1. Função Simples: Gerador de Assinatura de E-mail

Objetivo: Criar uma função que imprima uma assinatura de e-mail padronizada no console.

Exercício: Escreva uma função chamada exibirAssinatura que não recebe parâmetros e, quando chamada, imprime um bloco de texto formatado com um nome, cargo e empresa fictícios.

2. Função com Parâmetros: Cálculo de Desconto

Objetivo: Criar uma função que calcula o valor de um produto após um desconto.

Exercício: Desenvolva uma função chamada calcularPrecoComDesconto que recebe dois parâmetros double: precoOriginal e percentualDesconto. A função deve imprimir o novo preço no console.

3. Função com Retorno: Conversor de Moeda

Objetivo: Escrever uma função que converte um valor de Real para Dólar.

Exercício: Crie uma função chamada converterRealParaDolar que não aceita parâmetros, mas considera uma taxa de câmbio fixa (ex: 5.25) e retorna o valor de R$ 100 convertido para dólares. Armazene o resultado em uma variável e imprima-a.

4. Função com Parâmetros e Retorno: Cálculo de IMC

Objetivo: Criar uma função que calcula o Índice de Massa Corporal (IMC) de uma pessoa.

Exercício: Desenvolva uma função chamada calcularIMC que recebe dois parâmetros double: peso (em kg) e altura (em metros). A função deve retornar o valor do IMC calculado (IMC=peso/(altura2)).

5. Arrow Function: Verificador de Maioridade

Objetivo: Usar a sintaxe de seta para uma verificação simples.

Exercício: Crie uma *arrow function* chamada ehMaiorDeIdade que recebe uma idade (inteiro) e retorna true se a idade for maior ou igual a 18, e false caso contrário.

6. Função com Parâmetros Nomeados e Valores Padrão

Objetivo: Criar uma função de login flexível com um valor padrão.

Exercício: Escreva uma função efetuarLogin que recebe um usuario (String) obrigatório. Ela também deve ter um parâmetro nomeado opcional senha. Se a senha não for fornecida, ela deve assumir o valor padrão "123456". A função deve imprimir as credenciais usadas.

7. Função com Parâmetros Posicionais Opcionais: Gerador de Eventos

Objetivo: Criar uma função para agendar eventos onde o local é opcional.

Exercício: Escreva uma função agendarEvento que aceite três parâmetros: titulo (String), data (String) e um parâmetro posicional opcional local (String). A função deve imprimir os detalhes do evento.

8. Função Anônima: Filtrar Notas

Objetivo: Usar uma função anônima para filtrar itens de uma lista.

Exercício: Crie uma lista de double com várias notas de alunos. Use o método .where() da lista para criar uma nova lista contendo apenas as notas dos alunos aprovados (notas >= 6.0). Passe a lógica de verificação como uma função anônima.

9. Função como Parâmetro (Higher-Order): Processador de Dados

Objetivo: Criar uma função que aplica uma operação matemática customizada a cada item de uma lista.

Exercício: Desenvolva uma função chamada processarLista que recebe uma lista de inteiros (List<int>) e uma função operacao. A operacao deve receber um int e retornar um int. A processarLista deve retornar uma nova lista com a operacao aplicada a cada número.

10. Função com Retorno de Coleção: Gerar Tabuada

Objetivo: Criar uma função que retorna uma lista com os resultados da tabuada de um número.

Exercício: Escreva uma função gerarTabuada que recebe um número inteiro numero como parâmetro e retorna uma List<int> contendo os resultados da sua tabuada (de 1 a 10).