# Guild

## Preparation

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

**Pay attention to the name of the package guild, 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 players by creating the classes described below.

**Player**

First, write a Java class **Player** with the following fields:

* **name: String**
* **clazz: String**
* **rank: String – "Trial" by default**
* **description: String – "n/a" by default**

The class **constructor** should receive the **name and clazz**. You need to create the appropriate **getters and setters**. Override the **toString()** method in the following format:

**"Player {name}: {clazz}**

**Rank: {rank}**

**Description: {description}"**

**Guild**

**Next**, write a Java class **Guild** that has **a roster** (a collection that stores **Player** entities). All entities inside the repository have the **same fields**. Also, the **Guild** class should have those **fields**:

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

The class **constructor** should receive the **name** and **capacity**, also it should initialize the **roster** with a new instance of the collection.Implement the following features:

* Method addPlayer(Player player) - **adds** an **entity** to the roster **if** **there** **is** **room** for it
* Method removePlayer(String name) - removes a player by **given name,** if such **exists**, and **returns boolean**
* Method promotePlayer(String name) - **promote** (**set his rank to "Member"**) the **first player** with the **given name.** If the player is **already** a "Member", **do nothing.**
* Method **demotePlayer(String name)- demote (set his rank to "Trial")** the first player with the **given** name. If the player is **already** a "Trial",  **do nothing.**
* Method kickPlayersByClass(String clazz) - removes all the players by the given class and returns **all removed players** from that **class as an array**
* Method count() - **returns** the **number** of players
* Method **report()** - **returns** a **String** in the following **format:**
  + **"Players in the guild: {guildName}:  
    {Player1}  
    {Player2}  
    (…)"**

## Constraints

* The **names** of the players will be **always unique**.
* You will always have a player added before receiving methods manipulating the Guild's players.

## Examples

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

|  |
| --- |
| Sample code usage |
| package guild;  public class Main {  public static void main(String[] args) {  //Initialize the repository (guild)  Guild guild = new Guild("Weekend Raiders", 20);  //Initialize entity  Player player = new Player("Mark", "Rogue");  //Print player  System.*out*.println(player);  //Player Mark: Rogue  //Rank: Trial  //Description: n/a   //Add player  guild.addPlayer(player);  System.*out*.println(guild.count()); //1  System.*out*.println(guild.removePlayer("Gosho")); //false   Player firstPlayer = new Player("Pep", "Warrior");  Player secondPlayer = new Player("Lizzy", "Priest");  Player thirdPlayer = new Player("Mike", "Rogue");  Player fourthPlayer = new Player("Marlin", "Mage");   //Add description to player  secondPlayer.setDescription("Best healer EU");   //Add players  guild.addPlayer(firstPlayer);  guild.addPlayer(secondPlayer);  guild.addPlayer(thirdPlayer);  guild.addPlayer(fourthPlayer);   //Promote player  guild.promotePlayer("Lizzy");   //Remove Player  System.*out*.println(guild.removePlayer("Pep")); //true   Player[] kickedPlayers = guild.kickPlayersByClass("Rogue");  for (Player kickedPlayer : kickedPlayers) {  System.*out*.print(kickedPlayer.getName() + " ");  }  //Mark Mike   System.*out*.println(guild.report());  //Players in the guild: Weekend Raiders:  //Player Lizzy: Priest  //Rank: Member  //Description: Best healer EU  //Player Marlin: Mage  //Rank: Trial  //Description: n/a  } } |