# 03. Fishing Pond

A cartoon of a person fishing

Description automatically generated

*It is a beautiful autumn day. You have gone out into nature to enjoy your favourite hobby - fishing.*

**Preparation**

Download the skeleton provided in Judge. **Do not** change the **packages**!

**Pay attention to name the package fishingPond, all the classes, their fields and methods the same way they are presented in the following document. It is also important to keep the project structure as described.**

**Problem description**

Your task is to create a repository that stores fish by creating the classes described below.

**Fish**

First, write a class **Fish** with the following properties:

* **species: String**
* **age: int**
* **matingSeason: String**

The class **constructor** should receive **species, age and matingSeason**. You need to create the appropriate **getters and setters**. All fish species will be **unique.** It is guaranteed that there **will be no duplicates** of species.

Override the **toString()** method in the following format:  
**"This {species} is {age} years old and reproduces through {matingSeason}."**

**Pond**

**Next**, write a class **Pond** that has **data** (a List which stores the entity **Fish**). All entities inside the repository have the **same properties**. Also, the **Pond** class should have those **properties**:

* **capacity: int**
* **fishes: List<Fish>**

The class **constructor** should receive **capacity.** Also, it should initialise the **fishes** with a new instance of the collection.Implement the following features:

* **Method addFish(Fish fish)** – **adds** an **entity** to the data **if** an **empty space** for it
* **Method removeFish(String species)** – removes a fish by **given species,** if such **exists**, and **returns boolean** (true if it is removed, otherwise – false)
* **Method getOldestFish()**– **returns the oldest fish**
* **Method getFish(String species)** – **returns** the **fish** with the **given species**
* **Method getCount** – **returns** the **number** of **fishes**
* **Method getVacancies –** **returns** the number of **remaining** **vacancies** in the pond
* **Method report()** – **returns** a **string** in the following **format** (print the fishes in **order of addition**):
  + **"Fishes in the pond:  
    This {species} is {age} years old and reproduces through {matingSeason}.  
    This {species} is {age} years old and reproduces through {matingSeason}.  
    (…)"**

**Constraints**

* The **species** and **age** of the fish will always be **unique**.
* You will always have a fish added before receiving methods manipulating the Ponds's fish.

**Examples**

This is an example of how the **Pond** class is **intended to be used**.

|  |
| --- |
| **Sample code usage** |
| //Initialize the repository (Pond)Pond pond = **new** Pond(5); //Initialize entities (Fish)Fish trout = **new** Fish(**"**Trout**"**, 5, **"**winter**"**);  Fish perch = **new** Fish(**"**Perch**"**, 2, **"**summer**"**);  Fish pike = **new** Fish(**"**Pike**"**, 4, **"**spring**"**);  Fish catfish = **new** Fish(**"**Catfish**"**, 8, **"**summer**"**); //Add Fishpond.addFish(trout);  pond.addFish(perch);  pond.addFish(pike);  //Remove FishSystem.***out***.println(pond.removeFish(**"**Trout**"**)); //trueSystem.***out***.println(pond.removeFish(**"**Carp**"**)); //false  System.***out***.println(pike.getMatingSeason()); //spring  //Add Fishpond.addFish(catfish);  //Get the oldest fishSystem.***out***.println(pond.getOldestFish().getSpecies()); //Catfish  //Get pond vacanciesSystem.***out***.println(pond.getVacancies()); //2 System.***out***.println(pond.report()); //Fishes in the pond: //This Perch is 2 years old and reproduces through summer.  //This Pike is 4 years old and reproduces through spring.  //This Catfish is 8 years old and reproduces through summer. |

**Submission**

Submit **single .zip file**, containing **fishingPond** package, **with the classes inside** (**Pond** and **Fish** and the **Main** **class)**, there is no specific content required inside the **Main** class e. g. you can do any kind of local testing of you program there. However, there should be **main(String[] args)** method inside.