```
#1
def calculadora(num1, num2, operacao):
    try:
        if not isinstance(num1, (int, float)) or not isinstance(num2, (int, float)):
            raise ValueError("Os parâmetros devem ser números (int ou float).")

        if operacao == '+':
            return num1 + num2
        elif operacao == '-':
            return num1 - num2
        elif operacao == '*':
            return num1 * num2
        elif operacao == '/':
            if num2 == 0:
                raise ZeroDivisionError("Não é possível dividir por zero.")
            return num1 / num2
        else:
            raise ValueError("Operação inválida. Use '+', '-', '*' ou '/'.")

    except ZeroDivisionError as e:
        return f"Erro: {e}"
    except ValueError as e:
        return f"Erro: {e}"

print(calculadora(10, 10, '+'))
print(calculadora(10, 10, '/'))
print(calculadora(10, '5', '+'))
print(calculadora(10, 5, '%'))
```

```
20
1.0
Erro: Os parâmetros devem ser números (int ou float).
Erro: Operação inválida. Use '+', '-', '*' ou '/'.
```

```
#2. Crie uma função ler_inteiro() que solicita ao usuário um número inteiro.
#Se o usuário inserir um valor inválido (não inteiro), exiba uma mensagem
#e peça a entrada novamente até que um número válido seja fornecido.

def ler_inteiro():
    while True:
        try:
            numero = int(input("Por favor, insira um número inteiro: "))
            return numero
        except ValueError:
            print("Valor inválido! Por favor, insira um número inteiro válido.")
numero = ler_inteiro()
print(f"Você inseriu o número inteiro: {numero}")
```

```
Por favor, insira um número inteiro: 6,5
Valor inválido! Por favor, insira um número inteiro válido.
Por favor, insira um número inteiro: 10
Você inseriu o número inteiro: 10
```

```
#3. Crie uma função calcular_media(numeros) que recebe uma lista de
#números e retorna a média.
#Se a lista estiver vazia, a função deve tratar a exceção e exibir uma
#mensagem adequada.

def calcular_media(numeros):
    try:

        if not numeros:
            raise ValueError("A lista está vazia.")

        media = sum(numeros) / len(numeros)
        return media

    except ValueError as e:
        return str(e)

# Exemplos de uso:
numeros1 = []
numeros2 = [2,5,6,7]

print(calcular_media(numeros1))
print(calcular_media(numeros2))
```

```
A lista está vazia.
5.0
```

```
#4. Crie uma função divisao_segura(a, b) que retorne o resultado da divisão a / b.
#Se b for zero, a função deve retornar Erro Divisão por zero não
#permitida.

def divisao_segura(a, b):
    try:
        resultado = a / b
        return resultado

    except ZeroDivisionError:

        return "Erro: Divisão por zero não permitida."

print(divisao_segura(10, 2))
print(divisao_segura(10, 0))
```

```
5.0
Erro: Divisão por zero não permitida.
```

```
#5

def soma_lista(numeros):
    try:
        soma = sum(numeros)
        return soma

    except TypeError:

        return "Erro: A lista contém valores inválidos."

print(soma_lista([1, 1, 15, 4, 77]))
print(soma_lista([2, 'e', 8, 4]))
```

```
98
Erro: A lista contém valores inválidos.
```

```
#6

def multiplicar(a, b):
    try:
        produto = a * b
        return produto

    except TypeError:

        return "Erro: Ambos os valores devem ser números."
print(multiplicar(3, 4))
print(multiplicar(3, 'a'))
```

```
12
aaa
```

```
#7

def pegar_elemento(lista, indice):
    try:
        elemento = lista[indice]
        return elemento

    except IndexError:

        return "Erro: Índice fora do alcance da lista."
print(pegar_elemento([10, 20, 30, 40], 2))
print(pegar_elemento([10, 20, 30, 40], 5))
```

```
30
Erro: Índice fora do alcance da lista.
```

```
#8

def contar_caracteres(texto, caractere):
```

```
        try:
            if not isinstance(texto, str):
                raise TypeError("Erro: O parâmetro 'texto' deve ser uma string.")

            contador = texto.count(caractere)
            return contador

        except TypeError as e:
            return str(e)
    print(contar_caracteres("abracadabra", "a"))
    print(contar_caracteres("abracadabra", "b"))
    print(contar_caracteres(12345,'c'))
```

```
5
2
Erro: O parâmetro 'texto' deve ser uma string.
```