# 03. Surfers



*Surfers on the beach represent a vibrant blend of activity, community, and connection with nature. The beach serves as both a playground and a social hub, where surfers prepare for their time in the waves, relax, and share their experiences with others.*

## Preparation

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

**Pay attention to name the package surfers, 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 surfers by creating the classes described below.

### Surfer

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

* **name: String**
* **age: int**
* **experience: int**
* **ownsASurfBoard: Boolean** *(note that it must be a* ***B****oolean not* ***b****oolean)*
* **money: int**

The class **constructor** should receive the **name, age, experience, ownsASurfBoard, and money**. You need to create the appropriate **getters and setters**. All Surfer names will be **unique.** It is guaranteed that there **will be no duplicates** of names.

Override the **toString()** method in the following format:

* **"Surfer {surfer name} is {surfer age} years old and has {surfer experience} years surfing experience."**

### Beach

**Next**, write a class **Beach**. The **Beach** class should have those **properties**:

* **name: String**
* **surfboardsForRent: int**
* **surfers: List<Surfer>**

The class **constructor** should receive **name,** and **surfboardsForRent.** Also, it should initialise the **Surfers** with a new **collection** instance.Implement the following features:

* **Method addSurfer(Surfer surfer)** – **adds** an **entity** to the **collection** of surfers if **the surfer has his surfboard** or he has **enough** money to **rent** it.If the surfer **does** not have his **surfboard**, the beach offers the possibility to rent a surfboard for **50 units of money**. If a surfboard is rented, the total number of surfboards for rent on the given beach decreases by one.

If the surfer does not have enough money to rent a surfboard return the following **String**:

* **"{surfer name} has not enough money to rent a surfboard."**

If the surfboards on this beach have run out return the following **String**:

* **"There are no free surfboards."**

If the surfer is added successfully to the beach return the following **String**:

* **"Surfer {surfer name} added."**
* **Method removeSurfer(String name)** – removes a **Surfer** by **given name,** if such **exists**, and **returns boolean** (**true** if it is removed, otherwise – **false**)
* **Method getMostExperiencedSurfer()**– **returns String** - the **most experienced Surfer** by **age of experience** in the given beach return:
* **"{surfer name} is most experienced surfer with {surfer experience} years experience."**

If there are no surfers on the beach, return:

* **"There are no surfers."**
* **Method getSurfer(String name)** – **returns** the **Surfer** with the **given name,** otherwise – returns **null**
* **Method getCount()** – **returns** the **count** of **Surfers** in the given beach
* **Method getSurfersWithPersonalSurfboards()** **–** **returns** **String** **–** returns the list of **surfers** in the given beach who have their surfboards in the following format:
* **"Surfers who have their own surfboards: {surfer name1}, {surfer name2}, …{surfer namen}"**

If there are no surfers on the beach, return:

* **"There are no surfers."**
* **Method getReport()** – **returns** a **String** in the following **format** (print the Surfers in **order of addition**):
  + **"Beach {beach name} was visited by the following surfers:  
    1. {surfer name} with {surfer experience} years experience.  
    2. {surfer name} with {surfer experience} years experience.  
    (…)**

1. **{surfer name} with {surfer experience} years experience."**

If the surfer has **0** years of experience return:

**n. {surfer name} with no experience.**

If there are **no** Surfers on some beaches print **only** this text:

* **"There are no surfers on {beach name} beach."**

## Constraints

* The **name** of the **Surfer** will always be **unique**.
* You will always have a Surfer added before receiving methods manipulating the Beach's surfers.

## Examples

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

|  |
| --- |
| **Sample code usage** |
| *//Initialize the repositories (Beach)* Beach malibu = new Beach("Malibu", 3);  Beach playaLaRopa = new Beach("Playa La Ropa", 2);  Beach veleka = new Beach("Veleka", 0); *//Initialize entities (Surfer)* Surfer john = new Surfer("John", 40, 10, true, 100);  Surfer mike = new Surfer("Mike", 20, 1, false, 59);  Surfer charlie = new Surfer("Charlie", 55, 19, true, 50);  Surfer hulk = new Surfer("Hulk", 18, 0, false, 49);  Surfer asen = new Surfer("Asen", 28, 10, false, 500);   System.*out*.println(malibu.addSurfer(mike));  System.*out*.println(malibu.addSurfer(john));  System.*out*.println(malibu.getSurfersWithPersonalSurfboards());  System.*out*.println(veleka.addSurfer(charlie));  System.*out*.println(veleka.addSurfer(asen));  System.*out*.println(playaLaRopa.addSurfer(hulk));  System.*out*.println(veleka.getMostExperiencedSurfer());  System.*out*.println(malibu.getReport());  System.*out*.println(playaLaRopa.getReport());  System.*out*.println(veleka.getReport());  //Surfer Mike added.  //Surfer John added.  //Surfers who have their own surfboards: John  //Surfer Charlie added.  //There are no free surfboards.  //Hulk has not enough money to rent a surfboard.  //Charlie is most experienced surfer with 19 years experience.  //Beach Malibu was visited by the following surfers:  //1. Mike with 1 years experience.  //2. John with 10 years experience.  //There are no surfers on Playa La Ropa beach.  //Beach Veleka was visited by the following surfers:  //1. Charlie with 19 years experience. |

## Submission

Submit **single .zip file**, containing **surfers** package, **with the classes inside** (**Beach**, **Surfer** 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 your program there. However, there should be **main(String[] args)** method inside.