

# XDES02

## Programação Orientada a Objetos

Aula 11: Prática - Fração e fração mista

# Fração

- Vamos trabalhar com um novo tipo de dado
  - ▣ Fração
    - Representado por dois números inteiros, um numerador e um denominador
  - ▣ Queremos ser capazes de criar objetos tipo fração e realizar operações
    - Impressão
    - Aritmética
    - etc.

# Fração

- Vamos começar definindo a classe fração

```
class Fracao:

    def __init__(self, num, den):
        self.__num = num
        self.__den = den

    @property
    def num(self):
        return self.__num

    @property
    def den(self):
        return self.__den
```

# Fração

- Queremos agora criar uma instância de fração e depois mostrá-la (imprimi-la)
- Em Python, quando a função print é usada para imprimir um objeto, o método `__str__` da classe desse objeto é invocado
- Precisamos, então, criar esse método como uma string que representa o objeto
  - ▣ Esse procedimento recebe o nome de sobrecarga de métodos (method overloading)

# Fração

```
class Fracao:
    def __init__(self, num, den):
        self.__num = num
        self.__den = den

    def __str__(self):
        return str(self.__num) + "/" + str(self.__den)

    @property
    def num(self):
        return self.__num

    @property
    def den(self):
        return self.__den

frac1 = Fracao(3,4)
print(frac1)
```

# Fração

- Uma operação importante quando lidamos com frações é a simplificação
- Não é adequado termos a fração  $2/8$ , já que ambos, numerador e denominador, podem ser divididos por 2
  - ▣ Então  $2/8 = 1/4$
- Para fazermos simplificação, temos que encontrar o MDC entre o numerador e o denominador da função

# Fração

```
# Algoritmo Euclidiano
```

```
def mdc(m, n):  
    while m%n != 0:  
        oldm = m  
        oldn = n  
        m = oldn  
        n = oldm%oldn  
    return n
```

```
def simplifica(self):  
    divComum = mdc(self.__num, self.__den)  
    self.__num = self.__num // divComum  
    self.__den = self.__den // divComum
```

```
frac1 = Fracao(2,8)  
print(frac1)  
frac1.simplifica()  
print(frac1)
```

# Fração

```
def simplifica(self):  
    divComum = mdc(self.__num, self.__den)  
    self.__num = self.__num // divComum  
    self.__den = self.__den // divComum
```

- // Operador floor division
- Como funciona?
  - ▣  $10 / 4 = 2.5$
  - ▣  $10 // 4 = 2$



# Ingualdade

## Entre objetos

- Frase 1: “José e Paulo tem o mesmo carro”
  - ▣ Tanto José como Paulo tem o carro da mesma marca e do mesmo modelo, mas é provável que existam dois objetos carro distintos (cada pessoa possui um)
- Frase 2: “Antônio e João tem a mesma mãe”
  - ▣ Aí existe só um objeto que é compartilhado por Antônio e João
- Para comparar igualdade entre objetos existem dois operadores:
  - ▣ ==
  - ▣ is

# Ingualdade

Entre objetos

```
def mesmaFracao(f1, f2):  
    return (f1.num == f2.num) and (f1.den == f2.den)
```

```
frac1 = Fracao (3,4)  
frac2 = Fracao(3,4)  
print(mesmaFracao(frac1 ,frac2))  
print(frac1 is frac2)  
frac2 = frac1  
print(frac1 is frac2)
```

# Aritmética de Frações

- Só é possível somar frações se elas tiverem o mesmo denominador. Assim, o método mais simples para somar duas frações quaisquer é multiplicar seus denominadores
- O que é feito para o denominador deve também ser feito para o numerador, então:

$$□ \frac{a}{b} + \frac{c}{d} = \frac{(ad + cb)}{bd}$$

# Aritmética de Frações

- Ao invés de criar um método com um nome qualquer, por exemplo, `adicionaFracao`, vamos invocar o método `add` fazendo:
  - ▣ `def __add__`
    - Sobrecarga do operador `+`
  - ▣ Dessa forma, podemos usar o operador `+` para somar frações:
    - `frac1 = frac2 + frac3`
  - ▣ Podemos fazer o mesmo com os demais operadores aritméticos
    - `- * /`

# Fração

```
def __add__(self, outraFrac):  
    novoNum = self.__num * outraFrac.den + self.__den * outraFrac.num  
    novoDen = self.__den * outraFrac.den  
    divComum = mdc(novoNum, novoDen)  
    return Fracao(novoNum//divComum, novoDen//divComum)
```

□  $a/b + c/d = (ad + cb)/bd$

▣ return cria uma função já simplificada

# Aritmética de Frações

- Faça a soma das seguintes frações e veja o resultado

- ▣  $\frac{3}{4} + \frac{5}{6} = \frac{19}{12}$

- ▣  $\frac{19}{12}$  na verdade é  $1 \frac{7}{12}$

# Exercício

- Crie a classe `fracaoMista`, composta de:
  - ▣ parte inteira
  - ▣ parte fracionária
- Ajuste a implementação de forma que o resultado da soma de duas frações resulte no objeto correto
  - ▣ `Fracao` se  $\text{num}/\text{den} < 1$
  - ▣ `Fracao mista`, caso contrário

# Exercício

- Sua implementação deve produzir os seguintes resultados:

- $7/6 + 13/7 = 3 \frac{1}{42}$

- $1/3 + 2/3 = 1$

- $3 \frac{1}{2} + 4 \frac{2}{3} = 8 \frac{1}{6}$