08

Faça como eu fiz

Nessa aula vimos como criar uma data class para armazenar os dados de livros:

```
data class Livro(
   val titulo: String,
   val autor: String,
   val anoPublicacao: Long,
   val editora: String? = null
)
```

Com a classe `Livro` criada, nós instanciamos alguns objetos em outro arquivo:

```
val livro1 = Livro(
   titulo = "Grande Sertão: Veredas",
   autor = "João Guimarães Rosa",
   anoPublicacao = 1956
)

val livro2 = Livro(
   titulo = "Minha vida de menina",
   autor = "Helena Morley",
   anoPublicacao = 1942,
   editora = "Editora A"
)

val livro3 = Livro(
   titulo = "Memórias Póstumas de Brás Cubas",
   autor = "Machado de Assis",
   anoPublicacao = 1881
)

val livro4 = Livro(
   titulo = "Iracema",
   autor = "José de Alencar",
   anoPublicacao = 1865,
   editora = "Editora B"
)
```

E então, construímos a lista com os livros criados. Fazemos isso de uma forma bem parecida com a criação de arrays, ou seja, chamamos uma função construtora passando os livros como argumento:

```
val meusLivros: MutableList<Livro> =
   mutableListOf(livro1, livro2, livro3, livro4)
```

Depois vimos como adicionar um livro para essa lista:

```
val livro5 = Livro(
   titulo = "Sagarana",
```

```
    autor = "João Guimarães Rosa",
    anoPublicacao = 1946
)

meusLivros.add(livro5)
```

E então como remover:

```
meusLivros.remove(livro3)
```

Para visualizar melhor as alterações na lista de livros, fizemos uma função para imprimir a lista formatada com marcadores indicando cada título e o autor do livro

```
fun MutableList<Livro>.imprimeListaComMarcadores() {
    val listaComMarcadores = this.joinToString(separator = "\n") {
        " - ${it.titulo} de ${it.autor}"
    }
    println("\n ### Lista de Livros ### \n${listaComMarcadores}")
}
```

Agora a gente pode chamar a função para imprimir nossa lista de livros formatada:

```
meusLivros.imprimeListaComMarcadores()
```

Depois de aprendermos a imprimir a lista de livros formatada de uma maneira visualmente mais interessante, vimos como ordenar essa lista implementando a interface `Comparable` :

```
data class Livro(
    val titulo: String,
    val autor: String,
    val anoPublicacao: Long,
    val editora: String? = null
): Comparable<Livro> {
    override fun compareTo(other: Livro): Int {
        return this.anoPublicacao.compareTo(other = other.anoPublicacao)
    }
}
```

Então, foi só chamar a função `sorted` :

```
meusLivros.sorted()
```

Ainda no mesmo tema, vimos uma forma mais simples de ordenar que não depende da implementação de nenhuma interface utilizando a função `sortedBy` :

```
meusLivros.sortedBy { it.anoPublicacao }
```

Juntando todos esses conceitos, nós chegamos em uma função para filtrar a lista de livros pelo nome do autor, ordená-la pelo ano de publicação e transformá-la em uma lista de títulos:

```
fun MutableList<Livro>.titulosPorAnoPublicacaoDoAutor(prefixoAutor: String)
: List<String> {
   return this
       .filter { it.autor.startsWith(prefixoAutor) }
       .sortedBy { it.anoPublicacao }
       .map { it.titulo }
}
```

A função pode ser chamada dessa forma:

```
val titulos: List<String> = meusLivros.titulosPorAnoPublicacaoDoAutor("João")
```