Capítulo 1: Explorando Funções do Console

1.1 Fácil: Crie uma função que imprima "Olá, mundo!" no console.

1.2 Fácil: Escreva uma função que aceite um parâmetro e o imprima no console.

1.3 Fácil: Implemente uma função que calcule a soma de dois números e a imprima.

1.4 Fácil: Crie uma função que receba uma string e a imprima invertida no console.

1.5 Moderado: Escreva uma função que gere um número aleatório entre 1 e 100 e o imprima.

1.6 Moderado: Implemente uma função que verifique se um número é par ou ímpar e imprima o resultado.

1.7 Moderado: Crie uma função que aceite um array de números e imprima a média deles.

1.8 Difícil: Desenvolva uma função que determine se uma palavra é um palíndromo e imprima o resultado.

1.9 Difícil: Escreva uma função que calcule a sequência de Fibonacci e a imprima.

Capítulo 2: Tipos Básicos

2.1 Fácil: Declare uma variável do tipo string e atribua um valor a ela.

2.2 Fácil: Crie uma variável do tipo number e atribua um número decimal.

2.3 Fácil: Declare uma constante do tipo booleano e atribua um valor verdadeiro.

2.4 Fácil: Defina uma variável com um tipo explícito e atribua um valor.

2.5 Moderado: Declare uma variável com tipo union que aceite string ou número.

2.6 Moderado: Crie um array de strings e atribua valores a ele.

2.7 Moderado: Defina um objeto com tipos explícitos para suas propriedades.

2.8 Difícil: Utilize tipos condicionais para definir um tipo com base em outra variável.

2.9 Difícil: Declare uma função com tipos de parâmetros e retorno explícitos.

Capítulo 3: Arrays

3.1 Fácil: Crie um array de números e imprima seus elementos.

3.2 Fácil: Implemente uma função que some todos os elementos de um array numérico.

3.3 Fácil: Escreva uma função que encontre o maior número em um array.

3.4 Fácil: Crie um novo array removendo os elementos duplicados de outro array.

3.5 Moderado: Desenvolva uma função que ordene um array de números.

3.6 Moderado: Crie uma função que filtre os números pares de um array.

3.7 Moderado: Implemente uma função que concatene dois arrays.

3.8 Difícil: Escreva uma função que encontre a sublista contígua de maior soma em um array.

Capítulo 4: Funções

4.1 Fácil: Crie uma função que receba dois números e retorne a soma deles.

4.2 Fácil: Implemente uma função que verifique se um número é positivo, negativo ou zero.

4.3 Fácil: Escreva uma função que calcule o fatorial de um número.

4.4 Fácil: Crie uma função que receba um array de números e retorne a média.

4.5 Moderado: Desenvolva uma função que receba uma string e retorne a quantidade de vogais.

4.6 Moderado: Implemente uma função que verifique se uma palavra é um palíndromo.

4.7 Moderado: Crie uma função que receba um array de strings e retorne o maior comprimento.

4.8 Difícil: Escreva uma função que realize a busca binária em um array ordenado.

Capítulo 5: Enums

5.1 Fácil: Declare um enum para representar os dias da semana.

5.2 Fácil: Utilize um enum para representar os meses do ano.

PROF

5.3 Fácil: Crie um enum para os pontos cardeais (norte, sul, leste, oeste).

5.4 Fácil: Defina um enum para representar as estações do ano.

5.5 Moderado: Utilize um enum para representar os tipos sanguíneos (A, B, AB, O).

5.6 Moderado: Declare um enum para os planetas do sistema solar.

5.7 Moderado: Crie um enum para os principais tipos de moedas.

5.8 Difícil: Desenvolva um enum para as peças de xadrez.

Capítulo 6: Concatenação com Strings

6.1 Fácil: Crie uma função que concatene duas strings.

6.2 Fácil: Implemente uma função que adicione um sufixo a uma palavra.

6.3 Fácil: Escreva uma função que remova espaços em branco no início e no final de uma string.

6.4 Fácil: Crie uma função que substitua todas as ocorrências de uma letra por outra em uma string.

6.5 Moderado: Desenvolva uma função que inverta as palavras em uma frase.

6.6 Moderado: Implemente uma função que capitalize a primeira letra de cada palavra em uma string.

6.7 Moderado: Crie uma função que remova caracteres especiais de uma string.

6.8 Difícil: Escreva uma função que formate uma data em formato específico.

Capítulo 7: Aninhamento de Tipos

7.1 Fácil: Declare um tipo que represente um endereço, contendo rua, cidade e código postal.

7.2 Fácil: Crie uma função que receba um objeto do tipo endereço e imprima suas propriedades.

7.3 Fácil: Defina um tipo para representar um livro, com título, autor e ano de publicação.

7.4 Fácil: Escreva uma função que aceite um array de livros e retorne o livro mais recente.

7.5 Moderado: Declare um tipo para representar um carro, contendo modelo, ano e um objeto de proprietário.

7.6 Moderado: Crie uma função que aceite um objeto do tipo carro e imprima as informações do carro e do proprietário.

7.7 Moderado: Defina um tipo para representar um curso, com nome, instrutor e um array de alunos.

7.8 Difícil: Escreva uma função que aceite um curso e retorne o aluno com a média mais alta.

Capítulo 8: Combinação de Tipos

PROF

8.1 Fácil: Declare um tipo para representar um usuário, com nome de usuário e e-mail.

8.2 Fácil: Crie uma função que combine dois objetos do tipo usuário.

8.3 Fácil: Defina um tipo para representar um produto, com nome, preço e descrição.

8.4 Fácil: Escreva uma função que aceite dois produtos e retorne um novo produto combinado.

8.5 Moderado: Declare um tipo para representar um evento, contendo título, data e um array de palestrantes.

8.6 Moderado: Crie uma função que aceite dois eventos e retorne um novo evento combinado.

8.7 Moderado: Defina um tipo para representar uma transação bancária, com tipo, valor e data.

8.8 Difícil: Escreva uma função que aceite duas transações bancárias e retorne um novo objeto com o saldo resultante.

Capítulo 9: Laços de Repetição

9.1 Fácil: Crie um loop que imprima os números de 1 a 10.

9.2 Fácil: Implemente um loop que calcule a soma dos números de 1 a 50.

9.3 Fácil: Escreva um loop que imprima os números pares de 2 a 20.

9.4 Fácil: Desenvolva um loop que preencha um array com os quadrados dos números de 1 a 5.

9.5 Fácil: Crie um loop que itere sobre um array de strings e imprima cada uma.

9.6 Moderado: Implemente um loop que calcule o produto dos números de 1 a 10.

9.7 Moderado: Escreva um loop que inverta um array de números.

9.8 Moderado: Desenvolva um loop que encontre o número primo mais próximo de 100.

9.9 Moderado: Crie um loop que imprima os números de Fibonacci até o décimo termo.

9.10 Moderado: Implemente um loop que preencha um array com os números quadrados perfeitos até 100.

9.11 Difícil: Desenvolva um loop que inverta cada palavra em uma frase.

9.12 Difícil: Implemente um loop que calcule a raiz quadrada dos números de 1 a 10.

9.13 Difícil: Escreva um loop que encontre o maior número primo menor que 1000.

9.14 Difícil: Crie um loop que imprima os números palíndromos de 1 a 100.

9.15 Difícil: Desenvolva um loop que calcule a sequência de Fibonacci até o vigésimo termo.

Capítulo 10: Condições Lógicas

10.1 Fácil: Crie uma função que verifique se um número é positivo.

10.2 Fácil: Implemente uma função que determine se um número é par.

10.3 Fácil: Escreva uma função que verifique se uma string é vazia.

10.4 Fácil: Desenvolva uma função que compare duas strings e retorne se são iguais.

10.5 Fácil: Crie uma função que determine se um número é positivo, negativo ou zero.

10.6 Moderado: Implemente uma função que verifique se um ano é bissexto.

10.7 Moderado: Escreva uma função que aceite uma idade e retorne "Criança", "Adolescente", "Adulto" ou "Idoso".

10.8 Moderado: Crie uma função que aceite um número e retorne "Fizz" se for divisível por 3, "Buzz" se for divisível por 5 e "FizzBuzz" se for divisível por ambos.

10.9 Moderado: Desenvolva uma função que determine se uma palavra é um anagrama de outra.

10.10 Moderado: Implemente uma função que verifique se uma pessoa pode votar com base na idade.

10.11 Difícil: Escreva uma função que determine se um número é primo.

10.12 Difícil: Desenvolva uma função que valide se uma senha atende a critérios de segurança (mínimo de caracteres, presença de letras e números, etc.).

10.13 Difícil: Crie uma função que determine se uma palavra é um palíndromo.

10.14 Difícil: Implemente uma função que classifique um triângulo com base em seus lados (equilátero, isósceles ou escaleno).

10.15 Difícil: Desenvolva uma função que valide se uma frase é um pangrama (contém todas as letras do alfabeto).

Capítulo 11: Árvore binária

11.1 Fácil: Inserção Ordenada

- A inserção ordenada em uma árvore binária envolve comparar o valor a ser inserido com o valor do nó atual. Se o valor for menor, insira à esquerda; se for maior, insira à direita. Repita esse processo até encontrar um lugar vazio.

11.2 Fácil: Verificação de Valor

- A verificação se um valor existe em uma árvore binária pode ser realizada de maneira semelhante ao processo de inserção. Compare o valor com o valor do nó atual e navegue para a esquerda ou direita conforme necessário.

11.3 Moderado: Verificando Simetria

- Para verificar se uma árvore binária é simétrica, compare recursivamente os nós à esquerda de um nó com os nós à direita do nó correspondente.

11.4 Moderado: Crie uma árvore binária de 1 a 10000 de forma programatica e então encontre o elemento primo mais próximo de 1000.

Capítulo 12: Classes

12.1 Fácil: Criando uma Classe Básica

- Crie uma classe simples chamada Pessoa com propriedades como nome, idade, e um método para imprimir os detalhes da pessoa.

12.2 Fácil: Adicionando Métodos

- Expanda a classe Pessoa adicionando métodos para aumentar a idade da pessoa e alterar o nome.

12.3 Fácil: Herança

- Crie uma classe Aluno que herde da classe Pessoa. Adicione propriedades específicas do aluno, como matricula e métodos relacionados.

12.4 Fácil: Sobrescrita de Método

- Na classe Aluno, sobrescreva o método de impressão da classe Pessoa para incluir informações específicas do aluno.

12.5 Fácil: Encapsulamento

- Modifique a classe Pessoa para tornar suas propriedades privadas e adicione métodos de acesso (getters e setters).

12.6 Moderado: Classes Estáticas

- Crie uma classe estática Calculadora com métodos estáticos para realizar operações matemáticas simples, como adição, subtração, multiplicação e divisão.

12.7 Moderado: Composição

- Crie uma classe Endereco e uma classe Pessoa que possui uma instância da classe Endereco. Adicione métodos para imprimir os detalhes da pessoa e do endereço.

12.8 Moderado: Interface

- Defina uma interface Veiculo com propriedades comuns a veículos (por exemplo, modelo, ano) e implemente essa interface em classes como Carro e Moto.

12.9 Moderado: Métodos Estáticos em Instâncias

- Adicione um método estático à classe Pessoa que conta o número total de instâncias da classe Pessoa criadas.

12.10 Moderado: Encadeamento de Métodos

- Adicione métodos à classe Calculadora que permitem encadear várias operações em uma única expressão.

12.11 Difícil: Gerenciamento de Herança

- Crie uma classe base Animal com propriedades comuns a animais e, em seguida, crie classes derivadas como Cachorro e Gato. Gerencie adequadamente a herança.

12.12 Difícil: Métodos Abstratos

- Adicione um método abstrato à classe Veiculo chamado calcularConsumo e implemente-o nas classes derivadas.

12.13 Difícil: Métodos de Instância Dinâmicos

- Crie um método de instância na classe Pessoa que adiciona uma nova habilidade à pessoa. As habilidades devem ser dinamicamente gerenciadas.

12.14 Difícil: Manipulação de Data

- Crie uma classe Data que represente uma data com propriedades como dia, mês e ano. Adicione métodos para calcular diferenças entre datas.

12.15 Difícil: Design de Classe Utilizando Padrões

- Implemente uma classe Singleton que garanta que apenas uma instância da classe seja criada.

12.16 Difícil: Composição com Design de Software

- Crie um sistema simples de loja online usando classes. Utilize composição para representar produtos, carrinho de compras e pedidos.

12.17 Difícil: Mixins

- Implemente um mixin chamado Registravel que adiciona um método para registrar informações de uma instância.

12.18 Difícil: Decoradores

- Crie um decorador que logue automaticamente a chamada de todos os métodos de uma classe.

12.19 Difícil: Classes Genéricas

- Desenvolva uma classe genérica chamada Caixa que possa armazenar qualquer tipo de objeto.

12.20 Difícil: Testes Unitários em Classes

- Escreva testes unitários para algumas das classes que você implementou neste capítulo. Utilize uma biblioteca de teste adequada.

Capítulo 13: Interfaces

13.1 Fácil: Criando uma Interface Simples

- Crie uma interface chamada Usuario com propriedades como nome, email e idade.

13.2 Fácil: Implementação de Interface

- Crie uma classe chamada Cliente que implementa a interface Usuario. Adicione métodos e propriedades específicos da classe.

13.3 Fácil: Herança de Interface

- Crie uma interface Animal com propriedades como nome e som. Em seguida, crie uma interface Ave que estenda a interface Animal e adicione propriedades específicas de aves.

13.4 Fácil: Métodos em Interfaces

- Defina uma interface FormaGeometrica com métodos como calcularArea e calcularPerimetro. Implemente essa interface em classes como Circulo e Retangulo.

13.5 Fácil: Parâmetros Opcionais

- Adicione um parâmetro opcional à interface Usuario chamado telefone. Atualize a classe Cliente para incluir ou não esse parâmetro.

13.6 Moderado: Interfaces para Funções

- Crie uma interface chamada OperacaoMatematica que representa uma função matemática. Implemente essa interface em funções como soma, subtracao e multiplicacao.

13.7 Moderado: Múltiplas Interfaces

- Defina interfaces Voo e Nadar para representar comportamentos de animais. Implemente essas interfaces em classes como Pato e Peixe.

13.8 Moderado: Extensão de Interfaces

- Crie uma interface Veiculo com propriedades como modelo e ano. Em seguida, crie uma interface Carro que estenda a interface Veiculo e adicione propriedades específicas de carros.

13.9 Moderado: Index Signatures

- Defina uma interface chamada Dicionario que utilize index signatures para representar um dicionário com chaves e valores de tipos específicos.

13.10 Moderado: Readonly Properties

PROF

- Adicione propriedades readonly à interface Produto para representar informações que não devem ser modificadas após a criação do objeto.

13.11 Difícil: Interface para Componentes Visuais

- Crie uma interface ComponenteVisual que represente propriedades comuns a componentes visuais, como cor e tamanho. Implemente essa interface em classes como Botao e CaixaTexto.

13.12 Difícil: Extensão de Múltiplas Interfaces

- Defina interfaces Envio e Rastreio para representar informações sobre o envio e rastreio de um pacote. Crie uma interface Pacote que estenda ambas as interfaces.

13.13 Difícil: Módulos e Interfaces

- Crie um módulo que exporta uma interface Configuracao para representar configurações gerais do aplicativo. Utilize essa interface em diferentes partes do seu código.

13.14 Difícil: Interfaces para Banco de Dados

- Desenvolva interfaces que representem entidades em um banco de dados (por exemplo, Usuario, Produto). Utilize essas interfaces em funções que interagem com o banco de dados.

13.15 Difícil: Interface para Autenticação

- Crie uma interface Autenticacao com métodos como login e logout. Implemente essa interface em classes como AutenticacaoEmail e AutenticacaoBiometrica.

13.16 Difícil: Interface para UI/UX

- Defina interfaces que representem interações de usuário em uma aplicação, como ClickHandler, Draggable, etc. Implemente essas interfaces em componentes específicos.

13.17 Difícil: Index Signatures Avançado

- Crie uma interface chamada RegistroDinamico que permita adicionar propriedades dinamicamente, com valores de tipos específicos, sem saber de antemão quais serão.

13.18 Difícil: Manipulação de Tipos em Interfaces

- Desenvolva interfaces que utilizem manipulação de tipos para garantir que determinadas propriedades são do tipo desejado (por exemplo, todas as strings).

13.19 Difícil: Interface para Cache de Dados

- Crie uma interface CacheDados com métodos como adicionar, remover e obter. Implemente essa interface em uma classe que gerencie o cache de dados.

13.20 Difícil: Interface para Plugin

- Defina uma interface Plugin que permita a extensão de funcionalidades em um sistema. Implemente essa interface em diferentes plugins.

PROF