            CAP.03.md

1.O que são parâmetros opcionais em funções?

          R: Os parâmetros opcionais são aqueles que podem ou não ser fornecidos ao chamar a função. Para indicar que um parâmetro é opcional, basta adicionar o símbolo ? após o nome do parâmetro.

Parâmetros opcionais em funções são aqueles que não precisam necessariamente ser fornecidos quando a função é chamada. Eles permitem definir um valor padrão que será utilizado se nenhum valor for passado para esse parâmetro. Isso torna as funções mais flexíveis, permitindo que elas sejam chamadas com diferentes quantidades de argumentos.

2.Qual é a principal diferença entre uma função anônima e uma arrow function?

            R: A  principal diferença entre função anônima e arrow function, em sintaxe mais curta, herda o “this” do contexto externo e não pode ser usada como construtora. O arrow function têm vantagens em simplificação de código e possibilidade de “this”.

03.Como você define uma classe em TypeScript?

     R: Em typescript classe é definida usando a palavra chave “class”, uma classe pode conter propriedades (atributos), métodos (funções), construtores, e pode ser usada para criar objetos.As classes são blocos fundamentais para a criação de objetos em TypeScript, elas permitem agrupar dados e comportamentos em uma estrutura unificada.

04.O que é uma interface e como ela é usada em TypeScript?

     R: Em typescript uma interfaces define contratos, a estrutura de um objeto especificando  nomes e tipos das propriedades,  métodos como devem se comportar, Interfaces ajudam a garantir que os objetos sigam um formato específico e também são úteis para facilitar a reutilização.

05.O que é herança e como ela funciona em TypeScript?

     R: A herança em TypeScript permite que uma classe derive de outra, reutilizando e estendendo o comportamento da classe base. Isso facilita a organização do código, permitindo que as subclasses implementem ou sobrescrevam funcionalidades conforme necessário.

06.Explique o conceito de polimorfismo com um exemplo.

    R: o polimorfismo permite que objetos de diferentes classes compartilhem a mesma interface(como exemplo “emitirSom”)  implemente esse método de formas específicas. Isso simplifica o código e aumenta a flexibilidade, permitindo que novas classes sejam integradas facilmente sem alterar o código existente.

07.Qual é a função do encapsulamento em classes?

     R: O encapsulamento protege dados de uma classe, controla acesso e manipulação, promovendo boas práticas de desenvolvimento de código.

08.Como você cria uma função com um valor padrão para um parâmetro?

   R: em typeScript, você pode definir valores padrão para os parâmetros de uma função, ou seja significa que se o argumento correspondente não for fornecido quando a função for chamada, o parâmetro assumirá o valor padrão especificado.

09.O que acontece se você não passar um parâmetro opcional em uma função?

     R:Quando um parâmetro opcional não é passado em uma função TypeScript, seu valor é “undefined” isso permite que você escreva funções mais flexíveis, mas requer que você implemente verificações para garantir que seu código se comporte conforme esperado quando os argumentos opcionais não são fornecidos.

10.Quais são os benefícios de utilizar interfaces ao invés de classes?

   R: Interfaces oferecem maior flexibilidade, implementação de múltiplas interfaces, clareza na documentação e código mais limpo e modular. Classes são melhores para lógica e comportamento.

CAP.04.md

01.O que é um módulo em TypeScript e como ele é utilizado?

            R: Módulos são uma maneira de organizar e reutilizar código em TypeScript, facilitando a manutenção e a escalabilidade dos projetos. Permitem que você separe seu código em diferentes arquivos e componentes

02.Qual a diferença entre export e export default?

            R:   R: a diferença entre export e export default é que o export exporta várias entidade e assim deve ser chamada usando o mesmos nomes. Já o export default  exporta uma única entidade como padrão e pode ser importada com qualquer nome.

03.Como namespaces ajudam a organizar o código?

            R:     R: ajudam a organizar o código como agrupando funções, classes com um único escopo lógico, prevenindo conflitos, isolar diferentes partes do código e reutilizar o código.

04.Como você pode evitar conflitos de nome em um projeto grande?

            R: para evitar conflitos é melhor preferir isolar o código, usar namespaces se precisar de agrupamentos dentro do arquivo,estabeleça classes e interfaces para contextos específicos, importar entidades com o mesmo nome, estruture o  projeto para facilitar a organização, evitar variáveis globais.

05.O que é o Webpack e como ele auxilia no desenvolvimento?

            R: ele  auxilia no desenvolvimento ao automatizar o processo de empacotamento,  otimizar o desempenho, e fornecer ferramentas como  Dev Server para um ciclo de desenvolvimento mais eficiente.

 06.Qual o propósito do Babel em projetos TypeScript?

     R: Babel complementa TypeScript com suporte a funcionalidades modernas e compatibilidade com navegadores antigos.

  07.Como você manipula o DOM com TypeScript?

           R: Ferramentas de build como o Babel são essenciais em projetos TypeScript, pois otimizam o bundle, garantem compatibilidade, melhoram o desempenho e facilitam o desenvolvimento ágil e escalável.

08..Como eventos são tratados em TypeScript?

            R: o tratamento de eventos é feito de forma muito similar ao JavaScript, mas com a vantagem de contar com tipagem estática  e melhor suporte a ferramentas de desenvolvimento, garantindo  maior segurança e previsibilidade ao manipular eventos e seus objetos

09.O que são type assertions e por que são importantes na manipulação de DOM?

         R: type assertion são fundamentais na manipulação do DOM em typescript porque ajudam a garantir que os elementos do DOM estejam sendo manipulados corretamente e com segurança. Eles permitem que o desenvolvedor tenha mais controle sobre os tipos, oferecendo uma maneira de contornar limitações de inferência de tipos do TypeScript.

10.Quais são os benefícios de usar ferramentas de build como Webpack e Babel em projetos TypeScript?

       R: Ferramentas de build como o Babel são essenciais em projetos TypeScript, pois otimizam o bundle, garantem compatibilidade, melhoram o desempenho e facilitam o desenvolvimento ágil e escalável.