Tutorial de Funções em python 🤓

🤖 *Professor Rooney Coelho*

```
print('Olá, o meu nome é Rooney')
```

```
Olá, o meu nome é Rooney
```

```
def func(nome):
  print(f'Seja bem-vindo(a) {nome}')
```

```
func('Wilson')
```

```
Seja bem-vindo(a) Wilson
```

```
func('Flávia')
```

```
Seja bem-vindo(a) Flávia
```

```
def func2(nome1, nome2):
  '''
  Código da função func2. Esta função recebe duas strings de nome como entrada
  e imprime na tela. A função não retorna nada.

  Programador Rooney Coelho, 18 de Março de 2024
  '''
  print(f'Sejam bem-vindo(a)s {nome1} e {nome2}') # Entradas da função
```

```
help(func2)
```

```
Help on function func2 in module __main__:

func2(nome1, nome2)
    Código da função func2. Esta função recebe duas strings de nome como entra
    e imprime na tela. A função não retorna nada.

    Programador Rooney Coelho, 18 de Março de 2024
```

```
func2('Wilson', 'Flávia')
```

```
Sejam bem-vindo(a)s Wilson e Flávia
```

Funções matemáticas

```
def polinomio(x):
  a = 2
  b = 3
  c = 4
  return a*x**2 + b*x + c


y = polinomio(0)
print(y)
```

```
    4
```

```
polinomio(1)
```

```
    9
```

```
# Como receber algo de input do usuário
x = input('Digite x: ')
print(f'Você digitou {x}')
```

```
    Digite x: 10
    Você digitou 10
```

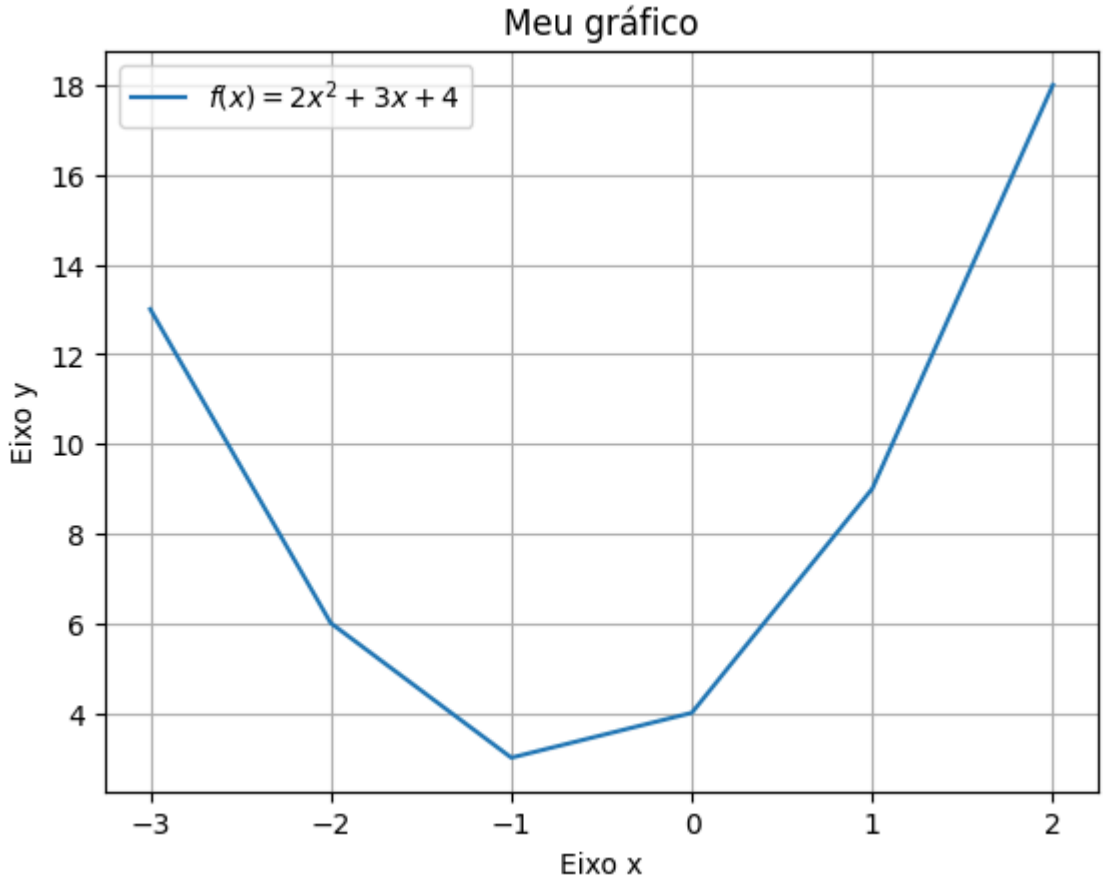
```
x = [-3,-2,-1,0,1,2]

y = []
for val in x:
  y.append(polinomio(val))
print(y)
```

```
    [13, 6, 3, 4, 9, 18]
```

```
import matplotlib.pyplot as plt

plt.plot(x,y)
plt.xlabel('Eixo x')
plt.ylabel('Eixo y')
plt.title('Meu gráfico')
#plt.legend(['Polinômio'])
plt.legend([r'$f(x) = 2x^2+3x+4$']) # Usando render LaTeX
plt.grid()
plt.show()
```

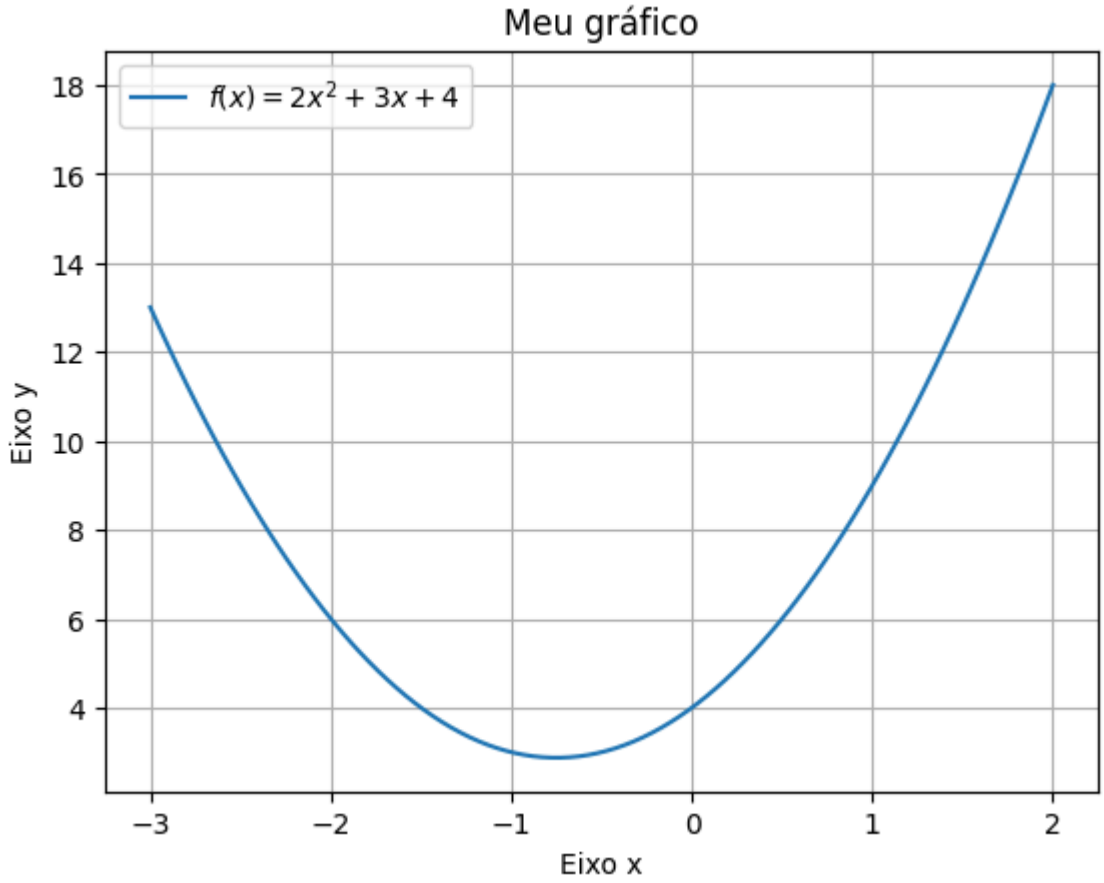


```
import numpy as np
import matplotlib.pyplot as plt

# Função externa para amostragem (mais usual)
x = np.linspace(-3,2,100)

y = []
for val in x:
  y.append(polinomio(val))


plt.plot(x,y)
plt.xlabel('Eixo x')
plt.ylabel('Eixo y')
plt.title('Meu gráfico')
plt.legend([r'$f(x) = 2x^2+3x+4$']) # Usando render LaTeX
plt.grid()
plt.show()
```



```
def f(x):
  a = 2
  b = 3
  c = 4
  return a*x**2 + b*x + c

def g(x):
  return x**2


f(3)
```

```
    31
```

```
g(3)
```

```
    9
```

```
f(g(3))
```

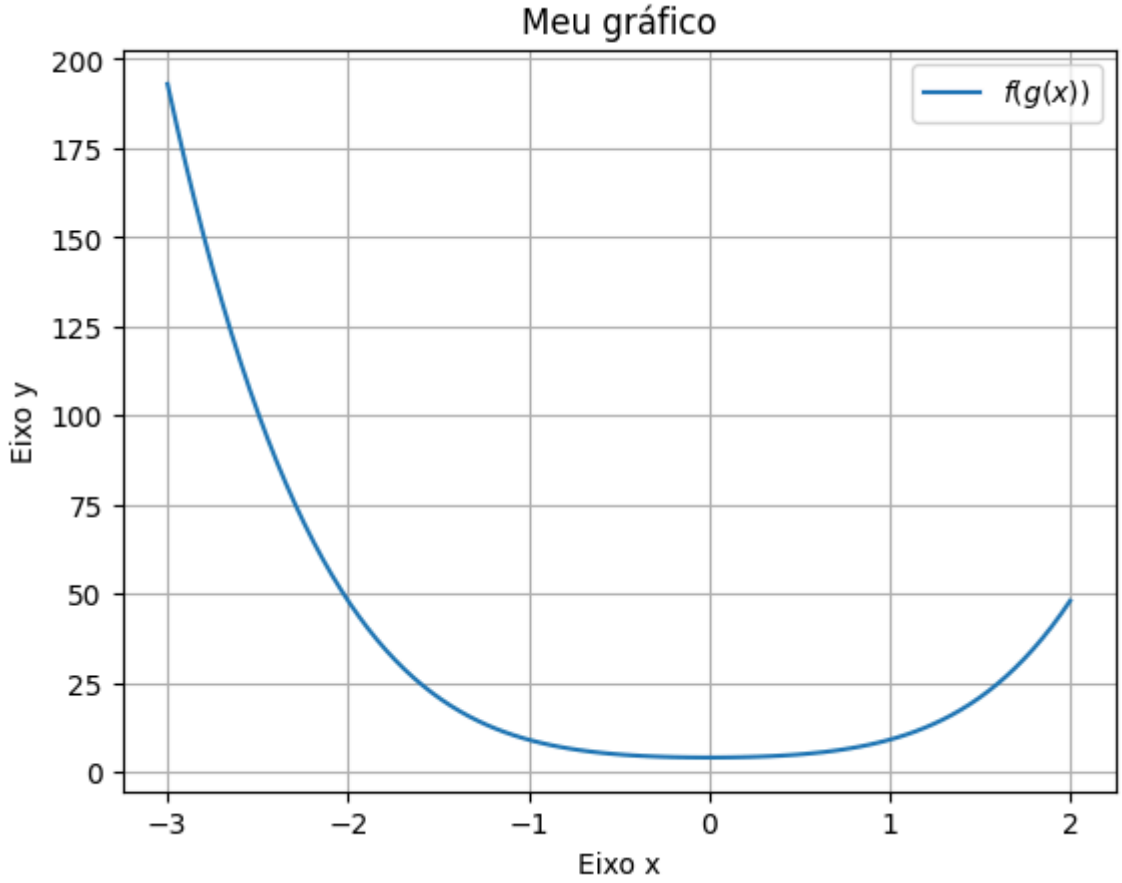
```
    193
```

```
import numpy as np
import matplotlib.pyplot as plt

# Função externa para amostragem (mais usual)
x = np.linspace(-3,2,100)

y = []
for val in x:
  y.append( f(g(val)) )

plt.plot(x,y)
plt.xlabel('Eixo x')
plt.ylabel('Eixo y')
plt.title('Meu gráfico')
plt.legend([r'$f(g(x))$']) # Usando render LaTeX
plt.grid()
plt.show()
```



In

```
!pip install sympy
```

```
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-
```

```
import sympy

# def de uma variavel simbolica
x = sympy.symbols('x')

# definicao de funcao
sympy.Function('f')(x)
```

$$f(x)$$

```
# Defin Funcao Simbolica

f = 2*x**2+3*x+4


# Avaliar funcao ce um ponto especifico
f.subs(x, 3)
```

$$31$$

```
# outras coisa legais
# definicao de uma funcao
h = sympy.Function('h',)(x)
h = x**2-4

sympy.roots(h)


# Funcao Composta

# Definicao de uma funcao
g = sympy.Function('g')('x')
g = x**2

f.subs(x, g)
```