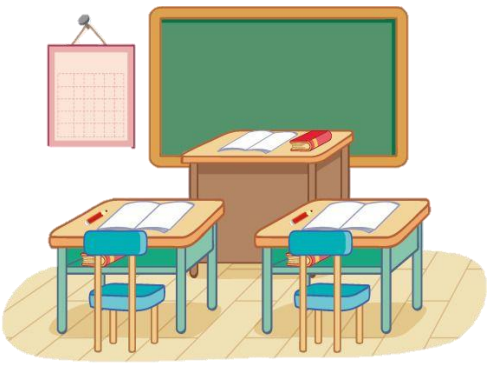
University



Preparation

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

Pay attention to name the package (university), 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.

Student

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

- **`firstName: String`**
- **`lastName: String`**
- **`bestSubject: String`**

The class **constructor** should receive (**`firstName`**, **`lastName`** and **`bestSubject`**).

The class also should have the methods:

- **`getFirstName()`**
- **`getLastName()`**
- **`getBestSubject()`**
- Override the **`toString()`** method in the following format:

```
"Student: {firstName} {lastName}, {bestSubject}"
```

University

Next, write a **Java** class **`University`** that has **students** (a collection which stores the entity **Student**). All entities inside the repository have the **same public fields**. Also, the **`University`** class should have those fields:

- **`capacity: int`**
- **`students: List<Student>`** - holds all added students in the university

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

Implement the following features:

- **`getCapacity()`**
- **`getStudents()`**
- **`getStudentCount()`** method– **returns** the **number** of **students in the university**
- **`registerStudent(Student student)`** method – **adds** an **entity** to the students **if there is room** for it
    - Returns **`"Added student {firstName} {lastName}"`** if the student is **successfully added**
    - Returns **`"Student is already in the university"`** if the student is already in the university
    - Returns **`"No seats in the university"`** if the university is full

- **`dismissStudent(Student student)`** method – **removes the student**
    - Returns **`"Student not found"`** if the student is not in the university

- **`getStudent(String firstName, String lastName)`** method - returns the student with the **given names**.
- **`getStatistics()`** – **returns** a **String** in the following **format**:
    - **`"==Student: First Name = {firstName}, Last Name = {lastName}, Best Subject = {bestSubject}`**
      `==Student: First Name = {firstName}, Last Name = {lastName}, Best Subject = {bestSubject}`
      `(…)"`

Constraints

- The **combinations** of **names** will **always be unique**.
- The **capacity** will always be **a positive number**.

Examples

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

Sample code usage
// Initialize the repository University university = new University(10); *// Initialize entities* Student student = new Student("John", "Smith", "Astrology"); Student studentTwo = new Student("Anna", "Cameron", "Geometry"); Student studentThree = new Student("Samy", "Johnson", "Algebra"); Student studentFour = new Student("Rihanna", "Fenty", "Music"); Student studentFive = new Student("Ellie", "Goulding", "Music"); *// Print Student* System.*out*.println(student); *// Student: John Smith, Astrology* *// Register Student* String register = university.registerStudent(student);

```
        System.out.println(university.getCapacity()); // 10
        System.out.println(register); // Added student John Smith
        String registerTwo = university.registerStudent(studentTwo);
        String registerThree = university.registerStudent(studentThree);
        String registerFour = university.registerStudent(studentFour);
// Dismiss Student
        String dismissed = university.dismissStudent(student);
        System.out.println(dismissed); // Removed student John Smith
        String dismissedTwo = university.dismissStudent(studentFive);
        System.out.println(dismissedTwo); // Student not found
// Get Student
        System.out.println(university.getStudent("Rihanna", "Fenty"));
// Student: Rihanna Fenty, Music
        System.out.println(university.getStudentCount()); // 3
        System.out.println(university.getStatistics());
//==Student: First Name = Anna, Last Name = Cameron, Best Subject = Geometry
 //==Student: First Name = Samy, Last Name = Johnson, Best Subject = Algebra
 //==Student: First Name = Rihanna, Last Name = Fenty, Best Subject = Music
```

Submission

Submit **single .zip file**, containing **university package, with the classes inside (Student, University 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.