# Aquarium Adventure



*You and your friends are told that the local aquarium is a must see attraction, so you decide to visit it. To remember the adventure, you decide to make a report for the aquarium.*

## Preparation

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

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

## Problem description

Your task is to create repository (aquarium) which stores departments by creating the classes, described below.

### Fish

First, write a Java **class**, called **Fish** with fields:

* **name: String**
* **color: String**
* **fins: int**

The **constructor** of Fish class should receive **(name, color, fins)**.

The class should also have the following methods:

* Getter **getName()**;
* Getter **getColor()**;
* Getter **getFins()**;
* Override **toString()** method in the format:

**"Fish: {name}**

**Color: {color}**

**Number of fins: {fins}"**

### Aquarium

Next step is to write **Aquarium** class that has a **collection** of object of type **Fish** with corresponding **unique** **name** of a fish. The name of the collection should have name **fishInPool**. All the entities of the fishInPool collection have the **same** fields. The Pool has also some additional fields:

* **name: String**
* **capacity: int**
* **size: int - the volume of the pool**

The **constructor** of the Aquarium class should receive **(name, capacity, size)**, also you should initialize the **collection** of fish with a new instance.

Implement the coming features:

* Getter **getName()**
* Getter **getCapacity()**
* Getter **getSize()**
* Getter **getFishInPool()** -returns the **number** of fish in the current pool
* Method **add(Fish fish)** - add the entity **if** there **isn't** a fish with the same **name** and **if** there is **enough** **space** for it
* Method **remove(String name)** - removes a fish from the pool with the given **name**, if such **exists** and returns **boolean** if the deletion is successful
* Method **findFish(String name)** - returns a **fish** with the given name, **if** doesn't exists return null
* Method **report()** - returns information about the aquarium and the fish inside it in the following format:

**"Aquarium: {name} ^ Size: {size}**

**{Fish1}**

**{Fish2}**

**… "**

## Constraints

* The name of each fish in the pool will always be unique
* Each fish will have different number of fins
* The fins of a fish and the size of the aquarium will always be positive numbers
* You will always be given fish added before receiving method for its manipulation

### Examples

|  |
| --- |
| Sample code usage |
| *// Sample Code Usage:  // Initialize Aquarium* Aquarium aquarium = **new** Aquarium(**"Ocean"**, 5, 15);  *// Initialize Fish* Fish fish = **new** Fish(**"Goldy"**, **"gold"**, 4);  *// Print Fish* System.***out***.println(fish.toString());  *//Fish: Goldy //Color: gold //Number of fins: 4  // Add Fish* aquarium.add(fish);  *// Remove Fish* System.***out***.println(aquarium.remove(**"Goldy"**)); *// true* Fish secondFish = **new** Fish(**"Dory"**, **"blue"**, 2); Fish thirdFish = **new** Fish(**"Nemo"**, **"orange"**, 5);  *// Add fish* aquarium.add(secondFish); aquarium.add(thirdFish);  *// Print Aquarium report* System.***out***.println(aquarium.report());  *//Aquarium Info: //Aquarium: Ocean ^ Size: 15 //Fish: Dory //Color: blue //Number of fins: 2 //Fish: Nemo //Color: orange //Number of fins: 5* |