semaine 15

1. Feuille de TD

semaine 15 : TD

Le but de ce TD est de créer votre propre implémentation d'un dictionnaire dont les clefs sont des chaînes de caractères et les valeurs des entiers (l'équivalent de Map<String, Integer>).

Version 0

Pour commencer, vous allez créer une classe **MapIUT** contenant une *liste* de *couples* : le premier élément de chaque couple représente la clé, le second la valeur associée. Les getters se nommeront *getFirrst()* et *getSecond()*

- Écrivez la classe **Couple**.
- Définissez la classe **MapIUT** minimale avec un attribut liste de couples.
- Ajoutez une méthode **boolean contains(String clef)** indiquant si une clef se trouve ou non dans le dictionnnaire.
- Définissez une méthode **Integer get(String clef)** retournant la valeur d'une clef du dictionnaire et levant une exception **ClefInconnueException** dans la cas contraire.
- Écrivez la méthode **Integer put(String cle, Integer valeur)**. Si la cle est défà présente dans le dictionnaire, la nouvelle valeur remplace l'ancienne. Cette méthode retournera l'ancienne valeur si la clef existe et null si la clef n'existe pas.
- Écrivez une exécutable permettant de stocker les couples suivants : {("bonjour", 7) ("toto", 5) ("salut", 20) et ("bonjour", 8)}. Écrivez une ligne permettant de récupérer la valeur de la clé "toto".

Modèle

MapIUT ne permet que de faire des dictionnaires dont les clés sont des chaînes et les valeurs des entiers. On va la modifier de manière à ce qu'on puisse définir un dictionnaire dont les clés sont quelconques mais dont les valeurs sont toujours des entiers.

- Créer une classe générique **ModeleCouple**.
- Définissez maintenant la classe **ModeleMapIUT** minimale paramétrée par le type de clef, possédant un *constructeur sans paramètre*, une méthode *contains*, une méthode *put* et enfin une méthode *get*.

Occurrences de mots dans un texte

Imaginez qu'on vous donne un texte et que l'on vous demande de donner, pour chaque mot, le nombre de fois qu'il apparaît dans le texte. Un algorithme pour ce problème serait de parcourir les mots du texte et, pour chaque mot, incrémenter un compteur correspondant au mot.

Cet algorithme serait aisément implémenté à l'aide d'une classe **Compteur** qui stocke les résultats et qui implémente l'interface suivante :

```
public interface CompteurInterface<T> {
   public void incremente(T cle);
   public Integer get(T cle);
}
```

La classe exécutable suivante vous en montre une utilisation

```java
public class Executable {
  public static void main(String [] args) {
      Compteur<String> c = new Compteur<>();
      c.incremente("toto");
      c.incremente("titi");
      c.incremente("toto");
      c.incremente("toto");
      System.out.println(c.get("toto")); // 3
      System.out.println(c.get("titi")); // 1
      System.out.println(c.get("tutu")); // 0
  }
}
```

- En étendant **ModeleMapIUT**, définissez une classe **Compteur** implémentant l'interface ci-dessus. Pour le moment ne donnez le code que du constructeur et de incremente(T cle).
- La méthode **get** doit retourner la valeur d'une clé si celle-ci est dans le dictionnaire et 0 sinon. Comment obtenir ce résultat (donnez le code de get pour la classe Compteur) ?
- Testez vos classes.