Week 7 Object-oriented Programming (OOP)

Download `bank_account.py` and `vehicle.py`.

Part 1 Bank Accounts

Task 1 Deposit

Add a method `deposit()` to deposit some money into the account.

```
>>> myAcc = BankAccount('Alan',1000)
>>> myAcc.showBalance()
Your balance is $1000.00
>>> myAcc.deposit(200)
>>> myAcc.deposit(400)
>>> myAcc.showBalance()
Your balance is $1600.00
```

Task 2 Secure Withdrawal

Add a control measure where a name must be provided for withdrawal and it must match the name in the account.

```
>>> myAcc = BankAccount('Alan',1000)
>>> myAcc.withdraw('Mary',100)
You are not authorized for this account
>>> myAcc.withdraw('Alan',10000)
Money not enough! You do not have $10000.00
0
>>> myAcc.withdraw('Alan',100)
100
>>> myAcc.showBalance()
Your balance is $900.00
```

Task 3 Interest

Add an attribute for the interest rate and initialize it in the **constructor**.

Implement a function `oneYearHasPassed()` which adds the interest gained after one year to the account.

```
>>> myAcc = BankAccount('Alan',1000,0.04)
>>> myAcc.showBalance()
Your balance is $1000.00
>>> myAcc.oneYearHasPassed()
>>> myAcc.showBalance()
Your balance is $1040.00
>>> myAcc.oneYearHasPassed()
>>> myAcc.showBalance()
Your balance is $1081.60
```

Task 4 Minimal Account

Create a `MinimalAccount` class that behaves the same as the normal `BankAccount` except that whenever one year has passed, if its balance is less than $1000, a $20 administration fee will be deducted from the account before the annual interest is gained. The balance will never be below zero.

```
>>> mySonAcc = MinimalAccount('John',40,0.04)
>>> mySonAcc.oneYearHasPassed()
>>> mySonAcc.showBalance()
Your balance is $20.80
>>> mySonAcc.oneYearHasPassed()
>>> mySonAcc.showBalance()
Your balance is $0.83
>>> mySonAcc.oneYearHasPassed()
>>> mySonAcc.showBalance()
Your balance is $0.00
```

Extra Tasks

- Method `transferTo()` in class `BankAccount`
  - Given another account, you can transfer your money to the account

    ```
    >>> myAcc.transferTo(myWifeAcc,500)
    ```

- Method `setupGiro()` in class `BankAccount`
  - Money will be deducted annually before interest is gained

    ```
    >>> myAcc = BankAccount('Alan',1100,0.04)
    >>> myAcc.setupGiro(40)
    >>> myAcc.setupGiro(60)
    >>> myAcc.oneYearHasPassed()
    >>> myAcc.showBalance()
    Your balance is $1040.00
    ```

- A new class `JointAccount`
  - An account has two owners and both owners can withdraw money from the account

Part 2 Vehicles

Task 1 Petrol

Let's try to be more realistic: every vehicle needs some petrol to move. Add a method `addPetrol(n)` (where?) that adds n litres of petrol into a vehicle. For every "move", the vehicle will consume one litre of petrol.

```
>>> myCar.addPetrol(2)
>>> myCar.move()
Move to (0, 80)
>>> myCar.move()
Move to (0, 160)
>>> myCar.move()
Out of petrol. Cannot move.
```

Task 2 Solar Tank

Implement a class `SolarTank` that does NOT need petrol. (Pulling your leg?) Just think about how you should organize your class structure.