# Space Station Recruitment



*Now that Stephen successfully established his own Space Station, he has to recruit some astronauts to work there. You are going to help him by building a system for that.*

## Preparation

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

**Pay attention to name the package (spaceStationRecruitment), 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 a repository which stores departments by creating the classes described below.

### Astronaut

First, write a Java class Astronaut with the following fields:

* **name: String**
* **age: int**
* **country: String**

The class **constructor** should receive (**name, age and country**).

The class also should have the methods:

* Getter getName()
* Getter getAge()
* Override the **toString()** method in the following format:

**"Astronaut: {name}, {age} ({country})"**

### SpaceStation

**Next**, write a **Java** class SpaceStation that has **data** (a collection which stores the entity **Astronaut**). All entities inside the repository have the **same fields**. Also, the SpaceStation class should have those fields:

* **name: String**
* **capacity: int**

The class **constructor** should receive (**name**, **capacity**), also it should initialize the **data** with a new instance of the collection**.**

Implement the following features:

* Field data – **collection** that holds added astronauts
* Getter getName
* Getter getCapacity
* Getter getCount – **returns** the **number** of astronauts
* Method add(Astronaut astronaut) – **adds** an **entity** to the data **if** **there** **is** **room** for him/her
* Method remove(String name) – removes an astronaut by **given name,** if such **exists**,   
  and **returns boolean**
* Method getOldestAstronaut() – returns the **oldest** astronaut
* Method getAstronaut(String name) – returns the astronaut with the **given name**
* report() – **returns** a **string** in the following **format** (print the astronauts in order of appearance):
  + **"Astronauts working at Space Station {spaceStationName}:  
    {Astronaut1}  
    {Astronaut2}  
    (…)**"

## Constraints

* The **names** of the astronauts will be **always unique**.
* The **age** of the astronauts will always be with **positive values**.
* You will always have an astronaut added before receiving methods manipulating the Space Station’s astronauts.

## Examples

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

|  |
| --- |
| Sample code usage |
| *// Initialize the repository* SpaceStation spaceStation = **new** SpaceStation(**"Apolo"**, 10);  *// Initialize entity* Astronaut astronaut = **new** Astronaut(**"Stephen"**, 40, **"Bulgaria"**);  *// Print Astronaut* System.***out***.println(astronaut); *// Astronaut: Stephen, 40 (Bulgaria)*  *// Add Astronaut* spaceStation.add(astronaut);  *// Remove Astronaut* spaceStation.remove(**"Astronaut name"**); *// false*  Astronaut secondAstronaut = **new** Astronaut(**"Mark"**, 34, **"UK"**); *// Add Astronaut* spaceStation.add(secondAstronaut);  Astronaut oldestAstronaut = spaceStation.getOldestAstronaut();  *// Astronaut with name Stephen*  Astronaut astronautStephen = spaceStation.getAstronaut(**"Stephen"**);  *// Astronaut with name Stephen*  *// Print Astronauts* System.***out***.println(oldestAstronaut); *// Astronaut: Stephen, 40 (Bulgaria)* System.***out***.println(astronautStephen); *// Astronaut: Stephen, 40 (Bulgaria)*  System.***out***.println(spaceStation.getCount()); *// 2* System.***out***.println(spaceStation.report());  *// Astronauts working at Space Station Apolo: // Astronaut: Stephen, 40 (Bulgaria) // Astronaut: Mark, 34 (UK)* |

## Submission

Submit **single .zip file**, containing **spaceStationRecruitment package, with the classes inside (Astronaut, SpaceStation 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.