# *Introduction to Programming (ITAS 185)*

# *Lab 7 – More Classes*

Date due: **Fri, November 3, 2023, by 14:00**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

1. Create simple Python classes
2. Work with inheritance, polymorphism and abstraction in Python classes

**To be handed in:**

1. The ***username185L07*** folder should be zipped and uploaded to the ITAS Portal. ***Username*** isyour logon username (mine would be allan.mcdonald).

**To start:**

1. Create a folder called ***username*185L07** that you will use for ALL the files in the lab.
2. Open this **FOLDER** in VS Code. Make sure you open the folder and not just the files.

# Vehicles

**To Do**:

## Create a class file called Vehicle.py and write a class named Vehicle that has the following attributes:

### colour: The colour field is a string attribute that holds the colour.

### manufacturer: A string for the manufacturer of the vehicle.

### speed: a float of the current speed of the vehicle.

### top\_speed: a float of the top speed of the vehicle (the fastest it can go).

### year: an int of the year of the vehicle.

## Add a constructor method (\_\_init\_\_) that is passed all the attributes except the speed. Initialize all the attributes to the values passed. Initialize the speed to zero (0.0).

## Add a method called accelerate that increases the speed by the amount passed as a parameter or default 3. If the speed would exceed the top\_speed, set the speed to top\_speed.

## Add a method called brake that decreases the speed by the amount passed or default 2. If the speed would be less than 0, set the speed to 0.

## Create the \_\_str\_\_ method which returns a string in the form: {Year} {Colour} {manufacturer} is travelling {speed} kph with a maximum speed of {top\_speed} kph. For example, 2012 Blue Chevrolet is travelling 80 kph with a maximum speed of 100 kph.

## Create the \_\_eq\_\_ method that returns true if the top speed of the two Vehicles is the same.

## Create a class called Car which extends Vehicle. It has the extra attributes of:

### number\_of\_doors: int

### is\_electric: bool

## The Car class has methods:

|  |  |
| --- | --- |
| Method | Purpose |
| Constructor  (\_\_init\_\_) | One constructor that accepts all attributes and set the values accordingly by first calling the super method and then assigning the extra attributes |
| accelerate | Has NO parameters. Overrides the accelerate method of the Vehicle class and calls the Vehicle class accelerate method with the value 2 |
| \_\_str\_\_ | Calls the super method to get the initial string and adds:  It has *num\_of\_doors* doors and is/is not electric onto the end of the string. For example,  2018 blue Mercedes is travelling 120 kph with a maximum speed of 180 kph. It has 5 doors and is not electric. |

## Create a class called Transport which extends Vehicle. It has the extra attribute of:

## payload: float

|  |  |
| --- | --- |
| Method | Purpose |
| Constructor | A constructor that accepts all attributes and set the values accordingly by first calling the super method and then assigning the extra attribute |
| accelerate | Has NO parameters. Overrides the accelerate method of the Vehicle class and calls the Vehicle class accelerate method with the value 1 |
| toString() | Calls the super method to get the initial string and adds:  It can carry a payload of *payload* tons to end of the string. For example,  1999 brown Mac is travelling 100 kph with a maximum speed of 160 kph. It can carry a payload of 300 tons. |

## Create a class called Pickup which extends Car.

|  |  |
| --- | --- |
| Method | Purpose |
| Constructors | One constructor that accepts all attributes and set the values accordingly by calling the super method. |
| accelerate | Has no parameters and calls the accelerate method of the Car class with the value 1.5 |

## Create a class called SportsCar which extends Car.

|  |  |
| --- | --- |
| Method | Purpose |
| Constructor | One constructor that accepts all attributes and set the values accordingly by calling the super method. |
| accelerate | Has no parameters and calls the accelerate method of the Car class with the value 4 |

## Create a test program called vehicle\_test.py which creates a list of vehicles. Create at least one vehicle of each type.

## Once the vehicles are created the user is prompted to select a vehicle and then accelerate or brake the vehicle. For brake, prompt the user for the amount of braking and call the brake method with the value. For accelerate call the accelerate method appropriately based on the vehicle type. To determine the vehicle type you can use the \_\_class\_\_.\_\_name\_\_ attribute.

## You need to determine test cases to show ALL functionality and to display the value of the vehicle (print the string returned to the console). When you show me the finished product, make sure you know what tests that you need to run to demonstrate all functionality. This time it is on YOU to determine the proper testing.

# To finish:

## 1. Show your completed lab to me, to be marked.

2. Zip your **entire** folder.

3. Submit the zip file to the ITAS portal.