

POO (MiEI/LCC)

2016/2017

Ficha Prática #06

Map<K,V>

Conteúdo

1	Objetivos	3
2	API essencial de Map	3
3	Exercícios	3

1 Objectivos

- Aprender a trabalhar com Set.
- Aprender a trabalhar com Map.

2 API essencial de Map

A API do tipo `Map<K, V>`, comum a ambas as implementações, é apresentada de seguida.

Categoria de Métodos	API de <code>Map<K, V></code>
Inserção de elementos	<code>put(K k, V v);</code> <code>putAll(Map<?extends K, ?extends V> m);</code>
Remoção de elementos	<code>remove(Object k);</code>
Consulta e comparação de conteúdos	<code>V get(Object k);</code> <code>boolean containsKey(Object k);</code> <code>boolean isEmpty();</code> <code>boolean containsValue(Object v);</code> <code>int size();</code>
Criação de Iteradores	<code>Set<K> keySet();</code> <code>Collection<V> values();</code> <code>Set<Map.Entry<K,V>> entrySet();</code>
Outros	<code>boolean equals(Object o);</code> <code>Object clone();</code>

Para mais informações sobre as APIs consulte os apontamentos e a API do Java 8.

3 Exercícios

1. Desenvolva uma classe `Lugar` que represente a informação básica de um lugar de estacionamento, existente num dado parque. Sobre cada lugar pretende ter-se a seguinte informação:

```

1  /** Matricula do veiculo alugado */
2  private String matricula;
3  /** Nome do proprietario */
4  private String nome;
5  /** Tempo atribuido ao lugar, em minutos */
6  private int minutos;
7  /** Indica se lugar é permanente, ou de aluguer
8  */
9  private boolean permanente;
```

Crie em seguida uma classe `Parque` contendo o nome do parque em questão e uma representação dos lugares do parque, associando a cada matrícula, a informação do lugar associado.

Para além dos construtores e métodos usuais, a classe `Parque` deverá definir ainda os seguintes métodos de instância:

- Método que devolve todas as matrículas dos lugares ocupados;
- Método que regista um novo lugar;
- Método que remove o lugar de dada matrícula;
- Método que altera o tempo disponível de um lugar, para uma dada matrícula;
- Método que devolve a quantidade total de minutos atribuídos. Implemente com **iterador interno** e **iterador externo**;
- Método que verifica existe lugar atribuído a uma dada matrícula;
- Método que cria uma lista com as matrículas com tempo atribuído $> x$, em que o lugar seja permanente. Implemente com **iterador interno** e **iterador externo**;
- Método que devolve uma cópia dos lugares;
- Método que devolve a informação de um lugar para uma dada matrícula;

2. Crie uma classe `Países` que estabeleça uma correspondência entre o nome de um dado país e a informação sobre a sua capital (`FichaDeCapital`), designadamente: nome da cidade, população, número de veículos, salário médio (real) e custo de vida mensal médio (real). Implemente os seguintes métodos na classe **Países**:

- Determinar o número total de países;
- Devolver os nomes dos países com capitais com população acima de um valor dado. Implemente com **iterador interno** e **iterador externo**;
- Dado o nome de um país, devolver ficha completa da sua capital;
- Alterar a população da capital de um dado país;
- Inserir a informação de um novo país;
- Criar uma listagem com os nomes de todas as capitais registadas. Implemente com **iterador interno** e **iterador externo**;
- Determinar o somatório de todas as populações das capitais. Implemente com **iterador interno** e **iterador externo**;
- Dada um Map de nome de país para `FichaDeCapital`, para cada país que exista na lista de países alterar a sua ficha de capital e para cada país novo ignorar a sua informação.

- Dada um conjunto de nomes de países, remover as suas fichas de capital;
3. Considerando a classe `ContaPrazo` anteriormente desenvolvida, crie agora uma classe `Banco` que associe a cada código de conta uma **`ContaPrazo`**. A classe **`Banco`** deverá implementar métodos que realizem as seguintes operações:
- Inserir uma nova conta;
 - Determinar o conjunto de códigos das contas pertencentes a dado titular. Implemente com **`iterador interno`** e **`iterador externo`**;
 - O mesmo que o anterior mas para um conjunto de nomes de titulares;
 - Determinar os códigos das contas com capital superior a um valor dado;
 - Criar um Map das contas com taxa de juro superior a um valor dado. Implemente com **`iterador interno`** e **`iterador externo`**;
 - Conjunto dos códigos das contas que vencem juros no dia de hoje. Implemente com **`iterador interno`** e **`iterador externo`**;
 - Dada uma lista de códigos de contas incrementar as suas taxas de juro de um valor X. Implemente com **`iterador interno`** e **`iterador externo`**;
 - Devolver os nomes de todos os titulares de contas. Implemente com **`iterador interno`** e **`iterador externo`**;
 - Criar um Map que associe a cada nome de titular existente o valor total do capital investido nas suas várias contas (use métodos auxiliares);
4. Cada e-mail recebido numa dada conta de mail é guardado contendo o endereço de quem o enviou, a data de envio, a data de receção, o assunto e o texto do mail (não se consideram anexos, etc.). Crie uma classe `MailMap` que associe a cada endereço de envio todos os mails recebidos (cf. classe `Mail`) e implemente as seguintes operações:
- Determinar o total de endereços a partir dos quais se recebeu mail;
 - Guardar um novo mail recebido;
 - Determinar quantos mails têm por origem um dado endereço. Implemente com **`iterador interno`** e **`iterador externo`**;
 - Criar uma lista contendo todos os endereços que enviaram mails contendo no seu assunto uma lista de palavras dada como parâmetro. Implemente com **`iterador interno`** e **`iterador externo`**;
 - O mesmo que a questão anterior, mas criando um conjunto contendo os mails;

- Eliminar todos os e-mails recebidos antes de uma data que é dada como parâmetro. Implemente com **iterador interno** e **iterador externo**;
- Criar uma lista dos endereços que hoje enviaram mails. Implemente com **iterador interno** e **iterador externo**;
- Dada uma lista de palavras, eliminar todos os mails de um dado endereço que no seu assunto contenham uma qualquer destas (anti-spam). Implemente com **iterador interno** e **iterador externo**;
- Eliminar todos os mails anteriores a uma data dada. Implemente com **iterador interno** e **iterador externo**;
- Criar uma listagem com todos os endereços de mail oriundos de Portugal. Implemente com **iterador interno** e **iterador externo**;