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.

1. Preparation

Download the skeleton provided in Judge. **Do not** change the **`StartUp`** class or its **namespace**.

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.

2. 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, contactNumber**.

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 childFullName)`** - removes a **`Child`** by a **given full name**. The `childFullName` will be a **composition of the first name and the last name of the `Child`, separated by a single space.** If removal is successful, returns **True**, otherwise, returns **False**.
- Getter **`ChildrenCount`** - **Returns** the **number** of all children registered
- Method **`GetChild(string childFullName)` – Returns** the **`Child`** with the **given full name**. . The `childFullName` will be a **composition of the first name and the last name of the child, separated by a single space. If no child with the given `childFullName` is found, return null.**
- Method **`RegistryReport()`** – Orders the children by age descending, 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("Space S", 8);

Child childOne = new Child("Greta", "Garbo", 3, "Karl Gustafsson", "0468 888 888");
Child childTwo = new Child("Elona", "Muskova", 4, "Maye Musk", "1 888 518 3752");
Child childThree = new Child("George", "Bush", 5, "George Bush Sr.", "xx xxx xxx
xxx");
Child childFour = new Child("Ruzha", "Ignatova", 6, "Veska Ignatova", "+49 30
901820");
Child childFive = new Child("Greta", "Thinberg", 3, "Allen White", "541-754-3010");
Child childSix = new Child("T", "Rex", 2, "Steven Spielberg", "63 001 09 93");
Child childSeven = new Child("S", "Rex", 2, " Steven Spielberg ", "63 001 09 93");
Child childEight = new Child("Greta", "Thunberg", 3, "Pablo Gaviria", "0888 888
888");
Child childNine = new Child("Tim", "Duncan", 6, "William Duncan", "(555) 555-1234");

Console.WriteLine(kindergarten.AddChild(childOne));
// True
Console.WriteLine(kindergarten.AddChild(childTwo));
```

```
// True
Console.WriteLine(kindergarten.AddChild(childThree));
// True
Console.WriteLine(kindergarten.AddChild(childFour));
// True
Console.WriteLine(kindergarten.AddChild(childFive));
// True
Console.WriteLine(kindergarten.AddChild(childSix));
// True
Console.WriteLine(kindergarten.AddChild(childSeven));
// True
Console.WriteLine(kindergarten.AddChild(childEight));
// True
Console.WriteLine(kindergarten.AddChild(childNine));
// False

Console.WriteLine(kindergarten.RemoveChild("Ruzha Ignatova"));
// True
Console.WriteLine(kindergarten.RemoveChild("George Bush"));
// True
Console.WriteLine(kindergarten.RemoveChild("Elona Muskova"));
// True

Console.WriteLine(kindergarten.RemoveChild("Ruzha Ignatova"));
// False
Console.WriteLine(kindergarten.RemoveChild("Tim Duncan"));
// False

Console.WriteLine(kindergarten.ChildrenCount);
// 5

Console.WriteLine(kindergarten.GetChild("S Rex"));
//Child: S Rex, Age: 2, Contact info: Steven Spielberg - 63 001 09 93

Console.WriteLine(kindergarten.RegistryReport());
//Registered children in Space S:
//Child: Greta Garbo, Age: 3, Contact info: Karl Gustafsson - 0468 888 888
//Child: Greta Thinberg, Age: 3, Contact info: Allen White - 541-754-3010
//Child: Greta Thunberg, Age: 3, Contact info: Pablo Gaviria - 0888 888 888
//Child: S Rex, Age: 2, Contact info:  Steven Spielberg  - 63 001 09 93
//Child: T Rex, Age: 2, Contact info: Steven Spielberg - 63 001 09 93
```

Submission

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