Le type `String` vu précédemment est en fait une classe prédéfinie du langage Java qui définit des "variables" contenant des suites de caractères (des **mots**, des **chaînes de caractères**).

La classe `String` est un type de données composé de méthodes qui permettent la recherche de mots, ou de caractères dans un texte (méthode `substring()`). Les mots peuvent aussi être comparés suivant l'ordre alphabétique (méthode `compareTo()`) ou transformés en d'autres formats (méthode `format()`). La liste complète des méthodes de la classe `String` est décrite sur la page http://docs.oracle.com/javase/7/docs/api/java/lang/String.html.

L'étude des objets de type `String` nous montre qu'une classe est une association de **données** (information, valeur de tout type) **et** de **méthodes** (outils d'accès et de transformation des données). Ces méthodes, définies dans une classe, ne peuvent s'appliquer qu'aux données de cette même classe.

Le langage Java offre aussi la possibilité au programmeur de développer ses propres classes.

Construire une **classe**, c'est définir un nouveau **type**. Pour cela, il est nécessaire de :

- déterminer les caractéristiques communes à ce que l'on souhaite décrire. Ce sont les **données**, les **attributs**, les **propriétés,** ou encore les **membres** de la classe.
- définir toutes les opérations et traitements réalisables sur ces éléments. Ces opérations sont aussi appelées **méthodes** ou encore, **comportements**.

Une **classe** définissant un type structuré, n'est pas une application directement exécutable. Elle ne contient pas de fonction `main()`.

Les types structurés sont utilisés dans les applications, en déclarant des "variables" dont le type correspond au nom de la classe définie précédemment, comme le montre l'instruction suivante :

```
TypeDeLObjet chose = new TypeDeLObjet();
```

L'opérateur **`new`** détermine l'adresse où stocker les informations relatives à l'objet déclaré. Il réserve l'espace mémoire nécessaire pour stocker les données et les méthodes de la classe. Les **données sont initialisées** à `0` pour les entiers, `0.0` pour les réels, `'\0'` pour les caractères et `null` pour tous les autres types structurés.

A cette étape, la variable est appelée dans le jargon informatique un **objet**. Un objet est donc un élément particulier, un représentant d'une classe définissant un type structuré. On dit aussi que c'est **une instance** de la classe. Les données (propriétés ou attributs) qui la définissent sont appelées les **variables d'instance**.

L'accès aux variables d'instance ainsi qu'aux méthodes de la classe se fait par l'intermédiaire de l'opérateur '.' (point). Comme le montre l'exemple suivant :

```
chose.nomDeLaDonnee = valeur du bon type ;
chose.nomDeLaMethode(liste des paramètres éventuels) ;
```

En supposant que la donnée `nomDeLaDonnee` et la méthode `nomDeLaMethode` ont été préalablement définies pour le type de l'objet `chose`.

1. Créer une classe Livre

Sachant qu'un livre est défini par :

- un titre
- les nom et prénom de l'auteur
- un prix
- une catégorie (Policier, SF, Junior, …)
- un numéro ISBN
- un code d'enregistrement construit à partir des deux premières lettres des nom et prénom de l'auteur, de la catégorie du livre et des deux derniers chiffres du code ISBN.

a. Définir les propriétés de la classe `Livre` en mode privé.

b. Écrire les méthodes en `set` et en `get` pour chaque propriété de la classe `Livre`, sauf celle du code d'enregistrement.

c. Décrire le constructeur par défaut qui permet de saisir au clavier dans le terminal les données d'un livre.

d. Décrire un constructeur qui initialise les propriétés d'un livre à partir des valeurs qui lui sont fournies en paramètre.

e. Décrire la méthode `affiche()` qui affiche les propriétés d'un livre.

f. Décrire la méthode `getCodeEnregistrement()` qui calcule le code d'enregistrement d'un livre.

Vous pourrez utiliser la méthode `substring()` de la classe `String`.

`String`	`substring(int beginIndex)` Returns a new string that is a substring of this string.
`String`	`substring(int beginIndex, int endIndex)` Returns a new string that is a substring of this string.

2. Créer une classe GestionLivres

Écrire l'application permettant la saisie et l'affichage des livres sachant qu'une librairie est constituée d'un ensemble de livres, enregistré sous la forme d'un tableau dont la dimension est saisie en cours d'exécution de l'application.

3. Créer une classe Bibliotheque

Une bibliothèque est définie comme un tableau de livres.

a. Définir les propriétés de la classe `Bibliotheque` en mode privé.

b. Écrire le constructeur qui prend en entrée le nombre de livres de la bibliothèque, et crée le tableau de livres, puis appelle une méthode `setTableauLivre()` pour remplir ce tableau.

c. Écrire la méthode `setTableauLivre()` qui remplit chaque case du tableau avec un livre créé par le constructeur `Livre()`.

d. Écrire la méthode `affiche()` qui affiche le contenu de la bibliothèque, c'est à dire l'ensemble des livres qu'elle contient.