**Name**: Steve Hommy

**Pair: -**

**Amount of completed tasks: 7**

**Which tasks were left undone or incomplete: 0**

Self-assessment:

This exercise was easy/difficult/ok/etc. for me because…

This exercise was easy for me because I have worked with these before. Only one thing gave me a headache and it was finding the txt file path.

Doing this exercise, I learned…

Using os module

I am still wondering…

-

I understood/did not understand that… ; I did/did not know that… ; I did/did not manage to do…

I pretty much understood everything

## Test report

Write the test report yourself to each coding task (task number, input/action, desired output and then the testing evidence (actual output)). Add rows if necessary. Include answers to theoretical questions and pseudocode to this return document as well in addition to code screen captures. Actual output can be a screen capture of the terminal showing the output.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Input / action** | **Desired output** | **Actual output (use red color if desired output != actual output)** |
| **3** | <Run Program> | Here are our students:    First name: Steve  Last name: Hommy  Student ID: 1  First name: Jhon  Last name: Snow  Student ID: 2  Let's give our student a pet  Species of the pet: Dog  Name of the pet: Brak  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Owner of the pet: Jhon  Would you like to remove one of these pets? yes  Which student pet would you like to remove? Steve  There are 2 pets in a list.  How many would you like to remove? 1  Species of the pet: Dog  Name of the pet: Brak  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Frist index is 0. Which pet would you like to remove from the list? 0  Here are the pets that remain  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Owner of the pet: Jhon | Here are our students:    First name: Steve  Last name: Hommy  Student ID: 1  First name: Jhon  Last name: Snow  Student ID: 2  Let's give our student a pet  Species of the pet: Dog  Name of the pet: Brak  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Owner of the pet: Jhon  Would you like to remove one of these pets? yes  Which student pet would you like to remove? Steve  There are 2 pets in a list.  How many would you like to remove? 1  Species of the pet: Dog  Name of the pet: Brak  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Frist index is 0. Which pet would you like to remove from the list? 0  Here are the pets that remain  Species of the pet: Cat  Name of the pet: Snuf  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Owner of the pet: Jhon |
| **5** | <Run Program> | Here are our students:    First name: Steve  Last name: Hommy  Student ID: 1  First name: Jhon  Last name: Snow  Student ID: 2  Let's give our student a pet and a car  Steve owns these:  Species of the pet: Dog  Name of the pet: Brak  Size of the pet: 150  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Size of the pet: 100  Owner of the pet: Steve  Brand: Toyota  Model: Avensis  Boot size: 200  Owner: Steve  Jhon owns these:  Species of the pet: Rabbit  Name of the pet: Snug  Size of the pet: 50  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Size of the pet: 10  Owner of the pet: Jhon  Brand: VW  Model: Golf  Boot size: 150  Owner: Jhon  Would you like to remove one of these pets? no  Here are the pets that remain  Species of the pet: Dog  Name of the pet: Brak  Size of the pet: 150  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Size of the pet: 100  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Size of the pet: 50  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Size of the pet: 10  Owner of the pet: Jhon  Let's check if pets will fit into the car  Steve pets won't fit we need a trailer  Jhon pets will fit | Here are our students:    First name: Steve  Last name: Hommy  Student ID: 1  First name: Jhon  Last name: Snow  Student ID: 2  Let's give our student a pet and a car  Steve owns these:  Species of the pet: Dog  Name of the pet: Brak  Size of the pet: 150  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Size of the pet: 100  Owner of the pet: Steve  Brand: Toyota  Model: Avensis  Boot size: 200  Owner: Steve  Jhon owns these:  Species of the pet: Rabbit  Name of the pet: Snug  Size of the pet: 50  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Size of the pet: 10  Owner of the pet: Jhon  Brand: VW  Model: Golf  Boot size: 150  Owner: Jhon  Would you like to remove one of these pets? no  Here are the pets that remain  Species of the pet: Dog  Name of the pet: Brak  Size of the pet: 150  Owner of the pet: Steve  Species of the pet: Cat  Name of the pet: Snuf  Size of the pet: 100  Owner of the pet: Steve  Species of the pet: Rabbit  Name of the pet: Snug  Size of the pet: 50  Owner of the pet: Jhon  Species of the pet: Fish  Name of the pet: Blub  Size of the pet: 10  Owner of the pet: Jhon  Let's check if pets will fit into the car  Steve pets won't fit we need a trailer  Jhon pets will fit |
| **6** | <Run Program>  <User input> | Uruguay  Give capital:  Wrong answer the correct answer is Montevideo  Philippines  Give capital:  Wrong answer the correct answer is Manila  Samoa  Give capital:  Wrong answer the correct answer is Apia  Mali  Give capital:  Wrong answer the correct answer is Bamako  Liechtenstein  Give capital:  Wrong answer the correct answer is Vaduz  Singapore  Give capital:  Wrong answer the correct answer is Singapore  Kyrgyzstan  Give capital:  Give capital:  Wrong answer the correct answer is Victoria  Azerbaijan  Give capital:  Wrong answer the correct answer is Baku  Brazil  Give capital:  Wrong answer the correct answer is Brasilia  Score:  0/10  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming>  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming> & C:/Users/steve/AppData/Local/Programs/Python/Python38-32/python.exe c:/Users/steve/OneDrive/Desktop/Object-Oriented-Programming/Exercise7/exercise7/main.py  Our first car is:  Brand: Honda  Brand: Tesla  Tyre: Nokia  Body style: Sedan  0 to 100 in: 4.5 seconds  Electric power: 250W  Battery size: 1000 000A  Honda will reach 0 to 100 in 8.5 seconds  Tesla will reach 0 to 100 in 4.5 seconds  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming>  > & C:/Users/steve/AppData/Local/Programs/Python/Python38-32/python.exe c:/Users/steve/OneDrive/Desktop/Object-Oriented-Programming/Exercise7/exercise6/main.py  Ghana  Give capital: Accra  Correct!  Comoros  Give capital: Don't know  Wrong answer the correct answer is: Moroni  Kyrgyzstan  Give capital: Bishek  Wrong answer the correct answer is: Bishkek  Finland  Give capital: Helsinki  Correct!  Norway  Give capital: Oslo  Correct!  Nicaragua  Give capital: Don't know  Wrong answer the correct answer is: Managua  Comoros  Give capital: Don't know  Wrong answer the correct answer is: Moroni  Belarus  Give capital: Minsk  Correct!  Liechtenstein  Give capital: Vaduz  Correct!  Luxembourg  Give capital: Luxembourg  Correct!  Score:  6/10 | Uruguay  Give capital:  Wrong answer the correct answer is Montevideo  Philippines  Give capital:  Wrong answer the correct answer is Manila  Samoa  Give capital:  Wrong answer the correct answer is Apia  Mali  Give capital:  Wrong answer the correct answer is Bamako  Liechtenstein  Give capital:  Wrong answer the correct answer is Vaduz  Singapore  Give capital:  Wrong answer the correct answer is Singapore  Kyrgyzstan  Give capital:  Give capital:  Wrong answer the correct answer is Victoria  Azerbaijan  Give capital:  Wrong answer the correct answer is Baku  Brazil  Give capital:  Wrong answer the correct answer is Brasilia  Score:  0/10  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming>  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming> & C:/Users/steve/AppData/Local/Programs/Python/Python38-32/python.exe c:/Users/steve/OneDrive/Desktop/Object-Oriented-Programming/Exercise7/exercise7/main.py  Our first car is:  Brand: Honda  Brand: Tesla  Tyre: Nokia  Body style: Sedan  0 to 100 in: 4.5 seconds  Electric power: 250W  Battery size: 1000 000A  Honda will reach 0 to 100 in 8.5 seconds  Tesla will reach 0 to 100 in 4.5 seconds  PS C:\Users\steve\OneDrive\Desktop\Object-Oriented-Programming>  > & C:/Users/steve/AppData/Local/Programs/Python/Python38-32/python.exe c:/Users/steve/OneDrive/Desktop/Object-Oriented-Programming/Exercise7/exercise6/main.py  Ghana  Give capital: Accra  Correct!  Comoros  Give capital: Don't know  Wrong answer the correct answer is: Moroni  Kyrgyzstan  Give capital: Bishek  Wrong answer the correct answer is: Bishkek  Finland  Give capital: Helsinki  Correct!  Norway  Give capital: Oslo  Correct!  Nicaragua  Give capital: Don't know  Wrong answer the correct answer is: Managua  Comoros  Give capital: Don't know  Wrong answer the correct answer is: Moroni  Belarus  Give capital: Minsk  Correct!  Liechtenstein  Give capital: Vaduz  Correct!  Luxembourg  Give capital: Luxembourg  Correct!  Score:  6/10 |
| **7** | <Run Program> | Our first car is:  Brand: Honda  Tyre: Continental  Body style: Hatchback  0 to 100 in: 8.5 seconds  Engine size: 1.6l  Tank size: 100l  Our second car is:  Brand: Tesla  Tyre: Nokia  Body style: Sedan  0 to 100 in: 4.5 seconds  Electric power: 250W  Battery size: 1000 000A  Honda will reach 0 to 100 in 8.5 seconds  Tesla will reach 0 to 100 in 4.5 seconds | Our first car is:  Brand: Honda  Tyre: Continental  Body style: Hatchback  0 to 100 in: 8.5 seconds  Engine size: 1.6l  Tank size: 100l  Our second car is:  Brand: Tesla  Tyre: Nokia  Body style: Sedan  0 to 100 in: 4.5 seconds  Electric power: 250W  Battery size: 1000 000A  Honda will reach 0 to 100 in 8.5 seconds  Tesla will reach 0 to 100 in 4.5 seconds |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Answer the following questions.

a. What does polymorphism (in object-oriented programming) mean? Also give a short (coding) example, e.g. google for examples).

**Polymorphism is the method in an object-oriented programming language that performs different things as per the object’s class, which calls it. With Polymorphism, a message is sent to multiple class objects, and every object responds appropriately according to the properties of the class.**

**class** Bird:

**def** intro(self):

**print**("There are many types of birds.")

**def** flight(self):

**print**("Most of the birds can fly but some cannot.")

**class** sparrow(Bird):

**def** flight(self):

    print("Sparrows can fly.")

**class** ostrich(Bird):

**def** flight(self):

**print**("Ostriches cannot fly.")

obj\_bird **=** Bird()

obj\_spr **=** sparrow()

obj\_ost **=** ostrich()

obj\_bird.intro()

obj\_bird.flight()

obj\_spr.intro()

obj\_spr.flight()

obj\_ost.intro()

obj\_ost.flight()

**Output:**

**There are many types of birds.**

**Most of the birds can fly but some cannot.**

**There are many types of birds.**

**Sparrows can fly.**

**There are many types of birds.**

**Ostriches cannot fly.**

b. What is a class variable and how are they used?

**A class variable defines a specific attribute or property for a class and may be referred to as a member variable or static member variable. They are associated with the class, rather than with any object. Every instance of the class shares a class variable, which is in one fixed location in memory. Any object can change the value of a class variable, but class variables can also be manipulated without creating an instance of the class**

c. What is an instance variable and how is it different from the class variable?

**Instance variables are owned by instances of the class. This means that for each object or instance of a class, the instance variables are different. Unlike class variables, instance variables are defined within methods.**

d. What is a UML sequence diagram used for?

**A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.**

e. What is a lifeline in UML sequence diagrams?

**A lifeline represents an individual participant in a sequence diagram. A lifeline will usually have a rectangle containing its object name. If its name is "self", that indicates that the lifeline represents the classifier which owns the sequence diagram.**

2. More theory tasks

a. Multiple choice

i. In an inheritance relationship, the \_\_\_\_\_\_ is the general class.

1. Child class

2. Subclass

3. Superclass

4. Specialized class

ii. In an inheritance relationship, the \_\_\_\_\_\_ is the specialized class:

1. Superclass

2. Master class

3. Parent class

4. Subclass

iii. Let’s say we have two classes in our program: BankAccount and SavingsAccount. Which one of them would most likely be the subclass?

1. BankAccount

2. SavingsAccount

3. Neither of them

4. Both of them.

iv. Which one of the options you will use if you want to check whether an object is an instance of a class?

1. The instance operator

2. The is\_object\_of function

3. The isinstance function

4. There is not a way to check that at all.

v. Which one of the UML diagrams is a behavioral diagram?

1. Class diagram

2. Sequence diagram

3. Object diagram

4. Deployment diagram

vi. Which one of the UML diagrams is a structural diagram?

1. Use case diagram

2. State machine diagram

3. Activity diagram

4. Composite structure diagram

vii. In UML class diagrams, what does the notation \* mean.

1. Multiplication operation

2. Power of operation

3. Multiplicity 0..n

4. Multiplicity 0..1

b. True or false?

i. It is not possible to call a superclass’s \_\_init\_\_ method from a subclass’s \_\_init\_\_ method.

False

ii. A subclass never inherits any methods or attributes from the superclass.

False

iii. A superclass can inherit methods from subclass, if they have been denoted with pass\_to\_super function.

False

iv. In a subclass it is possible to have methods and attributes in addition to those that the subclass inherits from superclass.

True

v. In Python, multiple inheritance does not exist.

False

vi. Aggregation and composition shall never be used in UML class diagrams.

False

vii. Aggregation and composition mean exactly the same thing in UML class diagrams.

False

A picture containing graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

# Code

# File name: pet.py

# Author: Steve Hommy

# Description: Create a Pet Class

class Pet:

    def \_\_init\_\_(self, species, name):

        self.\_\_species = species

        self.\_\_name = name

        self.\_\_owner = None

    def set\_species(self):

        self.\_\_species = input("Give species for pet: ")

    def get\_species(self):

        return self.\_\_species

    def set\_name(self):

        self.\_\_name = input("Give name for pet: ")

    def get\_name(self):

        return self.\_\_name

    def set\_owner(self, owner):

        self.\_\_owner = owner

    def get\_owner(self):

        return self.\_\_owner

    def \_\_str\_\_(self):

        return f"""

        Species of the pet: {self.\_\_species}

        Name of the pet: {self.\_\_name}

        Owner of the pet: {self.\_\_owner}"""

# File name: student.py

# Author: Steve Hommy

# Description: Create a Student Class

class Student:

    def \_\_init\_\_(self, first\_name, last\_name, student\_ID):

        self.\_\_first\_name = first\_name

        self.\_\_last\_name = last\_name

        self.\_\_student\_ID = student\_ID

        self.\_\_pets = []

    def set\_first\_name(self):

        self.\_\_first\_name = input("Student first name: ")

    def get\_first\_name(self):

        return self.\_\_first\_name

    def set\_last\_name(self):

        self.\_\_last\_name = input("Student last name: ")

    def get\_last\_name(self):

        return self.\_\_last\_name

    def set\_student\_ID(self):

        self.\_\_student\_ID = input("Student ID: ")

    def get\_student\_ID(self):

        return self.\_\_student\_ID

    def add\_pets(self, add\_pet):

        try:

            if add\_pet.get\_owner() == None:

                self.\_\_pets.append(add\_pet)

                add\_pet.set\_owner(self.\_\_first\_name)

            else:

                print("Pet has an owner already")

        except ValueError:

            return print("Wrong value given")

    def remove\_pets(self):

        for i in range(int(input("\nThere are " + str(len(self.\_\_pets)) + " pets in a list.\nHow many would you like to remove? "))):

            for pets in self.\_\_pets:

                print(pets)

            self.\_\_pets.pop(int(input("\nFrist index is 0. Which pet would you like to remove from the list? ")))

    def print\_pets(self):

        for pets in self.\_\_pets:

            print(pets)

    def \_\_str\_\_(self):

        return f"""

        First name: {self.\_\_first\_name}

        Last name: {self.\_\_last\_name}

        Student ID: {self.\_\_student\_ID}

        """

# File: main.py

# Author: Steve Hommy

# Description: Main function

from student import Student

from pet import Pet

def main():

    student1 = Student("Steve", "Hommy", 1)

    student2 = Student("Jhon", "Snow", 2)

    pet1 = Pet("Dog", "Brak")

    pet2 = Pet("Cat", "Snuf")

    pet3 = Pet("Rabbit", "Snug")

    pet4 = Pet("Fish", "Blub")

    print("Here are our students:\n", student1, student2)

    print("\nLet's give our student a pet")

    student1.add\_pets(pet1)

    student1.add\_pets(pet2)

    student2.add\_pets(pet3)

    student2.add\_pets(pet4)

    student1.print\_pets()

    student2.print\_pets()

    question = input("\nWould you like to remove one of these pets? ")

    if question == "yes":

        question2 = input("\nWhich student pet would you like to remove? ")

        if question2 == "Steve":

            student1.remove\_pets()

        elif question2 == "Jhon":

            student2.remove\_pets()

        else:

            print("\nHere are the pets that remain")

            student1.print\_pets()

            student2.print\_pets()

    print("\nHere are the pets that remain")

    student1.print\_pets()

    student2.print\_pets()

main()

# File name: car.py

# Author: Steve Hommy

# Description: Create a Car Class

class Car:

    def \_\_init\_\_(self, brand, model, boot\_size):

        self.\_\_brand = brand

        self.\_\_model = model

        self.\_\_boot\_size = boot\_size

        self.\_\_owner = None

    def set\_brand(self):

        self.\_\_brand = input("What brand is the car? ")

    def get\_brand(self):

        return self.\_\_brand

    def set\_model(self):

        self.\_\_model = input("What model is the car? ")

    def get\_model(self):

        return self.\_\_model

    def set\_boot\_size(self):

        self.\_\_boot\_size = int(input("What is the boot size? "))

    def get\_boot\_size(self):

        return self.\_\_boot\_size

    def set\_owner(self, owner):

        self.\_\_owner = owner

    def get\_owner(self):

        return self.\_\_owner

    def \_\_str\_\_(self):

        return f"""

        Brand: {self.\_\_brand}

        Model: {self.\_\_model}

        Boot size: {self.\_\_boot\_size}

        Owner: {self.\_\_owner}

        """

# File name: pet.py

# Author: Steve Hommy

# Description: Create a Pet Class

class Pet:

    def \_\_init\_\_(self, species, name, pet\_size):

        self.\_\_species = species

        self.\_\_name = name

        self.\_\_pet\_size = pet\_size

        self.\_\_owner = None

    def set\_species(self):

        self.\_\_species = input("Give species for pet: ")

    def get\_species(self):

        return self.\_\_species

    def set\_name(self):

        self.\_\_name = input("Give name for pet: ")

    def get\_name(self):

        return self.\_\_name

    def set\_pet\_size(self):

        self.\_\_pet\_size = int(input("Size of the pet"))

    def get\_pet\_size(self):

        return self.\_\_pet\_size

    def set\_owner(self, owner):

        self.\_\_owner = owner

    def get\_owner(self):

        return self.\_\_owner

    def \_\_str\_\_(self):

        return f"""

        Species of the pet: {self.\_\_species}

        Name of the pet: {self.\_\_name}

        Size of the pet: {self.\_\_pet\_size}

        Owner of the pet: {self.\_\_owner}"""

# File name: student.py

# Author: Steve Hommy

# Description: Create a Student Class

class Student:

    def \_\_init\_\_(self, first\_name, last\_name, student\_ID):

        self.\_\_first\_name = first\_name

        self.\_\_last\_name = last\_name

        self.\_\_student\_ID = student\_ID

        self.\_\_pets = []

        self.\_\_cars = []

    def set\_first\_name(self):

        self.\_\_first\_name = input("Student first name: ")

    def get\_first\_name(self):

        return self.\_\_first\_name

    def set\_last\_name(self):

        self.\_\_last\_name = input("Student last name: ")

    def get\_last\_name(self):

        return self.\_\_last\_name

    def set\_student\_ID(self):

        self.\_\_student\_ID = input("Student ID: ")

    def get\_student\_ID(self):

        return self.\_\_student\_ID

    def add\_pets(self, add\_pet):

        try:

            if add\_pet.get\_owner() == None:

                self.\_\_pets.append(add\_pet)

                add\_pet.set\_owner(self.\_\_first\_name)

            else:

                print("Pet has an owner already")

        except ValueError:

            return print("Wrong value given")

    def remove\_pets(self):

        for i in range(int(input("\nThere are " + str(len(self.\_\_pets)) + " pets in a list.\nHow many would you like to remove? "))):

            for pets in self.\_\_pets:

                print(pets)

            self.\_\_pets.pop(int(input("\nFrist index is 0. Which pet would you like to remove from the list? ")))

    def print\_pets(self):

        for pets in self.\_\_pets:

            print(pets)

    def add\_cars(self, add\_car):

        if len(self.\_\_cars) < 1:

            self.\_\_cars.append(add\_car)

            add\_car.set\_owner(self.\_\_first\_name)

        else:

            print("You already own 1 car")

    def remove\_car(self):

        self.\_\_cars.clear()

    def print\_cars(self):

        for cars in self.\_\_cars:

            print(cars)

    def \_\_str\_\_(self):

        return f"""

        First name: {self.\_\_first\_name}

        Last name: {self.\_\_last\_name}

        Student ID: {self.\_\_student\_ID}

        """

# File: main.py

# Author: Steve Hommy

# Description: Main function

from student import Student

from pet import Pet

from car import Car

def main():

    student1 = Student("Steve", "Hommy", 1)

    student2 = Student("Jhon", "Snow", 2)

    pet1 = Pet("Dog", "Brak", 150)

    pet2 = Pet("Cat", "Snuf", 100)

    pet3 = Pet("Rabbit", "Snug", 50)

    pet4 = Pet("Fish", "Blub", 10)

    car1 = Car("Toyota", "Avensis", 200)

    car2 = Car("VW", "Golf", 150)

    print("Here are our students:\n", student1, student2)

    print("\nLet's give our student a pet and a car")

    student1.add\_pets(pet1)

    student1.add\_pets(pet2)

    student1.add\_cars(car1)

    student2.add\_pets(pet3)

    student2.add\_pets(pet4)

    student2.add\_cars(car2)

    print(student1.get\_first\_name() + " owns these:")

    student1.print\_pets()

    student1.print\_cars()

    print(student2.get\_first\_name() + " owns these:")

    student2.print\_pets()

    student2.print\_cars()

    question = input("\nWould you like to remove one of these pets? ")

    if question == "yes":

        question2 = input("\nWhich student pet would you like to remove? ")

        if question2 == "Steve":

            student1.remove\_pets()

        elif question2 == "Jhon":

            student2.remove\_pets()

        else:

            print("\nHere are the pets that remain")

            student1.print\_pets()

            student2.print\_pets()

    print("\nHere are the pets that remain")

    student1.print\_pets()

    student2.print\_pets()

    print("\nLet's check if pets will fit into the car")

    if car1.get\_boot\_size() >= pet1.get\_pet\_size() + pet2.get\_pet\_size():

        print(student1.get\_first\_name() + " pets will fit")

    else:

        print(student1.get\_first\_name() + " pets won't fit we need a trailer")

    if car2.get\_boot\_size() >= pet3.get\_pet\_size() + pet4.get\_pet\_size():

        print(student2.get\_first\_name() + " pets will fit")

    else:

        print(student2.get\_first\_name() + " pets won't fit we need a trailer")

main()

# File: main.py

# Author: Steve Hommy

# Description: Main function

import random

filename = "Exercise7/exercise6/capitals.txt"

dictionary = {}

try:

    with open(filename) as file:

        for line in file:

            (key, value) = line.split()

            dictionary[key] = value

except FileNotFoundError:

    msg = "Sorry, the file " + filename + " does not exist.\n"

    print(msg)

while True:

    points = 0

    for i in range(10):

        country, capital = random.choice(list(dictionary.items()))

        print(country)

        answer = input("Give capital: ")

        if answer == dictionary[country]:

            print("Correct!\n")

            points += 1

        else:

            print("Wrong answer the correct answer is:", dictionary[country])

            print()

    print("Score:\n" + str(points) + "/10")

    break

# File name: electricVehicle.py

# Author: Steve Hommy

# Description: Inherit Vehicle Class and creating ElectricVehicle Class

from vehicle import Vehicle

class ElectricVehicle(Vehicle):

    def \_\_init\_\_(self, brand, tyre, body\_style, zero\_to\_hundred, electric\_power, battery\_size):

        Vehicle.\_\_init\_\_(self, brand, tyre, body\_style, zero\_to\_hundred)

        self.\_\_electric\_power = electric\_power

        self.\_\_battery\_size = battery\_size

    def \_\_str\_\_(self):

        return super().\_\_str\_\_() + f"""Electric power: {self.\_\_electric\_power}

        Battery size: {self.\_\_battery\_size}

        """

    def set\_electric\_power(self, electric\_power):

        self.\_\_electric\_power = electric\_power

    def set\_battery\_size(self, battery\_size):

        self.\_\_battery\_size = battery\_size

    def get\_electric\_power(self):

        return self.\_\_electric\_power

    def get\_battery\_size(self):

        return self.\_\_battery\_size

# File name: petrolVehicle.py

# Author: Steve Hommy

# Description: Inherit Vehicle Class and creating PetrolVehicle Class

from vehicle import Vehicle

class PetrolVehicle(Vehicle):

    def \_\_init\_\_(self, brand, tyre, body\_style, zero\_to\_hundred, engine\_size, tank\_size):

        Vehicle.\_\_init\_\_(self, brand, tyre, body\_style, zero\_to\_hundred)

        self.\_\_engine\_size = engine\_size

        self.\_\_tank\_size = tank\_size

    def \_\_str\_\_(self):

        return super().\_\_str\_\_() + f"""Engine size: {self.\_\_engine\_size}

        Tank size: {self.\_\_tank\_size}

        """

    def set\_engine\_size(self, engine\_size):

        self.\_\_engine\_size = engine\_size

    def set\_tank\_size(self, tank\_size):

        self.\_\_tank\_size = tank\_size

    def get\_engine\_size(self):

        return self.\_\_engine\_size

    def get\_tank\_size(self):

        return self.\_\_tank\_size

# File name: vehicleClass.py

# Author: Steve Hommy

# Description: Create a Vehicle Class

class Vehicle:

    def \_\_init\_\_(self, brand, tyre, body\_style, zero\_to\_hundred):

        self.\_\_brand = brand

        self.\_\_tyre = tyre

        self.\_\_body\_style = body\_style

        self.\_\_zero\_to\_hundred = float(zero\_to\_hundred)

    def \_\_str\_\_(self):

        return f"""

        Brand: {self.\_\_brand}

        Tyre: {self.\_\_tyre}

        Body style: {self.\_\_body\_style}

        0 to 100 in: {self.\_\_zero\_to\_hundred} seconds

        """

    def set\_brand(self, brand):

        self.\_\_brand = brand

    def set\_tyre(self, tyre):

        self.\_\_tyre = tyre

    def set\_body\_style(self, body\_style):

        self.\_\_body\_style = body\_style

    def set\_zero\_to\_hundred(self, zero\_to\_hundred):

        self.\_\_zero\_to\_hundred = zero\_to\_hundred

    def get\_brand(self):

        return self.\_\_brand

    def get\_tyre(self):

        return self.\_\_tyre

    def get\_body\_style(self):

        return self.\_\_body\_style

    def get\_zero\_to\_hundred(self):

        return self.\_\_zero\_to\_hundred

# File: main.py

# Author: Steve Hommy

# Description: Main function

from petrolVehicle import PetrolVehicle

from electricVehicle import ElectricVehicle

def main():

    honda = PetrolVehicle("Honda", "Continental", "Hatchback", 8.5, "1.6l", "100l")

    tesla = ElectricVehicle("Tesla", "Nokia", "Sedan", 4.5, "250W", "1000 000A")

    print("Our first car is:", honda)

    print("Our second car is:", tesla)

    how\_fast\_dict = {

        honda.get\_brand(): honda.get\_zero\_to\_hundred(),

        tesla.get\_brand(): tesla.get\_zero\_to\_hundred()

    }

    for key in how\_fast\_dict:

        print(key, "will reach 0 to 100 in", how\_fast\_dict[key], "seconds")

main()