LAB: 02

Task 1: Write a class of Dog, each dog must be of species type mammal. Each dog has its name and age. The class can have method for description () and sound () which dog produces. Create an object and perform some operations.

```
class Dog:
  # Class Attribute
  species = 'mammal'
  # Initializer / Instance Attributes
  def __init__(self, name, age):
     self.name = name
     self.age = age
  # instance method
  def description(self):
     return "{} is {} years old".format(self.name, self.age)
  # instance method
  def speak(self, sound):
     return "{} says {}".format(self.name, sound)
# Instantiate the Dog object
razer = Dog("Razer", 6)
# call our instance methods
print(razer.description())
print(razer.speak("Woof Woof"))
```

```
Razer is 6 years old
Razer says Woof Woof
>
```

Task 2: Write a class of Dog, each dog must be of species type mammal. Each dog has its name and age. The class can have method for description () and sound () which dog produces. Now this time you need to create two sub classes of Dogs one is Bull Dog and other is Russell Terrier Create few objects and perform some operations including the inheritance.

```
class Dog:
    species = 'mammal'
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def description(self):
        return "{} is {} years old".format(self.name, self.age)
    def speak(self, sound):
        return "{} says {}".format(self.name, sound)
class RussellTerrier(Dog):
    def run(self, speed):
        return "{} runs {}".format(self.name, speed)
class Bulldog(Dog):
    def run(self, speed):
        return "{} runs {}".format(self.name, speed)
thunder = Bulldog("Thunder", 9)
print(thunder.description())
print(thunder.run("slowly"))
spinter = Bulldog("Spinter", 12)
print(spinter.description())
print(spinter.run("fast"))
```

```
Thunder is 9 years old
Thunder runs slowly
Spinter is 12 years old
Spinter runs fast
>
```

Task 3: Extending question number 2, now we need to check that either the different dog classes and their objects link with each other or not. In this case we need to create a method to find either it's an instance of each other objects or not.

```
# Parent class
class Dog:
    species = 'mammal'
    def __init__(self, name, age):
     self.name = name
     self.age = age
    def description(self):
     return "{} is {} years old".format(self.name, self.age)
    def speak(self, sound):
     return "{} says {}".format(self.name, sound)
class RussellTerrier(Dog):
    def run(self, speed):
     return "{} runs {}".format(self.name, speed)
class Bulldog(Dog):
    def run(self, speed):
     return "{} runs {}".format(self.name, speed)
thunder = Bulldog("Thunder", 9)
print(thunder.description())
```

```
Thunder is 9 years old
>
```