1. (`package`) Reprendre le TP n°2 et
   - placer la classe `Refrigerateur` dans un package `appareils`,
   - créer une classe `Aliment` et la placer dans un package `aliments`, cette classe permettra d'encapsuler des données comme le nom de l'aliment, son poids et son volume (penser aux constructeurs),
   - modifier la classe `Refrigerateur` en conséquence, de sorte que l'on puisse ajouter un aliment *via* la méthode `addAliment` qui maintenant doit prendre en paramètre un aliment... Ajouter la méthode `listeAliments()` qui affiche l'ensemble des aliments contenus dans le réfrigérateur.
   - placer le reste du code (`main`, etc) dans un package `principal`.
   - modifier/créer les méthodes `toString` pour les classes de sorte que l'on puisse afficher correctement un réfrigérateur ou un aliment.

   On doit obtenir quelque chose comme :

```
// dans le main
Refrigerateur frigo = new Refrigerateur("Faure", 300, 70, 170, 80, LocalDate.of(2021,5,1));
Aliment a = new Aliment("Petits pois",1,1);
frigo.addAliment(a);
frigo.listeAliments();
System.out.println(frigo);
```

2. (`static`) Reprendre le TP n°2,
   - compléter la classe `Refrigerateur` de sorte qu'on puisse à tout instant connaître l'ensemble des `Refrigerateur`s créés,
   - modifier en conséquence les méthodes statiques `afficheRefrigerateurs` (qui ne doivent donc plus recevoir d'argument),
   - compléter la classe `Aliment` de sorte qu'on puisse à tout instant connaître l'ensemble des `Aliment`s créés, *via* une méthode statique qui permet d'afficher l'ensemble des aliments `void afficheTousLesAliments()`.

3. (exception) Modifier le code de sorte que l'on puisse, par un petit menu, choisir de créer un réfrigérateur ou un aliment en saisissant à chaque fois les données au clavier à l'aide (uniquement) de la méthode `nextLine()` de `Scanner`. Il faudra s'assurer de la bonne conversion des données numériques pour la date de fabrication des réfrigérateurs. D'autre part, on utilisera la classe `java.time.LocalDate` pour gérer cette date. Pour créer une date on utilisera la méthode statique `LocalDate.of(int year,int month,int dayOfMonth)`. Celle-ci pouvant générer une exception de type `java.time.DateTimeException` si les données de date sont incorrectes, type mois n°13 ou jour 30 en février... On rendra les choses agréables à l'utilisateur : messages d'erreur le guidant en cas d'erreur de saisie, etc.

4. (exception) Modifier le code de sorte que l'on puisse choisir parmi les aliments, l'un d'entre eux et le ranger dans un réfrigérateur choisi aussi. Bien entendu, il faudra gérer le problème de conversion de la chaîne en valeur numérique!

5. modifier le menu de sorte que l'on puisse choisir un réfrigérateur et dans celui-ci un aliment à y retirer, etc.