09

Mãos na massa

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Comece a armazenar as informações de uma playlist. Para começar, coloque os filmes e séries numa lista Python:

```
class Programa:
    def __init__(self, nome, ano):
        self._nome = nome.title()
        self.ano = ano
        self._likes = 0

    @property
    def likes(self):
        return self._likes

    def dar_likes(self):
        self._likes += 1

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

    @nome.setter
    def nome(self, nome):
        self._nome = nome

class Filme(Programa):
    def __init__(self, nome, ano, duracao):
        super().__init__(nome, ano)
        self.duracao = duracao

class Serie(Programa):
    def __init__(self, nome, ano, temporadas):
        super().__init__(nome, ano)
        self.temporadas = temporadas

vingadores = Filme('vingadores - guerra infinita', 2018, 160)
atlanta = Serie('atlanta', 2018, 2)
vingadores.dar_likes()
vingadores.dar_likes()
vingadores.dar_likes()

atlanta.dar_likes()
atlanta.dar_likes()

listinha = [atlanta, vingadores]

for programa in listinha:
    print(f'Nome: {programa.nome} - Likes: {programa.likes}')
```

As informações de filmes e séries são mostradas em um único `for in` , já que são tipos derivados de `Programa` , você consegue imprimir os seu valores sem precisar diferenciar cada tipo.

2) Há um problema pra resolver, o `Filme` e a `Serie` são diferentes, um tem duração e a outra tem temporadas. Implemente uma forma de verificar antes de imprimir, para mostrar informações mais completas sobre.

Na hora de imprimir, faça o seguinte:

```
for programa in listinha:
    if hasattr(programa, 'duracao'):
        detalhe = f'{programa.duracao} min'
    else:
        detalhe = f'{programa.temporadas} temporadas'

    print(f'Nome: {programa.nome} - {detalhe} - Likes: {programa.likes}')
```

Com este código, você verifica se tem o atributo `duracao` (atributo que um filme tem), caso contrário, será uma série.

3) O problema foi resolvido, mas tem uma outra forma de fazer o mesmo sem precisar desse `if` *intrujão*. Crie um método que imprime o programa como string.

Na classe `Programa` :

```
def imprime(self):
    print(f'Nome: {self.nome}  Likes: {self.likes}')
```

Na classe `Filme` :

```
def imprime(self):
    print(f'Nome: {self.nome} - {self.duracao} min - Likes: {self.likes}')
```

E na classe `Serie` :

```
def imprime(self):
    print(f'Nome: {self.nome} - {self.temporadas} temporadas - Likes: {self.likes}')
```

4) Agora, remova o `if` da parte da impressão, deixando assim:

```
for programas in listinha:
    programa.imprime()
```

Ficou bem mais enxuto né? Desta forma, você está usando o polimorfismo para executar o método `imprime` de qualquer programa, seja filme ou série, tendo um resultado diferente para cada um.

5) Melhorou o código, mas ainda não é o jeito mais *pythonico* de fazer isto. Para apresentar objetos como string de forma *pythonica*, é preciso implementar o método `__str__` . Substitua o método `imprime` pela implementação abaixo nas 3 classes.

Na classe `Programa` :

```
def __str__(self):
    return f'Nome: {self.nome} Likes: {self.likes}'
```

Na classe `Filme`:

```
def __str__(self):
    return f'Nome: {self.nome} - {self.duracao} min - Likes: {self.likes}'
```

Na classe `Serie`:

```
def __str__(self):
    return f'Nome: {self.nome} - {self.temporadas} temporadas - Likes: {self.likes}'
```

6) Note que agora é preciso retornar uma string ao invés de imprimir diretamente. Feito isso, modifique também a impressão, para deixar somente o objeto sendo impresso:

```
for programa in listinha:
    print(programa)
```