Assignment: BankAccount

Objectives

* Practice writing classes

As we continue thinking about our banking application, we realize that it would be more accurate to assign a balance not to the user directly, but that in the real world, users have *accounts*, and *accounts* have balances. This gives us the idea that maybe an account *is its own class*! But as we stated, it is not completely independent of a class; accounts only exist because users open them.

*For this assignment, don't worry about putting any user information in the BankAccount class. We'll take care of that in the next lesson!*

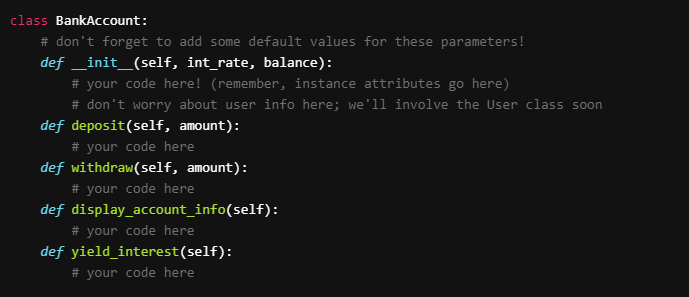
Let's first just get some more practice writing classes by writing a new *BankAccount* class.

The BankAccount class should have a balance. When a new BankAccount instance is created, if an amount is given, the balance of the account should initially be set to that amount; otherwise, the balance should start at $0. The account should also have an interest rate in decimal format. For example, a 1% interest rate would be saved as 0.01. The interest rate should be provided upon instantiation. (Hint: when using default values in parameters, the order of parameters matters!)

The class should also have the following methods:

* **deposit(self, amount)** - increases the account balance by the given amount
* **withdraw(self, amount)** - decreases the account balance by the given amount if there are sufficient funds; if there is not enough money, print a message "Insufficient funds: Charging a $5 fee" and deduct $5
* **display\_account\_info(self)** - print to the console: eg. "Balance: $100"
* **yield\_interest(self)** - increases the account balance by the current balance \* the interest rate (as long as the balance is positive)

This means we need a class that looks something like this:



 Create a BankAccount class with the attributes interest rate and balance

 Add a deposit method to the BankAccount class

 Add a withdraw method to the BankAccount class

 Add a display\_account\_info method to the BankAccount class

 Add a yield\_interest method to the BankAccount class

 Create 2 accounts

 To the first account, make 3 deposits and 1 withdrawal, then yield interest and display the account's info all in one line of code (i.e. chaining)

 To the second account, make 2 deposits and 4 withdrawals, then yield interest and display the account's info all in one line of code (i.e. chaining)

 NINJA BONUS: use a classmethod to print all instances of a Bank Account's info