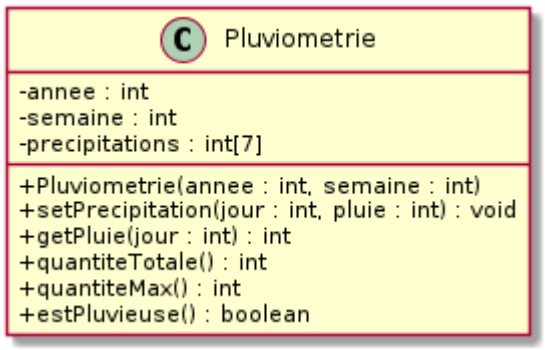
Listes - equals - contains

Exercice 1 : Pluviométrie

Voici un extrait des données pluviométriques sur Orléans en 2022. Les précipitations sont données en mm et un « - » indique que la mesure n'est pas disponible.

	lundi	mardi	mercredi	jeudi	vendredi	samedi	dimanche
semaine 35	3	•	0	0	16	3	0
semaine 36	11	0	2	0	0	0	0
semaine 37	0	1	0	0	•	•	0
semaine 39	6	4	5	1	1	4	0

On veut développer une application qui permet d'analyser la pluviométrie sur une semaine. Pour cela, on décide d'utiliser une classe **Pluviometrie** dont on donne la représentation UML.



Une semaine est considérée comme pluvieuse s'il a plus deux jours consécutifs.

On vous donne

```java
public class Executable{
   public static void main(String[] args){
     Pluviometrie s35 = new Pluviometrie(2022, 35);
     s35.setPrecipitation(0, 3);
     s35.setPrecipitation(2, 0);
     s35.setPrecipitation(3, 0);
     s35.setPrecipitation(4, 16);
     s35.setPrecipitation(5, 3);
     s35.setPrecipitation(6, 0);
     System.out.println(s35.getPluie(1));
     //null
     System.out.println(s35.quantiteTotale());
     //22
     System.out.println(s35.quantiteMax());
     //16
     System.out.println(s35.estPluvieuse());
     //true
   }
}
```

- Un diagramme UML étant indépendant du langage, on n'a pas fait la distinction entre *int* et *Integer*. Sachant que ce projet doit être implémenté en Java, identifiez les endroits où on peut utiliser un type primitif *int* et les endroits où il est nécessaire d'utiliser une classe enveloppe *Integer*.
- Recopiez le code de la classe Executable puis écrivez le CODE MINIMAL de la classe Pluviometrie pour que le projet compile.
- Complètez le code du constructeur ainsi que le code de la méthode *setPrecipitation()*. Pour le moment les autres méthodes ne font RIEN.
- Complètez l'exécutable avec d'autres données et ajoutez des tests pour chacune des méthodes à implémenter. Vérifiez que tous les tests ECHOUENT.
- Complètez le code du/des constructeur(s), puis écrivez le code de autres méthodes une à une. À chaque étape, compilez et vérifiez que les «pseudo-tests» qui sont dans l'exécutable passent.

Exercice 2 : Modélisation d'une cave à Vin

Dans cet exercice, on cherche à modéliser une cave à vin. Pour cela, on crée une classe **Bouteille** permettant de modéliser une bouteille. On doit pouvoir exécuter le code suivant. Pensez à ajouter des tests.

```
public class ExecutableBouteille{
  public static void main(String[] args){
      Bouteille pomerol = new Bouteille("Bordeaux", "Pomerol", 2007);
      assert "Bordeaux".equals(a.getRegion());
      assert "Pomerol".equals(a.getAppellation());
      assert 2007 == pomerol.getMillesime());
      Bouteille pomerol2007 = new Bouteille("Bordeaux", "Pomerol", 2007);
      Bouteille pomerol2003 = new Bouteille("Bordeaux", "Pomerol", 2003);
      assert pomerol.equals(pomerol2007);
      assert !pomerol.equals(pomerol2003);
  }
}
```

- Écrivez le code de cette classe et vérifiez que les pseudo-tests de l'exécutable passent.

On va maintenant modéliser une cave à l'aide d'une classe **Cave**. On veut pouvoir exécuter le code suivant :

```
public class ExecutableCave{
   public static void main(String[] args){
     // reprendre le code de la question précédente
     Cave maCave = new Cave();
     maCave.ajouteBouteille("Bordeaux", "Pomerol", 2005);
     maCave.ajouteBouteille("Bordeaux", "Pomerol", 2007);
     maCave.ajouteBouteille("Bourgogne", "Nuits St George", 2001);
     maCave.ajouteBouteille("Savoie", "Pinot Noir", 2012);
     maCave.ajouteBouteille("Bordeaux", "Pomerol", 2007);
     maCave.ajouteBouteille("Loire", "Chinon", 2017);

     assert 6 == maCave.nbBouteilles();
     assert 3 == maCave.nbBouteillesDeRegion("Bordeaux");

     Bouteille bouteille = maCave.plusVieilleBouteille();
     assert "Nuits St George".equals(bouteille.getAppellation());
     assert maCave.contient("Bordeaux", "Pomerol", 2007);
     assert !maCave.contient("Bordeaux", "Pomerol", 2003);
   }
}
```

- Écrire le code minimal de la classe *Cave* pour que le projet compile. Pour le moment ce code ne fait RIEN et les tests doivent ÉCHOUER.

- Complètez le code du/des constructeur(s), puis écrivez le code de autres méthodes une à une. À chaque étape, compilez et vérifiez que les «pseudo-tests» qui sont dans l'exécutable passent. Pensez à compléter l'exécutable pour ajouter des tests.

Exercice 3 : bonus

Implémentez les exercices de la feuille de TD.