**Title:** Zoo Algorithm

**Author:** Shaun Clarke

**Goal:** This program mimics an animal database, allowing us to add and retrieve animal details.

**Steps:**

1. Import Dict from typing for type hinting
2. Define a parent class Animal:
   1. This parent class has the basics needed to create an animal.
   2. Define a class variable \_\_zoo\_keeper, which is a dict that will hold animals.
   3. The construct or takes the animal’s name, species, and animal(type of animal) and initializes the following:
      1. The animals name.
      2. The animals species
      3. Calls a method that adds the created animal to the zoo\_keeper dict.
   4. Define a method \_\_add\_animal:
      1. This method adds the animal to the zoo dict.
   5. Define a method select\_animal(self, animal: str) -> object:
      1. This method allows us to select an animal from the zoo dict.
      2. For loop:
         1. If the animal type in the dict matches what we are looking for:
            1. Return that animal object.
   6. Define a method make\_sound(self) -> str:
      1. This method returns the sound the animal makes.
      2. Return sound
   7. Define a method info(self) -> str:
      1. This method returns the animal attributes formatted for output.
      2. Return info as a formatted string.
   8. Define a method get\_all\_animals(self) -> Dict:
      1. This method returns the \_\_zoo\_keeper dict with all animal objects.
         1. Return \_\_zoo\_keeper
   9. Define a method \_\_str\_\_(self) -> str:
      1. This magic method returns the output of the info method. When the object is printed.
      2. Return self.info()