

RASCUNHOS DA SENHORA DEVA

Python com Matemática Financeira

Material exclusivo da
Senhora-de-programa
+ legal da internet



ÍNDICE

3

Quem é Sr^a. Deva?

2

Resumão

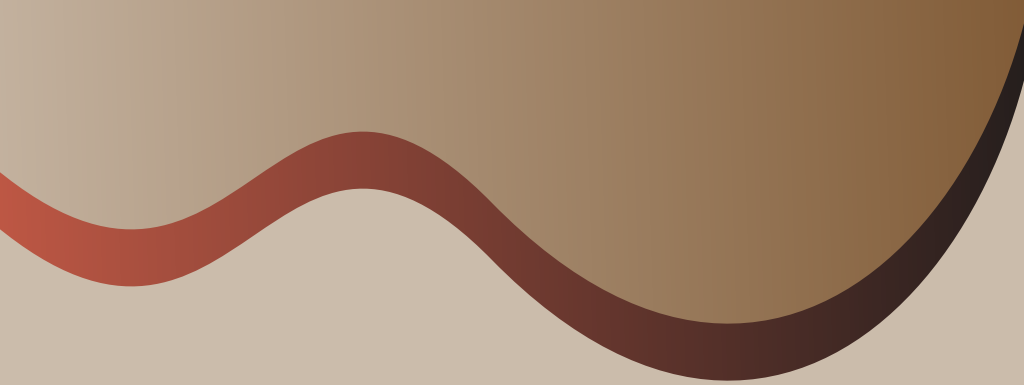
4

Ferramentas e
informações
importantes

5

Desenrola aí
Senhora Deva





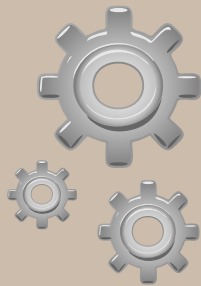
Senhora Deva, onde vive? Quem é?

Senhora Deva é uma iniciativa para distribuir conteúdo voltado para o universo da programação/tecnologia. Com uma forma descontraída e bem humorada, buscamos fomentar a participação de mais mulheres nesta área, bem como compartilhar nossas experiências e as dificuldades



Resumão da Senhora Deva

Olá, Devers! Se você quer descobrir como estou me desenvolvendo na programação, leia este e-Book até o final. Compartilharei aqui meu processo de estudo em Python e matemática financeira. Por enquanto, começo com técnicas bem básicas, a princípio cálculos de porcentagem.



Ferramentas e informações importantes

Para escrever os códigos foi utilizado o **Google Colab**, ferramenta online em ambiente de **notebooks Jupyter**. Uma das vantagens de utilizá-lo é não precisar de configurações da linguagem Python e, o melhor, dá para integrá-lo ao **Github**.



ANOTAÇÕES DEVA

Antes de colocar as ideias em prática, um dica marota! Faça da documentação a sua melhor amiga, isto vale para qualquer linguagem. Outro ponto importante, não iremos focar nas teorias matemáticas e programação.



Desenrola aí Senhora Deva

Para calcular a porcentagem de um valor podemos pensar o seguinte: $P\% = P / 100$. Por exemplo: 25% de 800, $25/100 * 800$. Bora transformar isto em Python?

```
▶ print("Calcular 25% de 800")  
p = 25  
value = 800  
res = float(p / 100) * value
```

```
↳ Calcular 25% de 800
```

```
[6] res
```

```
200.0
```

ANOTAÇÕES DEVA

Criar as variáveis em inglês(estudar inglês cerca de uma hora dia, ou dividir os estudos ao longo do dia).



Note que apenas a lógica matemática virou lógica de programação, legal né? Vou um pouco mais além! E se eu quisesse calcular várias porcentagens de uma vez? Ora, bem simples, basta criar listas para armazenar os valores e uma vazia para receber os resultados. Assim ó:

```
[7] percent_numbers = [10, 36, 25]
    val = [200, 940, 800]

    res = []

    for i in range(0, len(percent_numbers)):
        res.append(float((percent_numbers[i] / 100) * val[i]))
```

```
[8] res

[20.0, 338.4, 200.0]
```

Esta é a única e melhor forma de se fazer? Não, existem diversas maneiras, poderia utilizar alguma biblioteca, por exemplo, algo que vou fazer, mas apenas depois de entender os conceitos essenciais e conseguir deixar o código ainda mais enxuto. O importante é ter em mente que sempre dá para melhorar.

Pesquisar quais formas distintas posso fazer um `for{}`.



E se quisermos calcular porcentagem com aumentos e descontos? Podemos utilizar fator multiplicativo. Fórmula $(1 \pm i)$ o sinal de "mais" representa a porcentagem em caso de aumento e, o de "menos", para desconto. Bora desmembrar uma fórmula da ideia, fica algo assim, $\text{valor} * (1 \pm (i / 100))$. Vamos imaginar um problema, uma bicicleta custava R\$ 36.000, 00 e valorizou 8,5%. Qual o seu preço atual? Bicicleta cara essa aí, agora é implementar.

```
i = 8.5
val = 36000

var_atual = float(val * (1 + (i / 100)))

print('Total is: %.2f' % var_atual)

Total is: 39060.00
```

Mais um problema para praticar: um pokémon custava R \$5.630,00. Uma semana depois, o preço teve uma redução de 32%. Qual o preço atual desse pokémon ?

Pensando mais um pouco: este problema, eu vou resolver utilizando uma função, e também criar um módulo para converter a taxa. Isso, boa ideia! Vou chamar o arquivo de `percent_conversion.py`, em seguida criar uma função com o mesmo nome.



```
1      # -*- coding: utf-8 -*-  
2  
3      def percent_conversion(i):  
4          return i / 100  
5      |
```

Pesquisando sobre utilizar módulos no Colab, resolvi utilizar a importação. Para isso, criei o módulo na minha máquina e depois importei utilizando o seguinte código:



```
from google.colab import files  
uploaded = files.upload()
```

Escolher arquivos percent_conversion.py

- percent_conversion.py(n/a) - 132 bytes, last modified: 07/02/2021 - 100% done

Saving percent_conversion.py to percent_conversion.py

Se observarmos, basta ir até o arquivo que temos em nossa máquina e a mágica acontece. Então bora para a solução do problema, saca só que louco.

ANOTAÇÕES DEVA

Vou precisar passar dois parâmetros para a minha função, o valor anterior do produto e a porcentagem.



```
[3] import percent_conversion as pc
```



```
def current_price_bear(current_old_bear, i):  
    return float(current_old_bear * (1 - pc.percent_conversion(i)))
```

```
res = current_price_bear(5630, 32)  
print('Price bear is: %.2f' %res)
```

```
Price bear is: 3828.40
```

Para usar o módulo que criamos, basta importar. Podemos dar um apelido pra ele, neste caso, pc, mas você pode chamá-lo como quiser.

ANOTAÇÕES DEVA

Para os próximos
códigos, observar se:
os nomes das
variáveis estão claras
e intuitivas,
implementar juros
simples e compostos.



Compartilhe com seus amigos!

Se você gostou deste conteúdo e acredita que alguém possa se beneficiar com ele, que tal compartilhar?

Ah! Para mais dicas da Senhora Deva, me siga no **Instagram**! Lá, eu publico conteúdo sempre!



Quero seguir
a Sr^a. Deva