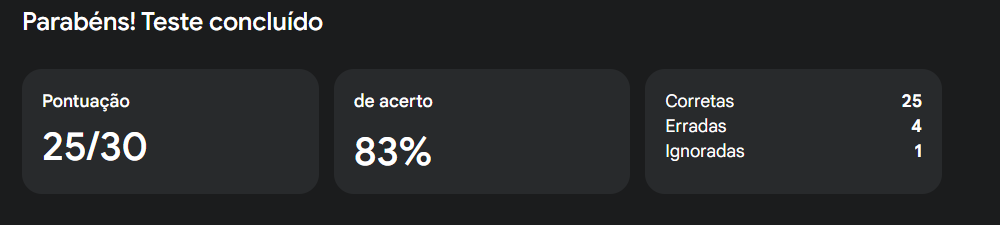
**Simulado**

**Pontuação:**

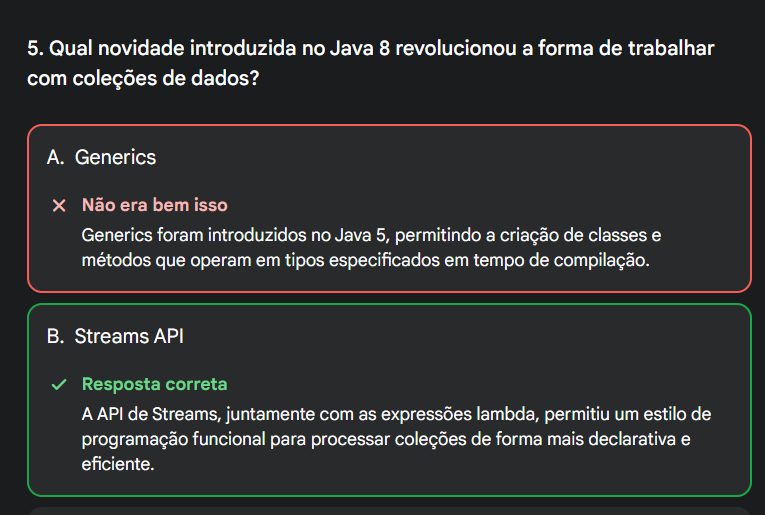
****

**Questões incorretas: 5, 20, 23, 25, 30(“ignorada’)**

**5. Qual novidade introduzida no Java 8 revolucionou a forma de trabalhar com coleções de dados?**

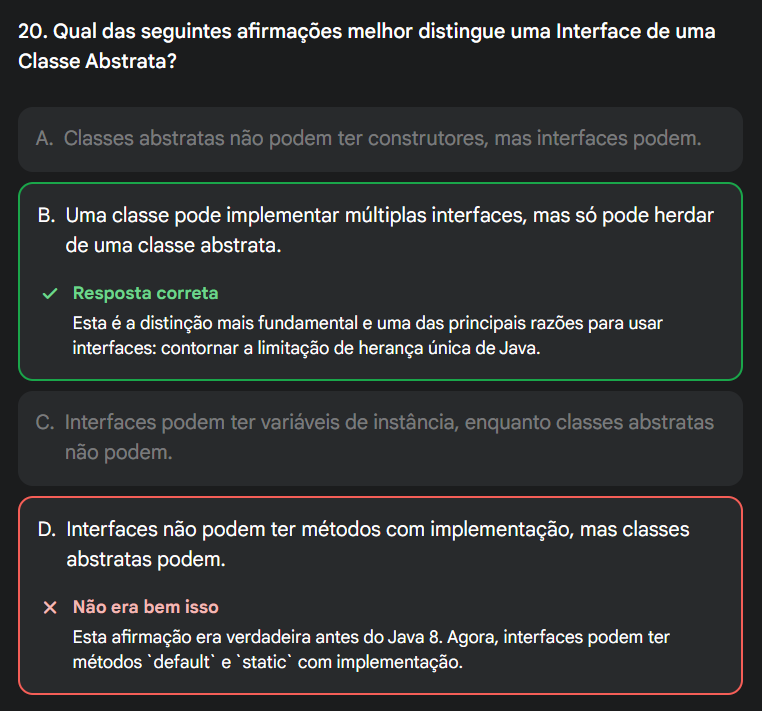
Antes do Java 8, a manipulação de coleções, como filtrar, mapear ou agrupar dados, geralmente exigia loops for ou while verbosos e, muitas vezes, com variáveis auxiliares. Isso tornava o código mais longo e menos legível.

Com a Streams API, o Java passou a adotar uma abordagem mais próxima da programação funcional, permitindo que você processe sequências de elementos de forma declarativa. Em vez de dizer como o processamento deve ser feito (passo a passo em um loop), você diz o que você quer alcançar.

****

**20. Qual das seguintes afirmações melhor distingue uma interface de uma Classes Abstrata?**

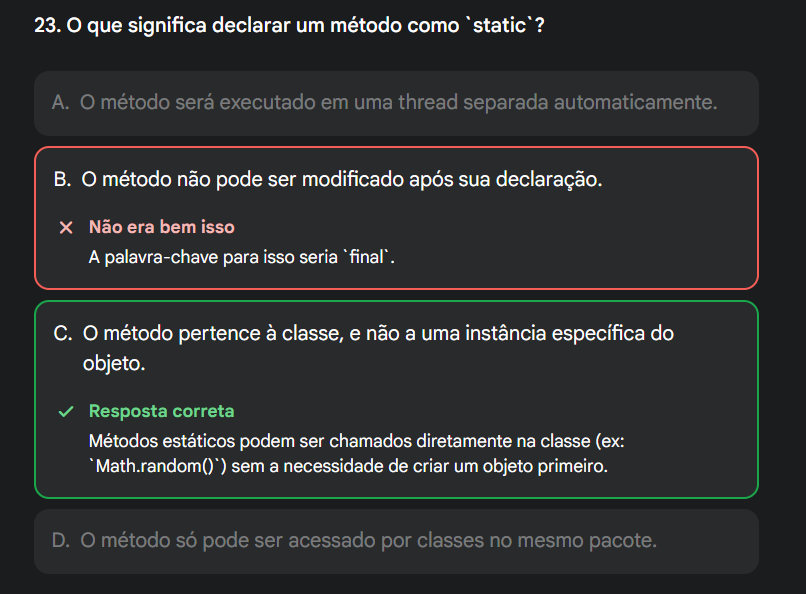
Uma interface define um contrato de comportamento que uma classe deve seguir, permitindo herança múltipla de tipos. Uma classe abstrata, por outro lado, fornece uma estrutura básica e pode incluir implementações de métodos, permitindo herança simples.

****

**23. O que significa declarar um método como ‘static’?**

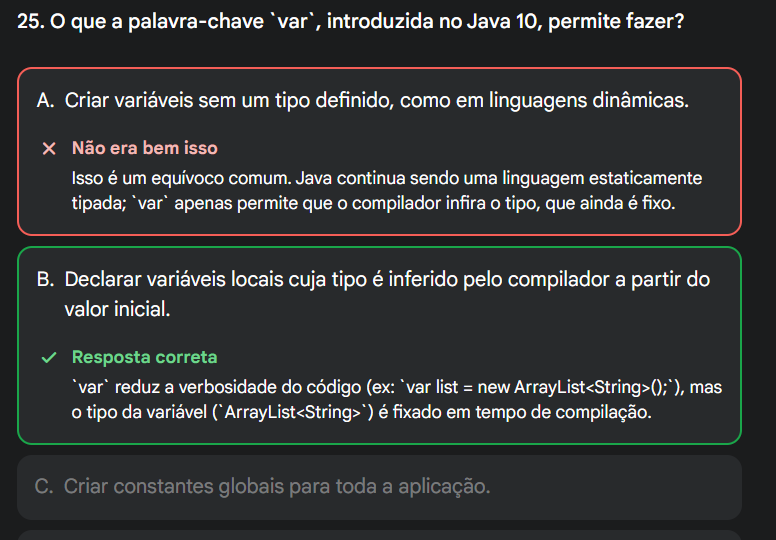
Acesso Direto: Você pode chamar um método static diretamente usando o nome da classe, sem a necessidade de criar um objeto. Isso é muito útil para funções utilitárias que não dependem do estado de um objeto.

Independência de Instância: Métodos estáticos não têm acesso a variáveis de instância (campos não estáticos) da classe. Eles só podem acessar outros membros estáticos (variáveis estáticas e outros métodos estáticos).

****

**25. O que a palavra-chave ‘var’, introduzida permite fazer?**

Permite que você declare uma variável local sem especificar explicitamente o seu tipo. O tipo da variável é inferido automaticamente pelo compilador com base no valor com o qual ela é inicializada. Isso significa que, em vez de você digitar o tipo, o compilador "descobre" qual o tipo correto para você.



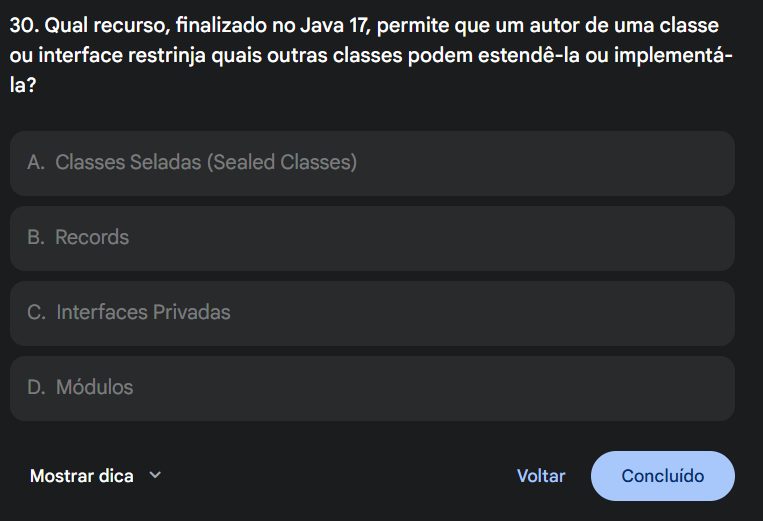
**30. Qual recurso, finalizado no Java 17, permite que um autor de uma classe ou interface restrinja quais outras classes podem estendê-la ou implementá-la?**

**\*Não consegui marcar\***

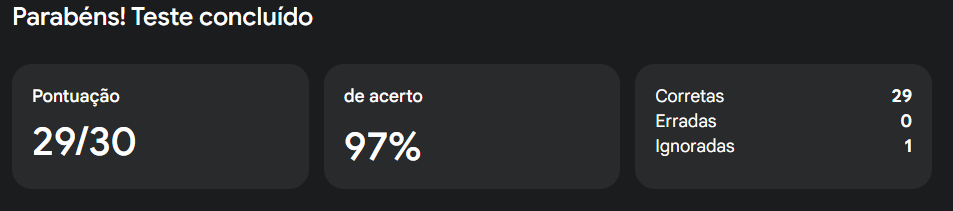
O recurso que permite a um autor de uma classe ou interface restringir quais outras classes podem estendê-la ou implementá-la, e que foi finalizado no Java 17, são as Sealed Classes (Classes Seladas) e Sealed Interfaces (Interfaces Seladas). Esse recurso oferece um nível de controle mais granular sobre a hierarquia de herança. Tradicionalmente, você tinha as seguintes opções:

* **Classes final:** Impedem completamente a herança.
* **Classes regulares:** Permitem a herança para qualquer outra classe.

Com as classes seladas, você pode ter um meio-termo, permitindo a herança, mas apenas para um conjunto de classes ou interfaces que você explicitamente define.

****

**Teste Refeito:**

****

Na Questão 30 não está sendo possível selecionar a opção escolhida.