L04 Exercise:

Create a java program in the following way:  
The program will be about the interaction between seagulls and turtles.

1. Create the interface “Expirable” which defines one abstract function “cycle(): void” and can be specified as what should happen with an expirable object during one cycle of “life”. See below.
2. Create an abstract Turtle class. This class contains information about a turtle: age, name, speed, diet (enum), stomach.  
   - When a new turtle is created, it has a chance to be snatched and eaten by a Seagull. This is random, and the chance of it happening decreases the higher the turtle’s speed.  
   If a turtle is indeed eaten by a seagull, the console should output “The cycle of life can be cruel.”, otherwise, if the turtle survived, “A turtle has made into the water!”.  
   - If there is a turtle instantiated with an age greater than 200, throw the custom “AgeIsHigherThan200Exception”  
   - Turtles have the “eat(Food food): boolean” which returns if the turtle could eat the food given. In case yes, its stomach should fill with the amount (see Food class).  
   - Turtles feel time passing, so they implement the method “cycle(): void” which decreases the amount of food inside a turtle’s stomach, and increases its age by 1. If a turtle reaches the age of 200, it dies, and the program should output “The cycle of life can be cruel.”  
     
   - Use a separate method for printing on the console, no printing should be found in your functions “constructor(), eat(), cycle()”.
3. Create concrete implementations of the Turtle class, for example, SnappingTurtle.
4. Create the abstract Food class. This class contains an enum with the type of diet the food fulfils (Fish food => Carnivore diet). Also contains amount.  
   - “edible(DietType dietType): boolean” returns whether a certain food is edible with the diet that was given. (this method should be used in Turtle’s “eat(Food food): boolean”  
   - Food is also expirable, but different food expire in different ways, so there should be no implementation here.  
   - Food can also become spoiled once expired. Spoiled is a state.
5. Create 2 implementations for Food, for example “Fish” and “Seaweed”.  
   Both should “expire” differently, for example fish expires quicker, while seaweed doesn’t expire, it just dries out after a time, but continues to age forward.
6. In your Program class (which contains the Main method) create a function that takes in a Collection of food and cleans out the spoiled ones.

