R1.01 – TP7(A)
Vecteurs en ArrayList – Tris

L'objectif de ce TP est de :
- *parfaire l'utilisation de vecteurs en ArrayList et la définition d'un ordre naturel sur une classe*
- *travailler les algorithmes de tri sur un vecteur trié (ici sur deux critères)*

CONTEXTE : ROMANS POLICIERS

Comme pour le TP6(B) :

Un roman policier est représenté par l'année de sa première parution, le nom de son auteur et son titre

Vous allez manipuler des `ArrayList` contenant des objets représentant des romans policiers, instances d'une classe `Polar` (la même que celle du TP6(B), à l'exception des méthodes `compareTo` et `toString`.

Nous vous fournissons dans un dossier `TP7A_Files` :

- un fichier texte nommé *Polars2.txt* qui contient les caractéristiques de 193 romans policiers
- la classe `Polar` que vous devrez compléter
- une classe java `CreationBib` qui contient une fonction qui contient une fonction permettant de générer un `ArrayList<Polar>` intialisé avec les données du fichier *Polars2.txt*

ATTENTION : LE FICHIER DE ROMANS POLICIERS EST INITIALEMENT TRIÉ PAR TITRE CROISSANT AU SENS STRICT

Avant de commencer...

- Ouvrez un terminal et placez-vous dans votre répertoire `R1.01`
  - ✓ Exécutez la commande : `cp -r /users/info/pub/1a/R1.01/TP7A_Files .`
  - ✓ Lancez **IJ** et créez un projet `TP7_A`
- Ouvrez le dossier `TP7A_Files` que vous avez copié dans votre répertoire R1.01 et effectuez les actions suivantes :
  1. **copie dans votre projet** `TP7_A` **du fichier** *Polars2.txt*
     - copiez le fichier *Polars2.txt* (clic sur le fichier + CTRL + C)
     - dans la fenêtre de votre projet TP7_A, sélectionnez avec le bouton droit de la souris, l'icône du projet 
     - collez le contenu du presse-papiers (CTRL + V) puis validez (clic sur OK)
  2. **copie dans le répertoire** `src` **de votre projet** `TP7_A` **de la classe java** `CreationBib`
     - copiez le fichier `CreationBib.java` (clic sur le fichier : CTRL + C)
     - dans la fenêtre de votre projet TP7_A, sélectionnez avec le bouton droit de la souris, l'icône du répertoire src
     - collez le contenu du presse-papiers (CTRL + V) puis validez (clic sur OK)
  3. **copie dans le répertoire** `src` **de votre projet** `TP7_A` **de la classe java** `Polar`
     - copiez le fichier `Polar.java` (clic sur le fichier + CTRL + C)
     - dans la fenêtre de votre projet TP7_A, sélectionnez avec le bouton droit de la souris, l'icône du répertoire src
     - collez le contenu du presse-papiers (CTRL + V) puis validez (clic sur OK)
     -

1. Vérification du tri du vecteur par titre croissant

1.1. Dans le projet `TP7_A` créez une classe `Utilitaire`

1.2. Dans cette classe, ajoutez et codez la fonction suivante :

```
public static boolean verifTri(ArrayList<Polar> vPolar) {
        //{ } => { résultat = vrai si vPolar est trié par titre
        //         croissant au sens strict }
```

2. Classe PolarsEtTris

2.1. Créez une classe `PolarsEtTris` et ajoutez-y une procédure `main` dans laquelle vous écrirez :

- la déclaration : `ArrayList<Polar> mesPolars = CreationBib.lesPolars();`
- les instructions nécessaires à la vérification du tri par titre strictement croissant, du vecteur `mesPolars`
  - ✓ affichage d'un message indiquant si le vecteur est bien trié par titre croissant (ou s'il ne l'est pas).
    ce message ne devra pas être réduit à `True` ou `False`, mais rendre compte en français, de l'état du tri du vecteur, selon le résultat retourné par la fonction `verifTri` de la classe `Utilitaire`
  - ✓ affichage du titre de chaque roman représenté dans le vecteur

2.2. Exécutez et vérifiez (à l'œil) que le vecteur **`mesPolars`** est trié par titre de roman

3. Classe Polar

On définit un ORDRE NATUREL sur la classe **`Polar`** :

Les objets de type **`Polar`** sont ordonnés sur deux critères : le nom de l'auteur et, à nom d'auteur égal, l'année de parution

ORDRE (auteur, annee)

3.1. Dans la classe **`Polar`** :

- Modifiez l'en-tête avec l'implémentation de la classe **`Comparable`** pour le type **`Polar`**
- Ajoutez une **fonction de comparaison** entre les objets de type `Polar` : ordre naturel (auteur, annee)

```
@Override
// ordre naturel (auteur, annee)
public int compareTo(Polar o) {
   //{ } => {* résultat = -1
   //          si l'auteur de cet objet précède celui de o
   //          dans l'ordre lexicographique (casse prise en compte)
   //          ou si l'auteur de cet objet est le même que celui de o
   //          et l'année de cet objet précède strictement celle de o
   //        * résultat = 0
   //          si l'auteur de cet objet et celui de o sont les mêmes
   //          et que l'année de cet objet est égale à celle de o
   //        * résultat = 1 sinon
```

- Ajoutez une fonction **de traduction en chaîne de caractères** d'un objet de type `Polar` sous la forme : `(auteur, annee, titre)`

```
@Override
// traduction en chaîne de caractères
public String toString() {
    // {} => {résultat = (auteur, annee, titre)}
```

3.2. Dans la classe `PolarsEtTris`

Collez les instructions suivantes (après celles écrites en **2.1**) :

```
// Intermède : vérification du code de compareTo et de toString
Polar p1 = new Polar(2000, "AUTEUR1", "Une oeuvre");
Polar p2 = new Polar(2000, "AUTEUR2", "Mon oeuvre");
Polar p3 = new Polar(1998, "AUTEUR1", "Oeuvre sans nom");
System.out.print("Roman p1 : "); System.out.println(p1);
System.out.print("Roman p2 : "); System.out.println(p2);
System.out.print("Roman p3 : "); System.out.println(p3);
System.out.println("Comparaison de p1 à p2 : " + p1.compareTo(p2));
System.out.println("Comparaison de p1 à p1 : " + p1.compareTo(p1));
System.out.println("Comparaison de p1 à p3 : " + p1.compareTo(p3));
```

- Exécutez et testez : Vous devez obtenir la trace d'exécution présentée ci-dessous.

```
TRACES D'EXÉCUTION :
    Roman p1 : (AUTEUR1, 2000, Une oeuvre)
    Roman p2 : (AUTEUR2, 2000, Mon oeuvre)
    Roman p3 : (AUTEUR1, 1998, Oeuvre sans nom)
    Comparaison de p1 à p2 : -1
    Comparaison de p1 à p1 : 0
    Comparaison de p1 à p3 : 1
```

- *Si vous n'obtenez pas la bonne trace d'exécution pour les trois premières lignes*, vérifiez et corrigez le code de la méthode `toString` de la classe `Polar`.
- *Si vous n'obtenez pas la bonne trace d'exécution pour les trois dernières lignes*, vérifiez et corrigez le code de la méthode `compareTo` de la classe `Polar`.
- *Si la trace d'exécution est rigoureusement identique à celle proposée*, vous pouvez passer à la suite...

4. Tri par sélection selon l'ORDRE (auteur, annee)

4.1. Dans la classe `Utilitaire`, ajoutez et codez la fonction suivante :

```
public static void triSelect(ArrayList<Polar> vPolar) {
    //{} => {vPolar est trié selon l'ORDRE(auteur, annee)
    //       selon la méthode DU TRI PAR SÉLECTION}
```

4.2. Dans la procédure `main` de la classe `PolarsEtTris`, ajoutez :

- l'appel de la procédure `triSelect`, avec comme paramètre effectif le vecteur `mesPolars`
- l'affichage d'un message annonçant que le vecteur a été trié selon l'ordre (auteur, annee) par la méthode du tri par sélection
- l'affichage de chaque élément du vecteur `mesPolars`

4.3. Exécutez et testez.

4.4. Vérifiez (à l'œil) que le vecteur a bien été trié selon l'ordre (auteur, annee)

5. Tri à bulles amélioré, selon l'ORDRE (auteur, annee)

5.1. Dans la classe `Utilitaire`, ajoutez et codez la fonction suivante :

```
public static void triBulle(ArrayList<Polar> vPolar) {
    //{} => {vPolar est trié selon l'ORDRE(auteur, annee)
    //       selon la méthode DU TRI À BULLES AMÉLIORÉ}
```

5.2. Dans la procédure `main` de la classe `PolarsEtTris`, ajoutez :

- la réinitialisation du vecteur `mesPolars` à sa valeur initiale :
  `mesPolars = CreationBib.lesPolars();`
- l'appel de la procédure `triBulle`, avec comme paramètre effectif le vecteur `mesPolars`
- l'affichage d'un message annonçant que le vecteur a été trié selon l'ordre (auteur, annee) par la méthode du tri à bulles amélioré
- l'affichage de chaque élément du vecteur `mesPolars`

5.3. Exécutez et testez.

5.4. Vérifiez (à l'œil) que le vecteur a bien été trié selon l'ordre (auteur, annee)

6. Tri par insertion, selon l'ORDRE (auteur, annee)

6.1. Dans la classe `Utilitaire`, ajoutez et codez la fonction suivante :

```
public static void triInsertion(ArrayList<Polar> vPolar) {
    //{} => {vPolar est trié selon ORDRE(auteur, annee)
    //       selon la méthode DU TRI PAR INSERTION}
```

6.2. Dans la procédure `main` de la classe `PolarsEtTris`, ajoutez :

- la réinitialisation du vecteur mesPolars à sa valeur initiale :
  `mesPolars = CreationBib.lesPolars();`
- l'appel de la procédure `triInsertion`, avec comme paramètre effectif le vecteur `mesPolars`
- l'affichage d'un message annonçant que le vecteur a été trié selon l'ordre (auteur, annee) par la méthode du tri à bulles amélioré
- l'affichage de chaque élément du vecteur `mesPolars`

6.3. Exécutez et testez. Le vecteur doit avoir été trié selon l'ordre (auteur, annee)