

python™

# Laboratório de Programação Competitiva

Profa. Silvia Brandão

2024.1

# Aula de hoje

- Elaboração de códigos
- Teoria dos Números x Teoria dos Números em Python
  - Algoritmo de Euclides,
  - Teorema fundamental da aritmética: MDC, MMC, Primos, Fatorial
  - Teorema Chinês do Resto,
  - Crivo de Eratóstenes.

## **MOMENTO N2:**

- Crie seu portfólio (notebook), no Google Colab, para o momento de avaliação N2. Qualquer exercício proposto, neste momento, deverá ser implementado no portfólio para posterior correção no valor de 10 pontos. Não se esqueça de incluir NOME e RA.
- Lista de exercícios no Beecrowd, valor de 5 pontos.
- Avaliação prática, valor de 10 pontos.

# NaN (*Not a Number*) e Inf (*Infinity*) em Python

## ➤ NaN

- É um valor especial usado para **representar resultados inválidos ou indefinidos** em operações matemáticas. É útil quando trabalhamos com números em Python.
- Exemplo: **divisões por zero ou operações com valores inválidos.**

## ➤ Inf

- O infinito é definido como um **número indefinido** que pode ser um **valor positivo ou negativo**.
- Todas as operações aritméticas realizadas sobre um valor infinito sempre levam a um número infinito, seja soma, subtração, multiplicação ou qualquer outra operação.



```
nro_nan = math.nan
print('Número nan: ', nro_nan)

positivo_inf = float('inf')
print('Positive Infinity: ', positivo_inf)

negativo_inf = float('-inf')
print('Negative Infinity: ', negativo_inf)
```

```
Número nan:  nan
Positive Infinity:  inf
Negative Infinity:  -inf
```

# NaN (*Not a Number*)

**Exemplo 1:** Divisão por zero

```
resultado = 0 / 0  
print(resultado)  # Saída: NaN
```

**Exemplo 2:** Operações inválidas

```
numero_negativo = -1  
raiz_quadrada =  
math.sqrt(numero_negativo)  
print(raiz_quadrada)  # Saída: NaN
```

**Exemplo 3:** Conversões inválidas

```
texto_invalido = "Olá, mundo!"  
numero_convertido = float(texto_invalido)  
print(numero_convertido)  # Saída: NaN
```



```
import math  
  
x = math.nan  
print(f"x contains {x}")  
  
# checks if variable is equal to NaN  
if(math.isnan(x)):  
    print("x == nan")  
else:  
    print("x != nan")
```

```
x contains nan  
x == nan
```

# Inf (*Infinity*)



```
def check(x):  
  
    if(math.isinf(x) and x > 0):  
        print("x is Positive inf")  
    elif(math.isinf(x) and x < 0):  
        print("x is Negative inf")  
    else:  
        print("x is not inf")
```

```
number = math.inf  
check(number)
```

```
number = -math.inf  
check(number)
```

```
x is Positive inf  
x is Negative inf
```

verifica se há infinito positivo e negativo



```
import numpy as np
```

```
print(np.isneginf([np.inf, 0, -np.inf]))  
print(np.isposinf([np.inf, 0, -np.inf]))
```

```
[False False  True]  
[ True False False]
```


Se lembre de instalar a biblioteca numpy:  
!pip install numpy





## 1.Exercício 1161, Beecrowd

# Soma de Fatoriais

Adaptado por Neilor Tonin, URI  Brasil

**Timelimit: 1**

Leia dois valores inteiros M e N indefinidamente. A cada leitura, calcule e escreva a soma dos fatoriais de cada um dos valores lidos. Utilize uma variável apropriada, pois cálculo pode resultar em um valor com mais de 15 dígitos.

## Entrada

O arquivo de entrada contém vários casos de teste. Cada caso contém dois números inteiros M ( $0 \leq M \leq 20$ ) e N ( $0 \leq N \leq 20$ ). O fim da entrada é determinado por eof.

## Saída

Para cada caso de teste de entrada, seu programa deve imprimir uma única linha, contendo um número que é a soma de ambos os fatoriais (de M e N).

### Exemplo de Entrada

4 4  
0 0  
0 2

### Exemplo de Saída

48  
2  
3

# Teoria dos Números em Python

## ➤ Algoritmo de Euclides

**Exemplo 2.2.1.** Calcule  $\text{mdc}(306, 657)$ .

*Realizando as divisões sucessivas, temos*

	2	6	1	4
657	306	45	36	9
45	36	9	0	

*ou*

$$\begin{aligned} 657 &= 2(306) + 45 \\ 306 &= 6(45) + 36 \\ 45 &= 1(36) + 9 \\ 36 &= 4(9) + 0 \end{aligned}$$

*Assim, temos*

$$\begin{aligned} \text{mdc}(657, 306) &= \text{mdc}(306, 45) = \text{mdc}(45, 36) = \\ &= \text{mdc}(36, 9) = \text{mdc}(9, 0) = 9. \end{aligned}$$

## 2.EXERCÍCIO:

Implemente, em Python, o cálculo do maior divisor comum entre dois números conforme o Algoritmo de Euclides.



# Números primos

- São uma parte crucial da Matemática e da Computação e entender como obtê-los de maneira eficiente para certos limites é muito importante na resolução de problemas.
- Os números primos são números que possuem apenas dois divisores: 1 e ele mesmo. Apesar da definição simples, muitos mistérios rondam estes números: qual a distribuição deles, dado um determinado número, é primo ou não é; quantos primos existem de um intervalo a outro, etc.
- Até hoje, não existe um método rápido para determinar se um número qualquer é primo ou não e isso é incrivelmente importante para a criptografia atual, que é baseada em números gigantescos que são fatorados em números primos igualmente enormes.

# TEOREMA FUNDAMENTAL DA ARITMÉTICA (TFA)

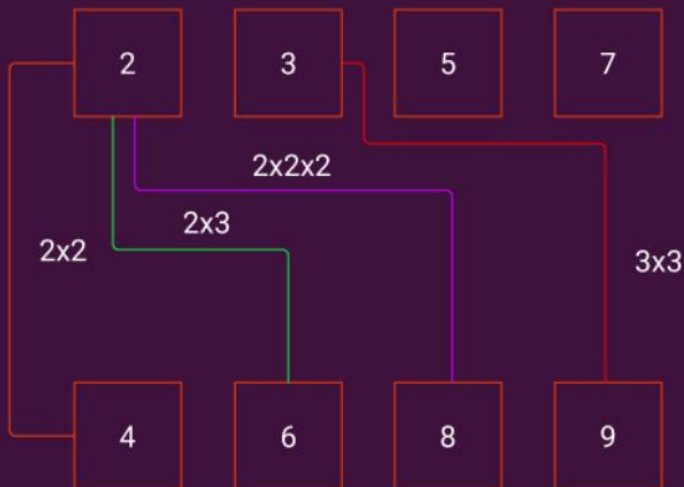
Seja  $n$  um número natural,  $n > 1$ .

Então existem números primos  $p_1 < p_2 < p_3 < \dots < p_r$  e, também, números naturais não nulos  $n_1, n_2, n_3, \dots, n_r$ , com  $r \geq 1$ , tais que:

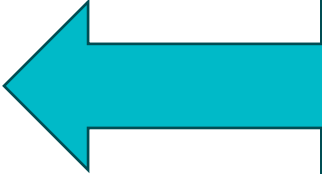
$$n = p_1^{n_1} p_2^{n_2} p_3^{n_3} \dots p_r^{n_r}$$

Além disso, essa decomposição é única.

Primos:



Compostos:


$$\begin{aligned} 4 &= 2^2 \\ 6 &= 2 \times 3 \\ 8 &= 2^3 \\ 9 &= 3^2 \end{aligned}$$

# Exercício

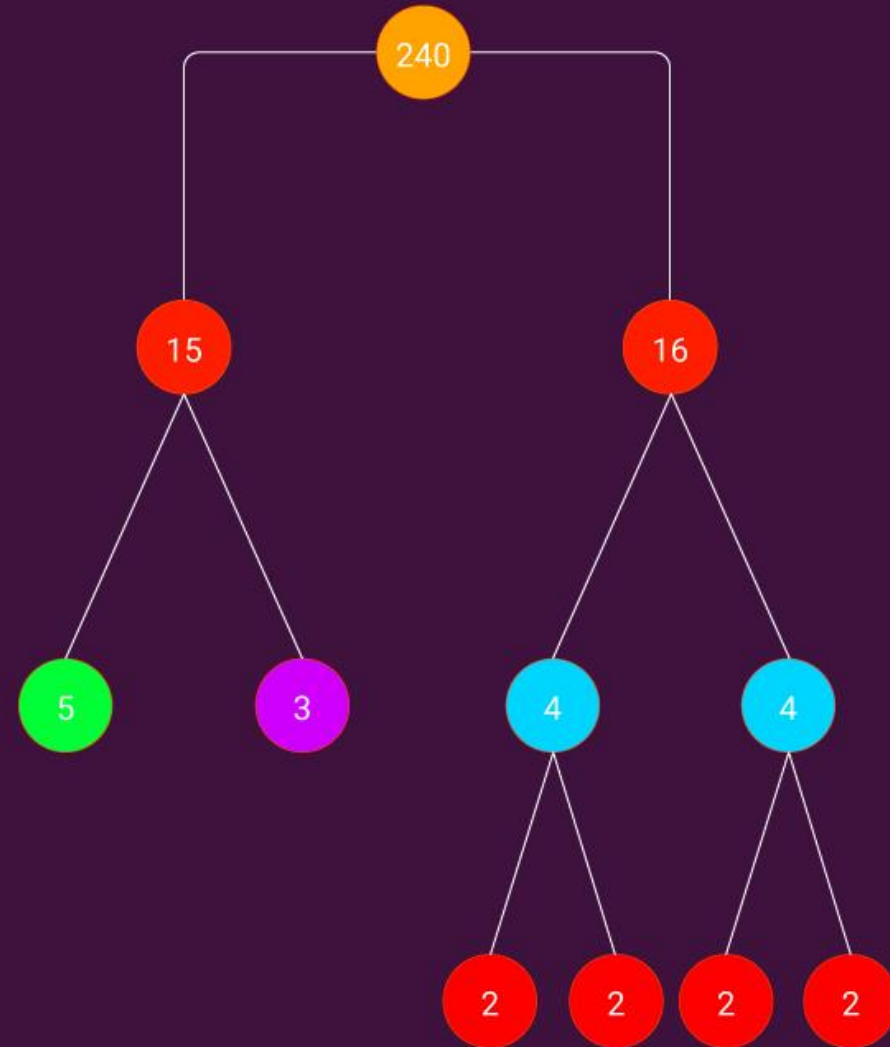
3. Escreva um programa em Python que leia um número e verifique se é ou não um número primo.

- Para fazer essa verificação, calcule o resto da divisão do número por 2 e depois por todos os números ímpares até o número lido.
- Se o resto de uma dessas divisões for igual a zero, o número não é primo.
- Observe que 0 e 1 não são primos e que 2 é o único número primo que é par.

# VISUALIZAÇÕES DA DECOMPOSIÇÃO GARANTIDA PELO TFA

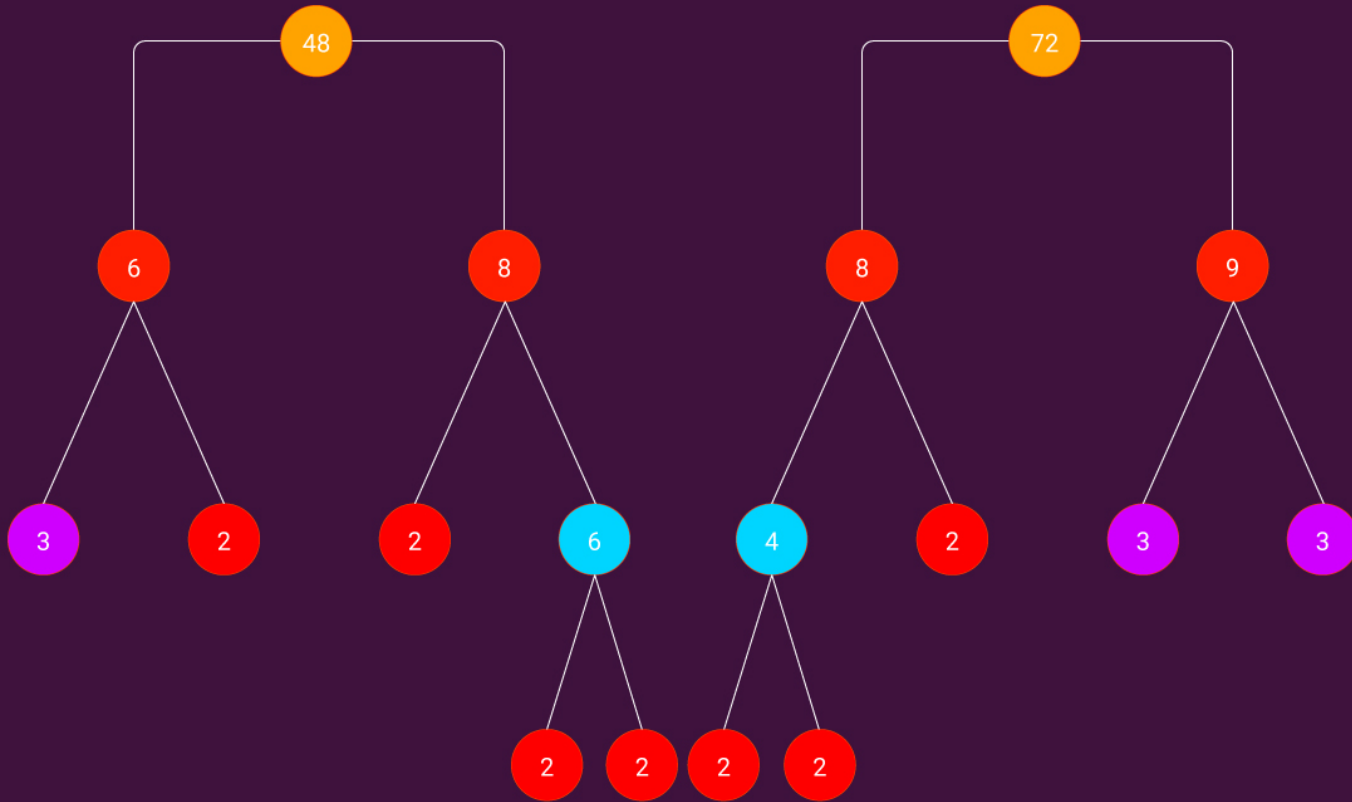


$$840 = 2^3 \times 3 \times 5 \times 7$$



$$240 = 2^4 \times 3 \times 5$$

## MMC E MDC USANDO O TEOREMA FUNDAMENTAL DA ARITMÉTICA



### Fatoração em primos de 48

$$2^4 \times 3^1$$

$$\text{MMC}(48, 72) = 2^4 \times 3^2 = 144$$

Produto da **MAIOR** potência de cada fator primo comum.

### Fatoração em primos de 72

$$2^3 \times 3^2$$

$$\text{MDC}(48, 72) = 2^3 \times 3^1 = 24$$

Produto da **MENOR** potência de cada fator primo comum.

# Exercício

4. Faça, em Python, o cálculo do maior divisor comum e do menor múltiplo comum usando o Teorema Fundamental da Aritmética, para isso implemente uma função para fatorar um número em seus fatores primos.



```
Fatores primos de num1: {2: 4, 3: 1}
```

```
Fatores primos de num2: {2: 3, 3: 2}
```

```
MDC de 48 e 72 : 24
```

```
MMC de 48 e 72 : 144
```

# Leituras:

- <https://impa.br/page-livros/teoria-dos-numeros-um-passeio-com-primos-e-outros-numeros-familiares-pelo-mundo-inteiro/>
- <https://docs.python.org/pt-br/3/library/math.html>
- <https://xtecna.gitbook.io/solucoes-da-beecrowd/base-teorica/matematica/numeros-primos>
- <https://portaldabmep.impa.br/uploads/msg/5ho9ahpkue4gg.pdf>
- Continue estudando:
  - <https://www.w3schools.com/python/default.asp>

# Próxima Aula

- Teorema Chinês do Resto,
- Crivo de Eratóstenes.



Não se esqueçam do Uniube+