Mais exercícios de Sobrecarga de Operadores e Objetos Funcionais

Juliana Pirolla - Revisão

Exercício 1

Imagine que você está trabalhando em um projeto de um sistema de controle de acesso para um evento. Neste sistema, é necessário verificar se uma pessoa é elegível para entrar no evento com base em sua idade. Para isso, você precisa implementar a classe Person com o método especial **bool** que retorna True se a pessoa tiver idade entre 18 e 65 anos, e False caso contrário.

Exercício 2

Suponha que você esteja trabalhando em um projeto que envolve o uso de vetores bidimensionais. Para facilitar a manipulação desses vetores, você implementou a classe Vector2D com os métodos necessários.

Crie uma instância da classe Vector2D chamada vector1 com os valores x = 3 e y = 4. Imprima os valores de x e y do objeto vector1. Chame o método especial **abs** no objeto vector1 e armazene o resultado em uma variável chamada magnitude. Imprima o valor da magnitude calculada.

Exercício 3

Você precisa desenvolver uma classe chamada ShiftedMul em Python, que representa uma multiplicação com deslocamento. Essa classe será responsável por executar cálculos de multiplicação com um deslocamento adicional.

A classe ShiftedMul possui os seguintes métodos:

`__init__(self, shift)`: Este é o método construtor da classe. Ele recebe um parâmetro shift, c

`__call__(self, a, b)`: Este método permite que a instância do objeto seja chamada como uma fu

A sua tarefa é implementar a classe ShiftedMul conforme a descrição acima.

Exercício 4

ocê foi desafiado a implementar uma função em Python chamada shifted_mul(shift), que realiza cálculos de multiplicação com um deslocamento adicional. A função shifted_mul() recebe um parâmetro shift e retorna uma função personalizada que executa a multiplicação com o deslocamento fornecido.

Sua tarefa é implementar a função shifted_mul() e utilizá-la para criar duas funções personalizadas: mul_com_deslocamento2 e mul_com_deslocamento5. A primeira função deve multiplicar dois números e adicionar um deslocamento de 2 ao resultado, enquanto a segunda função deve multiplicar dois números e adicionar um deslocamento de 5 ao resultado.

Depois de criar as funções personalizadas, você deve utilizar cada uma delas para calcular o resultado das seguintes operações:

```
mul_com_deslocamento2(3, 4)
mul_com_deslocamento5(2, 6)
```

Exercício 5

O código fornecido apresenta uma função principal chamada talk(f), que recebe uma função como parâmetro e a chama com argumentos fixos (1, 10).

Além disso, o código inclui duas funções definidas diretamente, simple(a, b) e lam, que são passadas como argumentos para a função talk(f). Há também uma classe chamada Func, com um método especial **call**() que é usado quando uma instância dessa classe é chamada como uma função.

```python
def talk(f):
    print(f(1, 10))

def simple(a, b):
    return 2 * a + b

lam = lambda a, b: 2 * b + a

class Func:
    def __init__(self, c):
        self._c = c
```

```
    def __call__(self, a, b):
        return self._c * a + b

talk(simple)
talk(lam)
talk(Func(2))
talk(Func(20))
```

Sua tarefa é analisar o código fornecido e responder a algumas perguntas:

1. Qual será a saída exibida na tela após a execução do código fornecido?
2. Qual é a relação entre a função simple(a, b) e a função lambda lam? Em que aspecto elas são semelhantes e em que aspecto diferem?
3. O que acontece quando uma instância da classe Func é passada como argumento para a função talk(f)? Explique o papel do método **call**() na classe Func.
4. Modifique o código para criar uma nova função ou método, juntamente com uma chamada correspondente à função talk(f). Observe como a função talk(f) pode ser usada para chamar diferentes implementações de funções ou métodos.