R1.01 – TP5(B)
Vecteurs en ArrayList – Parcours

L'objectif de ce TP est de confirmer votre pratique des vecteurs en ArrayList et des premiers algorithmes de parcours étudiés.

INTRODUCTION – À LIRE ATTENTIVEMENT !!!

Ce TP vous fait aller un peu plus loin sur les `ArrayList` : plutôt que de déclarer et manipuler des `ArrayList` de type `Integer`, `String` ou autre type classe enveloppe, vous allez définir un nouveau type Classe et déclarer et manipuler des `ArrayList` contenant des objets de ce nouveau type.

***EXEMPLE** : déclaration et manipulation d'un arraylist de `Point` (cf. cours 4 et TP4)*

- Déclaration (2 formes) :
  - ✓ `ArrayList<Point> vectPoint = new ArrayList<>();`
  - ✓ `ArrayList<Point> vectPoint = new ArrayList<>(Arrays.asList(unPoint, ..., ...));`

  où `unPoint` *et les éléments sont des objets de type* `Point`
  - *soit préalablement déclarés et initialisés*
  - *soit créés au fil de leur insertion dans la liste avec :* `new Point(valeur_de_x, valeur_de_y)`, *ou par saisie utilisateur*
- Ajout d'un élément dans `vectPoint` (initialisation s'il est vide) :
  1. En fin du vecteur : `vectPoint.add(cePoint);`
  2. À un indice `k` compris entre `0` et `vectPoint.size()` : `vectPoint.add(k, cePoint);`

  où `cePoint` *est un objet de type* `Point`
  - *soit préalablement déclaré et initialisé*
  - *soit créé lors de l'ajout avec :* `new Point(valeur_de_x, valeur_de_y)`, *ou par saisie utilisateur*
- Accès à l'élément d'indice `k` (objet de type `Point` stocké à l'indice `k`) : `vectPoint.get(k)`
- Accès à l'abscisse de l'élément d'indice `k` : `vectPoint.get(k).get(x)`

THÈME DU TP: BILAN SUR UN AN DE LA POLLUTION ATMOSPHÉRIQUE MENSUELLE EN DYOXIDE D'AZOTE DANS UN LIEU DONNÉ

Le dioxyde d'azote (NO_2) est majoritairement émis par le trafic routier.

Pour ce polluant, ont note chaque heure de la journée son maximum de concentration dans l'air (mesuré en µg/m^3), puis on fait la moyenne des maxima.

L'objectif de qualité recommandé en 2021[1] est de 10 µg/m^3 en moyenne annuelle, la valeur limite de concentration à ne pas dépasser pour la protection de la santé humaine[2] étant de 40 µg/m^3.

Un mois est considéré comme pollué (par le dioxyde d'azote) si la concentration relevée dépasse 40 µg/m^3.

Dans ce TP, un relevé mensuel sera représenté par un objet instanciant une classe nommée `ReleveMensuel`

Ses attributs seront le *mois* (*exemples* : *janvier*, *mars*) où il a été établi et la *concentration* moyenne calculée pour ce mois.

Avant de commencer

- Lisez entièrement les questions à tariter
- Ouvrez un terminal et placez-vous dans votre répertoire `R1.01`
- Lancez **IJ** avec la commande `idea`, puis créez un nouveau projet `TP5_B`

[1] https://apps.who.int/iris/bitstream/handle/10665/346555/9789240035423-fre.pdf?sequence=1&isAllowed=y
[2] https://www.ecologie.gouv.fr/sites/default/files/01_Tableau-Normes-Seuils%20r%C3%A9glementaires.pdf

1. Classe releveMensuel

1.1. Dans le projet `TP5_B` créez une classe `releveMensuel`

1.2. Dans la classe `releveMensuel` :

- ✓ Déclarez les **attributs** : `mois` (type `String`) et `concentration` (type `float`)
- ✓ Codez le **constructeur** :

```
public releveMensuel(String mois, float concentration) {
```

- ✓ Codez les **getters**:
- ✓ Ajoutez la fonction suivante (donnée) pour l'affichage d'un objet de type `releveMensuel`

```
public String toString() {
    return mois + " : " + concentration;
```

2. Classe Pollution (première approche d'un ArrayList de releveMensuel)

2.1. Dans le projet `TP5_B` créez une classe `Pollution` et inserez-y une procédure `main`

2.2. Dans la procédure `main` :

- ✓ Déclarez un `ArrayList` de `String`, nommé `lesMois` et initialisé par lot avec la liste :
  `("janvier","février","mars","avril","mai","juin","juillet","août", "septembre","octobre","novembre", "décembre")`
- ✓ Déclarez un `ArrayList` de `releveMensuel`, nommé `testReleve`
- ✓ Ajoutez à `testReleve` : un relevé pour le mois de *janvier*, avec une concentration égale à `20,6`, puis un relevé pour le mois de *mars*, avec une concentration égale à `40`
- ✓ Insérez en deuxième position dans `testReleve` un relevé pour le mois de *février*, de concentration égale à `45`
- ✓ Affichez `testReleve`

2.3. **Exécutez** et **vérifiez** que vous obtenez bien la trace d'exécution suivante :
[janvier : 20.6, février : 45.0, mars : 40.0]

3. Classe Utilitaire

3.1. Dans le projet `TP5_B` créez une classe `Utilitaire`

3.2. Ajoutez et codez dans cette classe les fonctions suivantes :

a) Saisie contrôlée d'un float positif ou nul

```
public static float saisieFloat() {
    //{ } => { résultat = un float saisi par l'utilisateur
    //                    dont la valeur est positive ou nulle }
```

b) Saisie d'un vecteur de relevés mensuels

```
public static ArrayList<releveMensuel>
saisieReleves(ArrayList<String> desMois) {
    // { desMois contient des chaînes représentant des mois de l'année }
    // => { résultat = un vecteur de releveMensuel
    //      Pour chaque élément du vecteur résultat:
    //      * mois est l'élément de même indice dans desMois
    //      * concentration est un float >= 0 saisi par l'utilisateur }
```

c) Moyenne des concentrations

```
public static float moyConc (ArrayList<releveMensuel> desReleves) {
    // { } => { résultat = moyenne des concentrations des éléments
    //                     du vecteur desReleves }
```

d) Concentration maximale

```
public static float maxConc(ArrayList<releveMensuel> desReleves) {
    //{ desReleves non vide } =>
    //       {résultat = concentration la plus élevée dans desReleves}
```

e) Concentration minimale

```
public static float minConc(ArrayList< releveMensuel> desReleves) {
    //{ des RelevesNonVide } =>
    //       {résultat = concentration la moins élevée dans desReleves}
```

f) Indicateur de pollution

```
public static boolean estPollue(ReleveMensuel unReleve, float seuil) {
    //{ } => {résultat = vrai si la concentration de unReleve
    //                   est supérieure à seuil }
```

g) Nombre de mois pollués

```
public static int nbMoisPollue(ArrayList<releveMensuel> desReleves,
                               float seuil) {
    // { } => { résultat = nombre de mois pollués dans desReleves }
```

h) Nom du premier mois pollué

```
public static String premMoisPol(ArrayList<releveMensuel> desReleves,
                               float seuil) {
    // { } => { résultat = nom du premier mois pollué dans desReleves,
    //                     une chaîne vide s'il n'y en a pas }
```

i) Niveau de pollution

```
public static String niveauPol(ReleveMensuel unReleve) {
    // { } =>
    //    { résultat = * "bon" si concentration <= 10
    //                 * "moyen" si concentration dans ]10, 25]
    //                 * "dégradé" si concentration dans ]25, 40]
    //                 * "mauvais" si concentration dans ]40, 55]
    //                 * "très mauvais" si concentration dans ]55, 70]
    //                 * "extrêmement mauvais" si concentration > 70 }
```

4. Classe Pollution (saisie et analyse d'un bilan annuel de relevés)

4.1. Dans la procédure `main` de la classe `Pollution`

- ✓ Mettez en commentaire les déclarations et instructions précédemment écrites à l'exception de la déclaration du vecteur `lesMois`
- ✓ Déclarez :
  - une constante de type float nommée `SEUIL_ALERTE` et initialisée à `40.0f`
  - un `ArrayList` de `ReleveMensuel`, nommé `bilanAnnuel`
- ✓ Ajoutez les instructions suivantes :
  - Initialisation par saisie de `bilanAnnuel` avec en paramètre le vecteur `lesMois`
  - Affichage de `bilanAnnuel`
  - Affichage des différents indicateurs sur l'année : *moyenne annuelle de la concentration en dioxyde d'azote, nombre de mois pollués* (leur concentration doit être supérieure au seuil d'alerte), *premier mois pollué* (s'il y en a un), *concentration maximum* et *concentration minimum*
  - Pour chaque mois de l'année, affichage de son *niveau de pollution*

4.2. **Exécutez et testez**

5. Comparaison de bilans annuels sur deux années dans un même lieu

5.1. Dans la classe `Utilitaire` ajoutez la fonction suivante :

```
public static int compareReleves(ArrayList<ReleveMensuel> desReleves1,
                                 ArrayList<ReleveMensuel> desReleves2
                                 float seuil) {
    // { il y a autant d'éléments dans desReleves1 et desReleves2 }
    // => { Résultat = * -1 si la moyenne des concentrations de desReleves1
    //                   est inférieure à celle de desReleves2, ou si les moyennes
    //                   sont égales et le nombre de mois pollués de desReleves1
    //                   est inférieur à celui de desReleves2
    //                 * 0 si les moyennes des concentrations des deux vecteurs
    //                   sont égales et s'ils ont le même nombre de mois pollués
    //                 * 1 dans tous les autres cas }
```

5.2. Dans la classe `Pollution` :

- ✓ Déclarez un nouvel `ArrayList` de `ReleveMensuel`, nommé `bilanAnnuel2`
- ✓ Initialisez-le par saisie
- ✓ Affichez-le, ainsi que sa *moyenne annuelle* et *son nombre de mois pollués*
- ✓ Affichez le résultat de la comparaison entre les vecteurs `bilanAnnuel` et `bilanAnnuel2`

  NOTE : ce résultat sera affiché sous la forme d'un message

  Exemple : Ce nouveau bilan montre une dégradation (*resp.* une amélioration *ou* la stabilité) de l'émission de dioxyde d'azote dans l'air, relativement au premier bilan