

## TP 3 : Classes génériques, collections et exceptions

---

### Exercice 1

Considérons une classe Java appelée **Personne** ayant les attributs suivants :

- **num** : un attribut privé de type **int**
- **nom** : un attribut privé de type **String**
- **prenom** : un attribut privé de type **String**

1. Créez la classe **Personne**
2. Générez les getters et setters de tous les attributs
3. Générez des constructeurs avec et sans paramètres
4. Définissez une méthode **getNomPrenom()** qui retourne le nom concaténé au prénom
5. Créez une classe **Main** contenant la méthode **public static void main (String [] args)**
6. Dans le **main**, créez quelques instances de la classe **Personne** et ajoutez les à un dictionnaire **Map <String, Personne> personnes = new HashMap<>()** : la clé étant la concaténation des nom et prénom et la valeur étant l'instance de **Personne**.
7. Dans une boucle **for**, affichez la clé pour les éléments de chaque itération d'indice pair et la valeur pour les autres éléments.

### Exercice 2

Considérons la classe **Personne** de l'exercice 1.

1. Créez une classe **ListePersonnes** contenant un attribut **Map<String, Personne> personnes = new HashMap<>()**
2. Écrivez une méthode **public void ajouterPersonne(Personne personne)** qui permet d'ajouter **personne** au dictionnaire **personnes** : la clé étant la concaténation des nom et prénom et la valeur étant l'instance de **Personne**.
3. Modifiez la méthode précédente pour qu'elle lève une exception de type **PersonneException** (à créer) si les nom et prénom de l'objet **personne** à ajouter existent déjà dans le dictionnaire.
4. Créez une classe **Main** contenant la méthode **public static void main (String [] args)**
5. Dans **main** :
  - instanciez un objet de la classe **ListePersonnes**,
  - demandez à l'utilisateur de saisir un numéro de personne positif,
  - redemandez-lui de saisir un numéro tant que la saisie est négative ou nulle,
  - demandez-lui de saisir un **nom** et un **prenom**,
  - créez un objet **Personne** avec les valeurs saisies et ajoutez le dans le dictionnaire de **ListePersonnes**,
  - proposez à l'utilisateur de recommencer,
  - si l'utilisateur décide de quitter le programme, affichez tout le contenu du dictionnaire de **ListePersonnes**.

## Exercice 3

Considérons une classe Java appelée **Nombre** ayant les attributs suivants :

- **var1** : un attribut privé de type **int**
- **var2** : un attribut privé de type **int**

1. Créez la classe **Nombre**
2. Générez les getters et setters de tous les attributs
3. Générez un constructeur avec paramètres

Considérons une deuxième classe appelée **Operation** ayant l'attribut suivant :

- **nombre** : un attribut privé de type **Nombre**

4. Créez la classe **Operation**
5. Générez les getters et setters de tous les attributs
6. Générez un constructeur avec paramètre
7. Définissez une première méthode **public int division()** qui retourne le résultat de la division de **var1** par **var2** si ce dernier est différent de zéro. Sinon, elle lève une exception **OperationException** (à créer)
8. Définissez une deuxième méthode **public int racineDeLaSomme()** qui retourne la racine carrée de la somme de **var1** et **var2** si la somme est positive. Sinon, elle lève une exception **OperationException** (à créer)
9. Créez la classe **OperationException**
10. Préparer le ou les constructeurs qui permettront de traiter les exceptions décrites dans les questions précédentes
11. Créez une classe **Main** contenant la méthode **public static void main (String [] args)**
12. Dans **main**,
  - demandez à l'utilisateur de saisir deux entiers,
  - créez un objet de la classe **Nombre**,
  - demandez à l'utilisateur de saisir 1 pour avoir le résultat de la division ou 2 pour la racine carrée de la somme,
  - utilisez la classe **Operation** pour calculer le résultat de l'opération choisie,
  - proposez à l'utilisateur de choisir une autre opération.

## Exercice 4

Étant donné le dictionnaire suivant :

```
Map<Integer, String> stringInt = new HashMap<Integer, String>();
stringInt.put("Paris", 7);
stringInt.put("Rome", 5);
stringInt.put("Manchester", 1);
stringInt.put("Barcelone", 3);
```

1. Écrire un programme Java qui permet de remplacer dans une chaîne de caractères les sous-chaînes figurant comme clés dans le dictionnaire `stringInt` par leurs valeurs respectives. Pour décomposer une chaîne de caractère en un ensemble de sous-chaîne selon une liste de séparateurs donnée, vous pouvez utiliser la classe `StringTokenizer`. Le constructeur de cette classe peut prendre deux paramètres :

- le premier : la chaîne à décomposer en `token`,
- le deuxième : l'ensemble de séparateur.

Cette classe offre certaines méthodes comme `nextToken()` qui retourne un `token` selon les séparateurs choisies et `hasMoreTokens()` qui retourne `true` s'il en reste d'autres tokens, `false` sinon.

**Pour la chaîne**

```
Manchester ou Barcelone mais jamais Paris ni Rome.
```

**Le résultat est :**

```
1 ou 3 mais jamais 7 ni 5.
```

**Exemple avec `StringTokenizer`**

```
String string = "Bonjour tout le monde.voici mon:code;Java";
StringTokenizer sToken = new StringTokenizer(string, " .;");
// les séparateurs ici sont " ", "." et ";"
while(sToken.hasMoreTokens())
    System.out.println(sToken.nextToken());
// affiche
Bonjour
tout
le
monde
voici
mon:code
Java
```

2. Refaire l'exercice en considérant le dictionnaire suivant (clés et valeurs inversées) :

```
Map<Integer, String> intString = new HashMap<Integer, String>();
stringInt.put(7, "Paris");
stringInt.put(5, "Rome");
stringInt.put(1, "Manchester");
stringInt.put(3, "Barcelone");
```