

# Lista de Exercícios de Programação

## Exercício 0001

### Enunciado:

Imagine que você tem uma lista de números. O objetivo é calcular o  $n$ -ésimo número da sequência de Fibonacci para cada número dessa lista e imprimir a soma total desses números.

A sequência de Fibonacci é como uma escada em espiral, onde cada número é a soma dos dois anteriores. Ela começa com 0 e 1, e continua infinitamente: 0, 1, 1, 2, 3, 5, 8, 13, 21...

Para isso, você precisará criar duas funções recursivas:

1. **A função `fibonacci`:** Essa função receberá um número inteiro  $n$  como entrada e, usando a recursão, calculará o  $n$ -ésimo número correspondente na sequência de Fibonacci. Lembre-se que o 0-ésimo termo é 0 e o 1-ésimo termo é 1.

2. **A função `soma_recursiva_lista`:** Essa função receberá uma lista de números inteiros como argumento e, novamente usando a recursão, somará todos os elementos dessa lista.

No final, seu programa deve ler uma sequência de números inteiros separados por vírgulas como entrada. Para cada número na sequência, calcule o seu valor na sequência de Fibonacci usando a função `fibonacci`. Em seguida, some todos esses valores de Fibonacci calculados utilizando a função `soma_recursiva_lista` e imprima na tela a soma total.

Lembre-se: a recursão é uma técnica onde uma função chama a si mesma para resolver problemas menores até chegar a um ponto de parada, chamado de "condição base".

### Testes

#### Teste 1

Entrada:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

Saída:

1596

#### Teste 2

Entrada:

10, 11, 9, 15, 17, 23, 13, 27, 12, 11, 22

Saída:

245637

# Exercício 0052

## Enunciado:

Imagine que você tem uma lista de números. O objetivo é calcular o N-ésimo número da sequência de Fibonacci para cada número dessa lista e imprimir a soma total desses números.

A sequência de Fibonacci é uma série onde cada número é a soma dos dois anteriores, começando com 0 e 1: 0, 1, 1, 2, 3, 5, 8, 13, 21...

Para isso, você precisará criar duas funções recursivas:

1. `fibonacci(n)`: Esta função receberá um número inteiro `n` e, usando a recursão, calculará o N-ésimo número na sequência de Fibonacci. As condições base para a recursão devem ser:

- \* Se `n` for 0, retorna 0.

- \* Se `n` for 1, retorna 1.

- \* Para `n` maior que 1, retorna a soma de `fibonacci(n-1)` e `fibonacci(n-2)`.

2. `soma_lista_recursiva(lista)`: Esta função receberá uma lista de números (que serão os resultados da função `fibonacci`) e, novamente usando a recursão, somará todos os elementos dessa lista. A condição base deve ser quando a lista está vazia.

No programa principal, você deve:

- \* Ler uma linha de números inteiros separados por espaço (que representarão os `n` para a função `fibonacci`).

- \* Processar cada número de entrada usando a função `fibonacci` para obter o valor correspondente na sequência.

- \* Coletar todos os resultados da `fibonacci` em uma nova lista.

- \* Finalmente, usar a função `soma_lista_recursiva` para obter a soma total de todos os valores de Fibonacci calculados.

- \* Imprimir na tela a soma total.

## Testes

### Teste 1

Entrada:

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

Saída:

```
1596
```

### Teste 2

Entrada:

```
10 11 9 15 17 23 13 27 12 11 22
```

Saída:

```
245637  
` ``
```

# Exercício 0061

## Enunciado:

Você deve criar uma função Python, chamada `calcular_potencia_recursiva`, que calcula o resultado de uma base elevada a um expoente inteiro e não negativo, utilizando **exclusivamente o conceito de recursividade**. A função não deve utilizar operadores de potência (`**`) ou funções built-in como `pow()` ou `math.pow()`. Apenas operações básicas de multiplicação são permitidas no passo recursivo.

A função receberá dois argumentos:

\* `base` (do tipo `float`): O número a ser elevado.

\* `expoente` (do tipo `int`): O número inteiro e não negativo que representa a potência.

Considere as seguintes diretrizes para sua implementação recursiva:

1. **Condição Base:** Qualquer número elevado a 0 é 1.
2. **Passo Recursivo:** Para um expoente `n` maior que 0,  $base^n$  é igual a  $base * base^{(n-1)}$ .

No programa principal, você deve ler a base e o expoente fornecidos pelo usuário em linhas separadas, chamar a função `calcular_potencia_recursiva` e imprimir o resultado formatado como 'Base^Expoente = Resultado'.

## Testes

### Teste 1

Entrada:

3  
2

Saída:

$3^2 = 9$

### Teste 2

Entrada:

2  
5

Saída:

$2^5 = 32$

### Teste 3

Entrada:

1  
10

Saída:

$1^{10} = 1$

### Teste 4

Entrada:

7  
0

Saída:

$7^0 = 1$

### Teste 5

Entrada:

2.5

2

Saída:

$2.5^2 = 6.25$

# Exercício 0080

## Enunciado:

Crie um programa Python que realize a conversão de temperaturas entre as escalas Celsius e Fahrenheit. O programa deve ser modularizado utilizando funções para as operações de conversão.

Sua solução deve incluir as seguintes funções:

1. `celsius_para_fahrenheit(celsius: float) -> float`: Esta função deve receber um valor de temperatura em graus Celsius como parâmetro e retornar o valor equivalente em graus Fahrenheit.

\* **Fórmula:**  $Fahrenheit = Celsius * 9/5 + 32$

2. `fahrenheit_para_celsius(fahrenheit: float) -> float`: Esta função deve receber um valor de temperatura em graus Fahrenheit como parâmetro e retornar o valor equivalente em graus Celsius.

\* **Fórmula:**  $Celsius = (Fahrenheit - 32) * 5/9$

No programa principal:

\* Solicite ao usuário que digite um valor de temperatura (numérico).

\* Em seguida, solicite ao usuário que digite a unidade original da temperatura ('C' para Celsius ou 'F' para Fahrenheit). A entrada da unidade deve ser tratada sem distinção entre maiúsculas e minúsculas (ex: 'c' ou 'C' devem ser aceitos para Celsius).

\* Com base na unidade informada, chame a função de conversão apropriada.

\* Imprima o resultado da conversão formatado para duas casas decimais, indicando as unidades de origem e destino.

## Testes

### Teste 1

Entrada:

25.0  
C

Saída:

A temperatura de 25.00°C é 77.00°F.

### Teste 2

Entrada:

68.0  
F

Saída:

A temperatura de 68.00°F é 20.00°C.

### Teste 3

Entrada:

0.0  
C

Saída:

A temperatura de 0.00°C é 32.00°F.

### Teste 4

Entrada:

212.0

f

Saída:

A temperatura de 212.00°F é 100.00°C.

### Teste 5

Entrada:

-40.0

C

Saída:

A temperatura de -40.00°C é -40.00°F.

# Exercício 0098

## Enunciado:

Você foi contratado para desenvolver um sistema de cálculo de preços para uma loja. Sua tarefa é criar uma função em Python que determine o preço final de um produto, aplicando um desconto base e, opcionalmente, um desconto adicional para clientes membros.

Crie uma função chamada `calcular_preco_final` que aceite os seguintes parâmetros:

- `preco_original`: um valor float que representa o preço inicial do produto.
- `percentual_desconto`: um valor float que representa o percentual de desconto a ser aplicado (ex: 10 para 10%).
- `e_membro`: um valor booleano (`True` ou `False`) que indica se o cliente é um membro da loja.

A função deve realizar os seguintes cálculos:

1. Aplicar o `percentual_desconto` sobre o `preco_original`.
2. Se `e_membro` for `True`, aplicar um **desconto adicional** de 5% sobre o preço já com o desconto base.
3. A função deve retornar o `preco_final` calculado.

Certifique-se de usar `type hints` para os parâmetros e o retorno da função, e inclua uma `docstring` clara que descreva o propósito da função, seus argumentos e o valor de retorno.

No programa principal, leia o preço original, o percentual de desconto e a informação de membro (como "Sim" ou "Nao") do usuário. Em seguida, chame a função `calcular_preco_final` com esses dados e imprima o preço final formatado com duas casas decimais.

### Exemplo de cálculo:

- Preço original: R\$ 100.00
- Desconto base: 10%
- Cliente membro: Sim
- Preço com desconto base:  $R\$ 100.00 * (1 - 0.10) = R\$ 90.00$
- Preço final (com 5% adicional):  $R\$ 90.00 * (1 - 0.05) = R\$ 85.50$

## Testes

### Teste 1

Entrada:

```
100.00
10.0
Nao
```

Saída:

```
O preço final é: R$ 90.00
```

### Teste 2

Entrada:

```
100.00
10.0
Sim
```

Saída:

```
O preço final é: R$ 85.50
```

### Teste 3

Entrada:

250.00

20.0

Nao

Saída:

O preço final é: R\$ 200.00

#### Teste 4

Entrada:

250.00

20.0

Sim

Saída:

O preço final é: R\$ 190.00

#### Teste 5

Entrada:

50.00

0.0

Sim

Saída:

O preço final é: R\$ 47.50

\\



# Exercício 0110

## Enunciado:

Desenvolva um programa Python para auxiliar na análise do desempenho escolar de um aluno. O programa deve ser capaz de receber uma série de notas e, a partir delas, calcular a média e identificar a maior nota obtida.

Para isso, siga as instruções:

1. **Leitura de Entrada:** O programa deve ler uma única linha de entrada contendo diversas notas (números de ponto flutuante), separadas por espaços.

2. **Conversão para Lista:** Converta essas notas em uma lista de números de ponto flutuante (`float`).

3. **Função `analisar_notas`:**

- \* Crie uma função chamada `analisar_notas` que receba como parâmetro uma lista de notas (`list[float]`).

- \* Esta função deve calcular a média aritmética de todas as notas na lista.

- \* Ela também deve encontrar a maior nota presente na lista.

- \* A função deve retornar esses dois valores como uma tupla: `(media, maior_nota)`.

- \* **Tratamento de Exceção:** Se a lista de notas estiver vazia, a função deve retornar `(0.0, 0.0)` para evitar erros de divisão por zero ou ao tentar encontrar o máximo em uma lista vazia.

4. **Programa Principal:**

- \* No programa principal, chame a função `analisar_notas` com a lista de notas lida da entrada.

- \* Imprima os resultados formatados com duas casas decimais, no seguinte formato: "Média das notas: X.XX, Maior nota: Y.YY".

## Exemplo:

Se a entrada for "7.5 8.0 6.0", a média é 7.166... e a maior nota é 8.0. A saída esperada seria "Média das notas: 7.17, Maior nota: 8.00".

## Testes

### Teste 1

Entrada:

```
7.5 8.0 6.0
```

Saída:

```
Média das notas: 7.17, Maior nota: 8.00
```

### Teste 2

Entrada:

```
5.0 5.0 5.0 5.0
```

Saída:

```
Média das notas: 5.00, Maior nota: 5.00
```

### Teste 3

Entrada:

```
10.0 9.5 9.8 10.0
```

Saída:

```
Média das notas: 9.82, Maior nota: 10.00
```

### Teste 4

Entrada:

3.0

Saída:

Média das notas: 3.00, Maior nota: 3.00

### **Teste 5**

Entrada:

Saída:

Média das notas: 0.00, Maior nota: 0.00