Estrutura de dados
Orientação a Objetos

Profª. Drª. Alana Morais

Exemplo Funcionário

Imagine que agora no seu projeto você precisa adicionar um nome e um id ao Setor.
Teste sua aplicação.

Planejamento OO

1. Identifique uma classe que deve existir no seu problema
    a. Pense nos atributos
    b. Implemente o construtor
    c. Implemente os métodos da classe
    d. Implemente os métodos gets e sets / properties
    e. Implemente o def __str__(self)
2. Verifique se há outras classes
    a. Repita o processo
3. Implemente o arquivo teste.py

Exemplo Conta

- Cliente
  - nome, cpf, dataNascimento, endereco

- Conta:
  - cliente, agencia, numero, limite e saldo
  - sacar, depositar e extrato

Vocês terminaram os get, os sets e as propriedades?

Exemplo Conta

- Cliente
  - nome, cpf, dataNascimento, endereco

- Conta:
  - cliente, agencia, numero, limite e saldo
  - sacar, depositar e extrato

Como se chama a técnica de OO de inserir um objeto em outra classe como atributo?

Exemplo Conta

- Cliente
  - nome, cpf, dataNascimento, endereco

- Conta:
  - cliente, agencia, numero, limite e saldo
  - sacar, depositar e extrato

Como se chama a técnica de OO de inserir um objeto em outra classe como atributo?

COMPOSIÇÃO

Exemplo Conta

```
class Conta:

    def __init__(self, cliente, agencia, conta, limite):
        self.__cliente = cliente  # Cliente
        self.__agencia = agencia
        self.__conta = conta
        self.__limite = limite
        self.__saldo = 0

    def extrato(self):
        print("Saldo de {} do titular {}".format(self.__saldo,
self.__titular))

    def deposita(self, valor):
        self.__saldo += valor

    def saca(self, valor):
        self.__saldo -= valor

    def transfere(self, valor, destino):
        self.saca(valor)
        destino.deposita(valor)
```

```
#Getters e Setters
def get_cliente(self):
    return self.__cliente

def set_cliente(self, novoTitular):
    self.__cliente = novoTitular

def get_agencia(self):
    return self.__agencia

def get_conta(self):
    return self.__conta

def get_limite(self):
    return self.__limite

def set_limite(self, novoLimite):
    self.__limite = novoLimite

def get_saldo(self):
    return self.__saldo

def set_saldo(self, novoSaldo):
    self.__saldo = novoSaldo
```

Exemplo Conta

```
#Properties - Modo 1
@property
def cliente(self):
   return self.__cliente

@cliente.setter
def cliente(self, novoCliente):
   self.__cliente = novoCliente

@property
def agencia(self):
   return self.__agencia

@property
def conta(self):
   return self.__conta

@property
def limite(self):
   return self.__limite

@limite.setter
def limite(self, limite):
   self.__limite = limite

@property
def saldo(self):
   return self.__saldo

@saldo.setter
def saldo(self, novoSaldo):
   self.__saldo = novoSaldo
```

Exemplo Conta

Será que não tem outra forma de fazer isso mais rápido?

Properties

- As propriedades que conhecemos

```
class Cliente:
    #construtor, get e set

    @property
    def nome(self):
        return self.__nome

    @nome.setter
    def nome(self, nome):
        self.__nome = nome
```

- O equivalente:

```
class Cliente:
    #construtor, get e set

    nome = property (get_nome, set_nome)
```

Properties

- As propriedades que conhecemos

```
class Cliente:
    #construtor, get e set

    @property
    def nome(self):
        return self.__nome

    @nome.setter
    def nome(self, nome):
        self.__nome = nome
```

- O equivalente:

```
class Cliente:
    #construtor, get e set

    nome = property (get_nome, set_nome)
```

Devem ter sido implementados. Não esqueça!

O que faremos agora?

Exercícios!!!

Dúvidas?

alanamm.prof@gmail.com