Exercícios Gerais Typescript

1. Tipos Básicos

- Exercício 1: Crie variáveis usando os tipos básicos (string, number, boolean, array) e inicialize-as com valores apropriados. Crie uma função que receba esses tipos como argumentos e retorne uma string descrevendo seus valores.
- Exercício 2: Crie um objeto que represente um livro, com propriedades como título, autor, número de páginas, e se está disponível (boolean). Escreva uma função que receba esse objeto e retorne uma frase formatada sobre o livro.

2. Funções e Parâmetros Opcionais/Default

- Exercício 3: Escreva uma função que calcule a área de um círculo, onde o raio é opcional e, se não for fornecido, deve assumir um valor padrão de 1. A função deve retornar a área.
- Exercício 4: Crie uma função que receba dois parâmetros: uma string e um número. O número é opcional e, se fornecido, a função deve repetir a string esse número de vezes. Caso contrário, deve repetir a string duas vezes.

3. Interfaces

- Exercício 5: Defina uma interface Pessoa com as propriedades nome, idade, e profissao. Crie uma função que receba um objeto Pessoa e retorne uma frase descrevendo essa pessoa.
- Exercício 6: Crie uma interface Produto com as propriedades nome, preco, e categoria. Crie uma função que receba um array de objetos Produto e retorne apenas os produtos de uma determinada categoria.

4. Classes

- Exercício 7: Crie uma classe Carro com as propriedades marca, modelo, e ano. Adicione um método que retorne uma descrição do carro. Crie algumas instâncias da classe e chame o método para ver as descrições.
- Exercício 8: Adicione um método estático à classe Carro que conte quantos carros foram criados. Use essa função em um exemplo para contar quantos carros foram instanciados.

5. Herança e Polimorfismo

- Exercício 9: Crie uma classe Animal com um método som() que retorna uma string "O animal faz um som". Crie classes derivadas Cachorro e Gato que sobrescrevem o método som() para retornar sons específicos ("O cachorro late" e "O gato mia"). Instancie os objetos e chame seus métodos.

- Exercício 10: Crie uma classe base Funcionario com as propriedades nome e salario, e um método calcularSalario(). Crie subclasses FuncionarioHorista e FuncionarioAssalariado, cada uma com seu próprio método calcularSalario(). Instancie os objetos e demonstre como o polimorfismo funciona.

6. Generics

- Exercício 11: Crie uma função genérica que receba um array de elementos e retorne o primeiro elemento. Teste a função com arrays de números, strings e objetos.
- Exercício 12: Crie uma classe genérica Caixa que pode conter um valor de qualquer tipo. Adicione métodos para armazenar e recuperar o valor. Teste a classe com diferentes tipos, como números, strings e objetos.

7. Tuplas e Enums

- Exercício 13: Crie uma tupla que represente um ponto no espaço 2D com coordenadas x e y. Escreva uma função que receba essa tupla e retorne a distância do ponto à origem (0, 0).
- Exercício 14: Defina um enum para representar os dias da semana. Crie uma função que receba um valor desse enum e retorne se o dia é um dia útil ou um fim de semana.

8. Manipulação de Arrays e Objetos

- Exercício 15: Crie uma função que receba um array de números e retorne o maior e o menor número como uma tupla.
- Exercício 16: Dado um array de objetos representando produtos, crie uma função que filtre os produtos por preço e retorne um novo array apenas com os produtos que custam mais de um determinado valor.

9. Manipulação de Strings

- Exercício 17: Crie uma função que receba uma string e retorne a mesma string, mas com as palavras em ordem inversa.
- Exercício 18: Escreva uma função que receba uma string e retorne a quantidade de vogais (a, e, i, o, u) nessa string.

10. Promises e Assíncrono

- Exercício 19: Crie uma função assíncrona que simule uma operação de busca de dados em um banco de dados (use setTimeout para simular a demora). A função deve retornar os dados ou lançar um erro caso algo dê errado. Teste a função com async/await.
- Exercício 20: Escreva uma função que faça duas chamadas assíncronas diferentes e, em seguida, combine os resultados em um único objeto. Use Promise.all para executar as chamadas em paralelo.

11. Manipulação de DOM (se aplicável)

- Exercício 21: Crie uma função que manipule o DOM, adicionando um novo elemento com texto dinâmico a um elemento <body> já existente. O texto deve ser passado como parâmetro.
- Exercício 22: Crie uma função que adicione uma lista de itens (<li>) a uma <ul> existente no DOM. Os itens da lista devem ser passados como um array de strings.

12. Tipos Avançados

- Exercício 25: Crie um tipo avançado que combine várias interfaces para representar um usuário com permissões específicas em um sistema. Crie instâncias desse tipo e demonstre o uso de cada propriedade.
- Exercício 26: Escreva uma função que receba um objeto e uma chave, e retorne o valor associado a essa chave. Use tipos condicionais para garantir que a chave exista no objeto.

13. Type Guards e Type Assertions

- Exercício 27: Escreva uma função que receba um parâmetro que pode ser uma string ou um número. Use type guards (typeof) para determinar o tipo do parâmetro e retorne uma mensagem diferente para cada tipo.
- Exercício 28: Crie uma função que receba um objeto Pessoa ou Empresa. Use instanceof para verificar o tipo do objeto e retornar informações específicas com base no tipo.

14. Programação Funcional

- Exercício 29: Usando as funções de array filter, e reduce, crie uma função que receba um array de produtos com preços e quantidades e retorne o valor total dos produtos no carrinho.
- Exercício 30: Usando a função reduce crie uma função “procurar Produtos Por Categoria” que recebe um produto e uma categoria no parâmetro e retorne uma nova lista filtrada.