Fatorial

```
def fatorial(n):

    # Verifica se n é negativo
    if n < 0:
        raise ValueError("Fatorial não definido para números
negativos.")

    # primeiro termo
    resultado = 1

    # calcula o fatorial usando um loop for
    for i in range(2, n + 1):
        resultado *= i

    # retorna o resultado do fatorial
    return resultado

# Testes
print(f'O fatorial de 3 é {fatorial(3)}')
print('O fatorial de 1 é', fatorial(1))
f10 = fatorial(4)
print(f'O fatorial de 4 é {f10}')
f0 = fatorial(0)
print('O fatorial de 0 é', f0)
```

```
O fatorial de 3 é 6
O fatorial de 1 é 1
O fatorial de 4 é 24
O fatorial de 0 é 1
```

Fibonacci

```
def fibonacci(n):

    if n <= 0:
        return []  # Retorna uma lista vazia para n <= 0
    elif n == 1:
        return [0]  # Retorna apenas o primeiro termo para n == 1

    # Inicializa os dois primeiros números da sequência
    sequencia = [0, 1]

    # Gera os próximos termos usando um loop for
    for _ in range(2, n):
        proximo = sequencia[-1] + sequencia[-2]
```

```
        sequencia.append(proximo)

    return sequencia

# Testes
print('Sequência dos 5 primeiros números de Fibonacci:', fibonacci(5))
# saída os 5 termos

f = fibonacci(20)
print('Sequência dos 20 primeiros números de Fibonacci:', f)
# saida os 20 termos

numero = 30
f = fibonacci(numero)
print(f'Sequência dos {numero} primeiros números de Fibonacci: {f}')
# saída 30 termos
```

```
Sequência dos 5 primeiros números de Fibonacci: [0, 1, 1, 2, 3]
Sequência dos 20 primeiros números de Fibonacci: [0, 1, 1, 2, 3, 5, 8,
13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181]
Sequência dos 30 primeiros números de Fibonacci: [0, 1, 1, 2, 3, 5, 8,
13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765,
10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229]
```

Bhaskara

```
import math
def bhaskara(a, b, c):


    if a == 0:
        raise ValueError("O coeficiente 'a' não pode ser zero em uma
equação do segundo grau.")

    # Calculando o valor de delta
    delta = b**2 - 4*a*c

    # Verifica o valor de delta
    if delta < 0:
        return "Não há raízes reais.", None

    # Calcula as duas raízes
    x1 = (-b + math.sqrt(delta)) / (2 * a)
    x2 = (-b - math.sqrt(delta)) / (2 * a)

    return x1, x2

# Testes
x1, x2 = bhaskara(2, 12, -14)
```

```
print('A primeira raiz é:', x1)
print('A segunda raiz é:', x2)

# Testando para delta < 0
resultado = bhaskara(1, 1, 1)
print('Resultado:', resultado)

# Testando para a = 0
try:
    bhaskara(0, 2, 3)
except ValueError as e:
    print('Erro:', e)
```

```
A primeira raiz é: 1.0
A segunda raiz é: -7.0
Resultado: ('Não há raízes reais.', None)
Erro: O coeficiente 'a' não pode ser zero em uma equação do segundo grau.
```

Divisores

```
def is_divisor(a, b):

    if b == 0:
        raise ValueError("O divisor não pode ser zero.")
    return a % b == 0

# exemplos
print(is_divisor(10, 2))
print(is_divisor(10, 3))
```

```
True
False
```