# 03. SoftUni Kindergarten



*You are all familiar with the problem of finding an available place in a nursery school for children. SoftUni has decided to help its employees and to build for their children a brand new, out-of-this-world, smartest possible building, fully autonomous and energy independent, and last but not least, forever cost-free SoftUni Kindergarten. You will participate in implementing some logic in the operating software.*

## Preparation

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

**Pay attention to name the package kindergarten, 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 data for every child by creating the classes described below.

### Child

You are given a class **Child,** create the following properties:

* **firstName: String**
* **lastName: String**
* **age: int**
* **parentName: String**
* **contactNumber: String**

The class **constructor** should receive **firstName, lastName, age, parentName** and **contactNumber**. You need to create the appropriate **getters and setters**. The class should override the **toString()** method in the following format:

**"Child: {firstName} {lastName}, Age: {age}, Contact info: {parentName} - {contactNumber}"**

### Kindergarten

**Next**, you are given a class **Kindergarten.** The **Kindergarten** will have **name** and **capacity** (the maximum number of children that can be registered), and **adding new children will be limited** by the capacity. **Kindergarten** also has **a registry** (a collection that stores **Child** entities). All entities inside the collection have the **same fields**. The **Kindergarten** class should have the following **properties**:

* **name: String**
* **capacity: int**
* **registry: List<Child>**

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

Implement the following features:

* Method addChild(Child child) - **Adds** a **child** to the **registry** **if** **there** **is** **room** for it and returns **true**. If there is no room for another child, returns **false.**
* Method removeChild(String firstName) - removes a child by a **given firstName**. If removal is successful, returns **true**, otherwise, returns **false**.
* Getter getChildrenCount - **Returns** the **number** of all children registered.
* Method **getChild(String firstName)** - **Returns** the **child** with the **given first name**.
* Method **registryReport()** – Orders the children by age ascending, then by last name ascending, then by first name ascending, and **returns** a **String** in the following **format:**
  + **"Registered children in {kindergartenName}:**

**--  
{child1}**

**--  
{child2}**

**--  
(…)**

**--**

**{childn}**"

### Constraints

* The combination of first and last names will **always be unique**.
* The **capacity** of the Kindergarten will always be with **positive value**.
* The **age** of the children will always be in the range **[2…6]**.

### Examples

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

|  |
| --- |
| Sample code usage |
| Kindergarten kindergarten = **new** Kindergarten(**"Sunshine"**, **8**);  Child childOne = **new** Child("**Lilyana**", "**Petrova**", **3**, "**Selena**", "**0899**"); Child childTwo = **new** Child("**Elona**", "**Muskova**", **4**,"**Max**", "**0888**"); Child childThree = **new** Child("**George**", "**Bush**", **5**,"**Ivan**", "**0988**"); Child childFour = **new** Child("**Ruzha**", "**Ignatova**", **6**,"**George**", "**0789**"); Child childFive = **new** Child("**Veselina**", "**Kostadinova**", **3**,"**Nikolas**", "**0788**"); Child childSix = **new** Child("**Tom**", "**Todorov**", **2**,"**Zendaya**", "**0799**"); Child childSeven = **new** Child("**Sara**", "**Gomez**", **2**,"**Victor**", "**0998**"); Child childEight = **new** Child("**Greta**", "**Thunberg**", **3**,"**Boris**", "**0999**");  Child childNine = **new** Child("**Anna**", "**Cameron**", **4**, "**Breja**", "**0999**");  System.out.println(kindergarten.addChild(childOne)); // **true** System.out.println(kindergarten.addChild(childTwo)); // **true** System.out.println(kindergarten.addChild(childThree)); // **true** System.out.println(kindergarten.addChild(childFour)); // **true** System.out.println(kindergarten.addChild(childFive)); // **true** System.out.println(kindergarten.addChild(childSix)); // **true** System.out.println(kindergarten.addChild(childSeven)); // **true** System.out.println(kindergarten.addChild(childEight)); // **true** System.out.println(kindergarten.addChild(childNine)); // **false**  System.out.println(kindergarten.removeChild("**Ruzha**")); // **true** System.out.println(kindergarten.removeChild("**George**")); // **true** System.out.println(kindergarten.removeChild("**Elona**")); // **true** System.out.println(kindergarten.removeChild("**Ruzha**")); // **false** System.out.println(kindergarten.removeChild("**Tim**")); // **false**  System.out.println(kindergarten.getChildrenCount()); // **5**  Child getChildTom = kindergarten.getChild("**Tom**"); Child getChildGreta = kindergarten.getChild("**Greta**");  System.out.println(getChildTom); // **Child: Tom Todorov, Age: 2, Contact info: Zendaya - 0799** System.out.println(getChildGreta); // **Child: Greta Thunberg, Age: 3, Contact info: Boris - 0999**  System.*out*.println(kindergarten.registryReport()); // **Registered children in Sunshine:** // **--** // **Child: Sara Gomez, Age: 2, Contact info: Victor - 0998** // **--** // **Child: Tom Todorov, Age: 2, Contact info: Zendaya - 0799** // **--** // **Child: Greta Thunberg, Age: 3, Contact info: Boris - 0999** // **--** // **Child: Lilyana Petrova, Age: 3, Contact info: Selena - 0899** // **--** // **Child: Veselina Kostadinova, Age: 3, Contact info: Nikolas - 0788** |

## Submission

Zip all the files in the project folder except the **bin** and **obj** folders

Submit a **single .zip file**, containing **kindergarten package, with the classes inside (Child, Kindergarten, 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 a **main(String[] args)** method inside.