# Records e classes Sealed

## Records:

* A primeira linha de cada construtor não canônico deve ser uma chamada para outro construtor. Esta regra é violada pelo construtor **sem argumentos** (no-args) no código fornecido. Portanto, ele não será compilado.
* A linha dada deve definitivamente ser inserida em //1. Mas a linha em //2 é inválida porque está tentando modificar variáveis ​​finais. Lembre-se de que todos os campos de instância de um registro são finais. Eles podem ser definidos apenas uma vez e isso também pelo construtor canônico. Outros construtores não podem modificá-los.
* Uma classe record não pode estender outras classes e não pode ser uma classe abstrata.
* Uma classe record não pode ter atributos de instância.
* Em assinaturas de métodos com *record patterns*, ao usar instanceof, é possível utilizar var para simplificar o tipo.
* Um record pode ter qualquer número de construtores sobrecarregados.
* Um record pode ter apenas um parâmetro varargs (ex: String... values).
* Um construtor compacto pode modificar diretamente os valores dos componentes do record.

## Sealed:

* Uma classe sealed nunca pode ser definida como final.
* Uma classe Sealed deve seguir a seguinte estrutura:
  1. Sealed
  2. Class
  3. Implements
  4. permits
* Em uma classe sealed, há uma exceção a essa regra: se todas as subclasses estiverem declaradas no mesmo arquivo de origem que a classe selada, a cláusula permits se torna opcional.
* Uma classe não pode permitir uma classe record, pois um record já estende implicitamente java.lang.Record.
* Uma interface pode ser Sealed.
* Pode-se utilizar permits para subclasses desde que estejam no mesmo pacote ou no mesmo módulo.
* Uma subinterface permitida deve ser declarada como sealed ou non-sealed. Como uma interface não pode ser final, só temos essas duas opções.
* Uma classe/interface sealed deve ter uma cláusula permits.
* Uma interface sealed pode ter o permits implícito, não definido na interface, mas precisa ter pelo menos uma implementação para que o código compile com sucesso.
* Uma subclasse permitida de uma classe sealed deve ser final, non-sealed ou sealed.
* Ao adicionar uma classe na cláusula permits de uma classe selada, é obrigatório que essa classe estenda a classe selada; caso contrário, ocorrerá um erro de compilação.