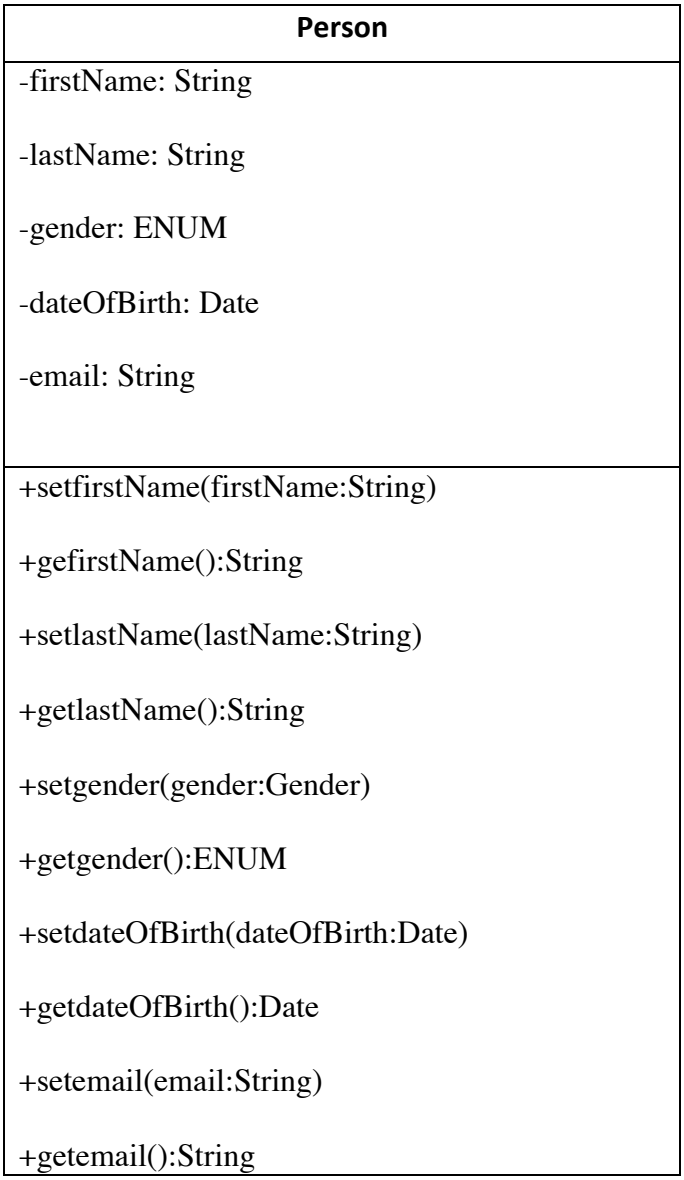
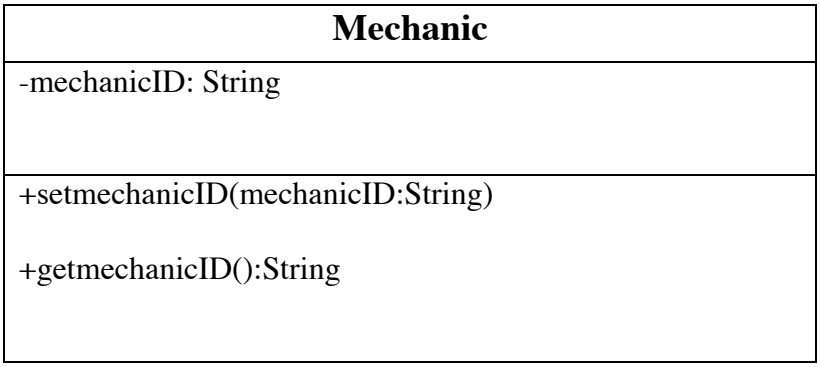
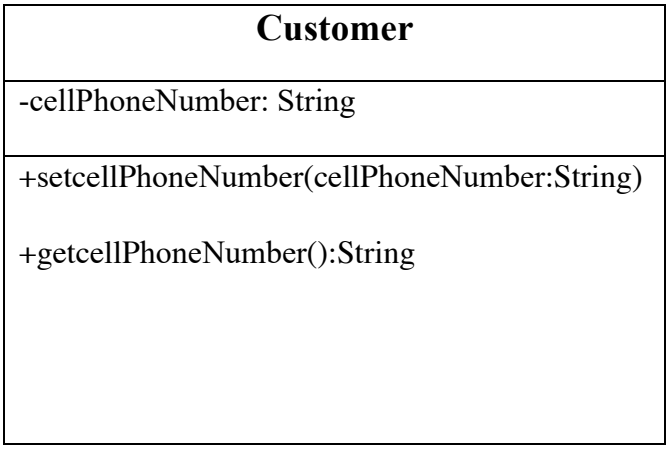
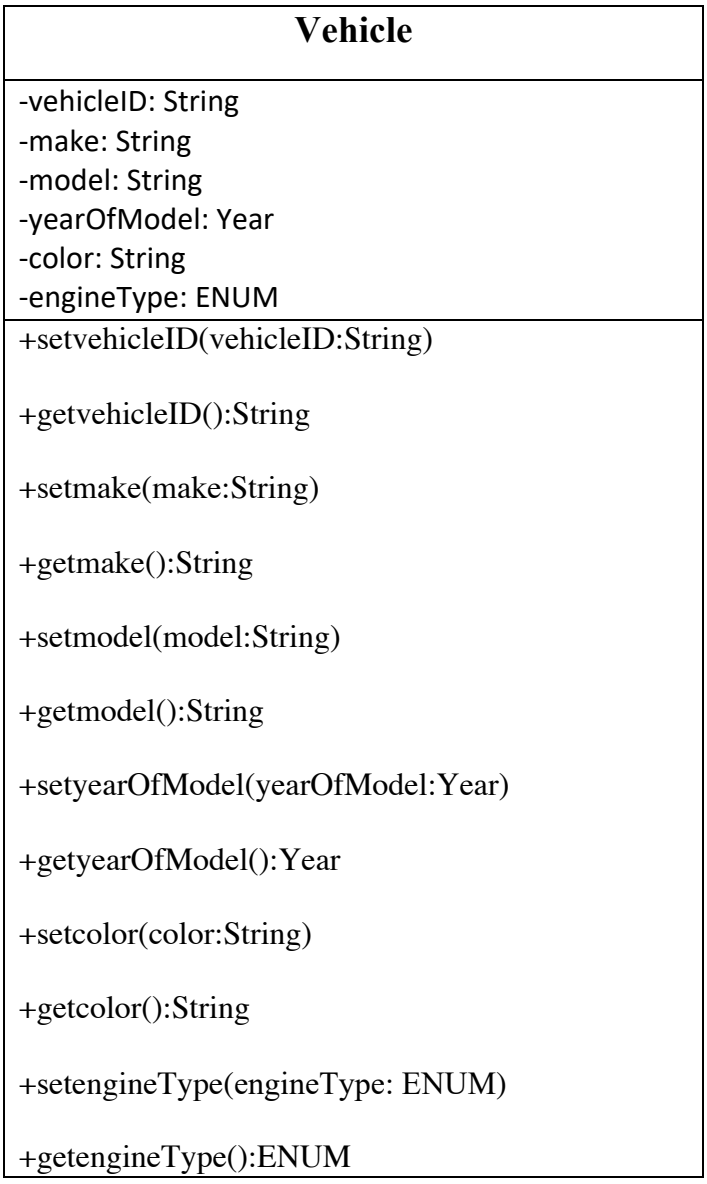
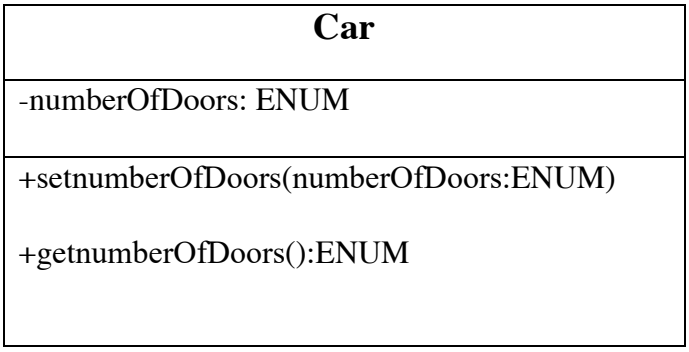

Person
-firstName: String
-lastName: String
-gender: ENUM
-dateOfBirth: Date
-email: String
+setfirstName(firstName:String)
+gefirstName():String
+setlastName(lastName:String)
+getlastName():String
+setgender(gender:Gender)
+getgender():ENUM
+setdateOfBirth(dateOfBirth:Date)
+getdateOfBirth():Date
+setemail(email:String)
+getemail():String


Mechanic
-mechanicID: String
+setmechanicID(mechanicID:String)
+getmechanicID():String


Customer
-cellPhoneNumber: String
+setcellPhoneNumber(cellPhoneNumber:String)
+getcellPhoneNumber():String


Vehicle
-vehicleID: String
-make: String
-model: String
-yearOfModel: Year
-color: String
-engineType: ENUM
+setvehicleID(vehicleID:String)
+getvehicleID():String
+setmake(make:String)
+getmake():String
+setmodel(model:String)
+getmodel():String
+setyearOfModel(yearOfModel:Year)
+getyearOfModel():Year
+setcolor(color:String)
+getcolor():String
+setengineType(engineType: ENUM)
+getengineType():ENUM


Car
-numberOfDoors: ENUM
+setnumberOfDoors(numberOfDoors:ENUM)
+getnumberOfDoors():ENUM

Service
-name: String -description: String -price: Float -approximateDuration: Float -numberOfMechanics: ENUM
+setname(name:String) +getname():String +setdescription(description:String) +getdescription():String +setprice(price: Float) +getprice():Float +setapproximateDuration(approximateDuration: Float) +getapproximateDuration():Float +setnumberOfMechanics(numberOfMechanics:ENUM) +getnumberOfMechanics():ENUM

I believe that these are 6 classes that will be necessary to print the data or information on the bill. In terms of class relationships. I have included 2 “is a” relationship which denotes inheritance. This relationship occurs between the class person and the classes customer and mechanic. Mechanic is a Person and Customer is a Person. Both mechanic and customer classes inherit all the attributes of person and have their own attributes. Moreover, the vehicle class and car class where the vehicle is the parent class and car is the child class. Car is a Vehicle. Car inherits all the attributes from vehicle and has its own attributes too!

Objects:

James: Customer
firstName = “James” lastName = “Jones” gender = Gender.Male dateOfBirth = [16/04/1992] email = “james.jones@gmail.com” cellPhoneNumber = “816-897-9862”

Hans: Mechanic
firstName = “Hans” lastName = “K” gender = Gender.Male dateOfBirth = [21/03/1989] email = “hans.k@gmail.com” mechanicID= “00012891456”

AD-89034: Car
vehicleID = “AD-89034” make = “Nissan” model = “Altima “ yearOfModel = [2014] color = “Silver” engineType = EngineType.Gasoline numberOfDoors = NumberOfDoors.FOUR

OilReplacement: Service
name = “Oil Replacement” description = “This is the process of removing old dirty oil from the vehicle and replacing it with clean oil” price = 120.00 approximateDuration = 0.75 numberOfMechanics = NumberOfMechanics.ONE

Code:

```
from enum import Enum

# Define Gender as an Enum
class Gender(Enum):
    MALE = 'Male'
    FEMALE = 'Female'

class Person: #define the class (this will be a parent class)
    # initialize the class objects with the specified attributes
    def __init__(self, firstName, lastName, gender, dateOfBirth, email):
        self._firstName = firstName
        self._lastName = lastName
        self._gender = gender
        self._dateOfBirth = dateOfBirth
        self._email = email

    # create setters and getters for each of the attributes defined above
    def setFirstName(self, firstName):
        self._firstName = firstName

    def getFirstName(self):
        return self._firstName

    def setLastName(self, lastName):
        self._lastName = lastName

    def getLastName(self):
        return self._lastName

    def setGender(self, gender):
        self._gender = gender

    def getGender(self):
        return self._gender

    def setDateOfBirth(self, dateOfBirth):
        self._dateOfBirth = dateOfBirth

    def getDateOfBirth(self):
        return self._dateOfBirth

    def setEmail(self, email):
        self._email = email

    def getEmail(self):
        return self._email

    def __str__(self):
        return "First Name: " + self._firstName + ", Last Name: " +
self._lastName + ", Gender: " + str(self._gender) + ", Date of Birth: " +
str(self._dateOfBirth) + ", Email: " + self._email

class Customer(Person): #define the child class
    # initialize the class objects with the specified attributes
```

```
    def __init__(self, firstName, lastName, gender, dateOfBirth, email,
cellPhoneNumber):
        super().__init__(firstName, lastName, gender, dateOfBirth, email)
        self._cellPhoneNumber = cellPhoneNumber

    # create setters and getters for each of the attributes defined above
    def setCellPhoneNumber(self, cellPhoneNumber):
        self._cellPhoneNumber = cellPhoneNumber

    def getCellPhoneNumber(self):
        return self._cellPhoneNumber

    def __str__(self):
        return super().__str__() + ", Cell Phone Number: " +
self._cellPhoneNumber


class Mechanic(Person): #define the child class class
    # initialize the class objects with the specified attributes
    def __init__(self, firstName, lastName, gender, dateOfBirth, email,
mechanicID):
        super().__init__(firstName, lastName, gender, dateOfBirth, email)
        self._mechanicID = mechanicID

    # create setters and getters for each of the attributes defined above
    def setMechanicID(self, mechanicID):
        self._mechanicID = mechanicID

    def getMechanicID(self):
        return self._mechanicID

    def __str__(self):
        return super().__str__() + ", Mechanic ID: " + self._mechanicID

# Define EngineType as an Enum
class EngineType(Enum):
    Gasoline = 'Gasoline'
    Diesel = 'Diesel'
    Electric = 'Electric'
    Hybrid = 'Hybrid'

class Vehicle: #define the class (this will be a parent class)
    # initialize the class objects with the specified attributes
    def __init__(self, vehicleID, make, model, yearOfModel, color,
engineType):
        self._vehicleID = vehicleID
        self._make = make
        self._model = model
        self._yearOfModel = yearOfModel
        self._color = color
        self._engineType = engineType

    # create setters and getters for each of the attributes defined above
    def setVehicleID(self, vehicleID):
        self._vehicleID = vehicleID

    def getVehicleID(self):
```

```
        return self._vehicleID

    def setMake(self, make):
        self._make = make

    def getMake(self):
        return self._make

    def setModel(self, model):
        self._model = model

    def getModel(self):
        return self._model

    def setYearOfModel(self, yearOfModel):
        self._yearOfModel = yearOfModel

    def getYearOfModel(self):
        return self._yearOfModel

    def setColor(self, color):
        self._color = color

    def getColor(self):
        return self._color

    def setEngineType(self, engineType):
        self._engineType = engineType

    def getEngineType(self):
        return self._engineType

    def __str__(self):
        return "Vehicle ID: "+str(self._vehicleID)+", Make: "+self._make+",
Model: "+self._model+", Year of Model: "+str(self._yearOfModel)+", Color:
"+self._color+", Engine Type: "+self._engineType.value

#Define NumberOfDoors as enum
class NumberOfDoors(Enum):
    TWO = '2 Doors'
    FOUR = '4 Doors'

class Car(Vehicle): #define the child class class
    # initialize the class objects with the specified attributes
    def __init__(self, vehicleID, make, model, yearOfModel, color,
engineType, numberOfDoors):
        super().__init__(vehicleID, make, model, yearOfModel, color,
engineType)
        self._numberOfDoors = numberOfDoors

    # create setters and getters for each of the attributes defined above
    def setNumberOfDoors(self, numberOfDoors):
        self._numberOfDoors = numberOfDoors

    def getNumberOfDoors(self):
        return self._numberOfDoors
```

```
    def __str__(self):
        return "Vehicle ID: "+str(self._vehicleID)+", Make: "+self._make+",
Model: "+self._model+", Year of Model: "+str(self._yearOfModel)+", Color:
"+self._color+", Engine Type: "+self._engineType.value+", Number of Doors:
"+self._numberOfDoors.value

#Define NumberOfMechanics as enum
class NumberOfMechanics(Enum):
    ONE = 1
    TWO = 2
    THREE = 3
    FOUR = 4
    FIVE_OR_MORE = 5

class Service: #define the class
    # initialize the class objects with the specified attributes
    def __init__(self, name, description, price, approximateDuration,
numberOfMechanics):
        self._name = name
        self._description = description
        self._price = price
        self._approximateDuration = approximateDuration
        self._numberOfMechanics = numberOfMechanics

    # create setters and getters for each of the attributes defined above
    def setName(self, name):
        self._name = name

    def getName(self):
        return self._name

    def setDescription(self, description):
        self._description = description

    def getDescription(self):
        return self._description

    def setPrice(self, price):
        self._price = price

    def getPrice(self):
        return self._price

    def setApproximateDuration(self, approximateDuration):
        self._approximateDuration = approximateDuration

    def getApproximateDuration(self):
        return self._approximateDuration

    def setNumberOfMechanics(self, numberOfMechanics):
        self._numberOfMechanics = numberOfMechanics

    def getNumberOfMechanics(self):
        return self._numberOfMechanics

    def __str__(self):
        return "Name: "+self._name+", Description: "+self._description+",
```

```
Price: "+str(self._price)+", Approximate Duration:
"+str(self._approximateDuration)+" hours, Number of Mechanics Required:
"+str(self._numberOfMechanics.value)

#Customer object with the given attributes
customer = Customer(firstName="James", lastName="Jones",
gender=Gender.MALE.name, dateOfBirth="16/04/1992",
email="james.jones@gmail.com", cellPhoneNumber="816-897-9862")
print(customer)
#Mechanic object with the given attributes
mechanic = Mechanic(firstName="Hans", lastName="K", gender=Gender.MALE.name,
dateOfBirth=[21, 3, 1989], email="hans.k@gmail.com",
mechanicID="00012891456")
print(mechanic)
#Car object with the given attributes
car = Car(vehicleID="AD-89034", make="Nissan", model="Altima",
yearOfModel=[2014], color="Silver", engineType=EngineType.Gasoline,
numberOfDoors=NumberOfDoors.FOUR)
print(car)
#Service object with the given attributes
service = Service(name="Oil Replacement", description="This is the process of
removing old dirty oil from the vehicle and replacing it with clean oil",
price=120.00, approximateDuration=0.75,
numberOfMechanics=NumberOfMechanics.ONE)
print(service)
```