each day (each turn in our program), the tortoise will move a certain distance, and its position will be updated accordingly.



# Task 4: Start the race

Now that everybody is ready, we can start the race

```
# Running the race
race_distance = 50
hare = Hare(race_distance)
tortoise = Tortoise(race_distance)

while not hare.has_finished() and not tortoise.has_finished():
    hare.move()
    tortoise.move()
    hare.report_position()
    tortoise.report_position()

if hare.has_finished() and tortoise.has_finished():
    print("It's a tie!")
elif hare.has_finished():
    print("The hare wins!")
else:
    print("The tortoise wins!")
```