# Student Management System

**A Concept for the Midterm Exam**

Aleksandre Maghlakelidze
Student at Ilia State University
alex.maglakelidze@gmail.com, aleksandre.maghlakelidze.1@iliauni.edu.ge

## Description

Create a Student Management System (SMS) in Java. An SMS is a software application used by educational institutions to manage student data and records. Our example is a basic one, which has the following features:

1. Storage for student records
2. Ability to add a new student record
3. Ability to remove a student record
4. Ability to print student information to the console

## SMS Structure

We will need the following classes for the software:

1. Student – the student record itself.
2. SMS – the student management system.
3. SMSTester – the tester class to test our management system.

## Class Student

The class Student should have several fields, including name, surname, and personal number (pn). This class can be implemented in the following way:

```
package midterm.aleksandre_maghlakelidze_1.task3;

public class Student {
   private String name, surname, pn;

   public String getName() {
       return name;
   }
```

```
    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getPn() {
        return pn;
    }

    public void setPn(String pn) {
        this.pn = pn;
    }
}
```

Pay attention to the setters and getters of the fields. In general, all the fields are private (unless special requirements are stated), and access functions are implemented as setters and getters. Read about the toString() function and implement it for the Student class.

## Class SMS

The student management system should have an internal structure for storing student records. The management system should have methods for adding new student records and removing existing ones. It should have the ability to print all student records when needed. The class can be implemented in the following way:

```
package midterm.aleksandre_maghlakelidze_1.task3;

import java.util.ArrayList;
import java.util.List;

public class SMS {
    // Mapping with Student and their index in the storage
    private List<Student> storage = new ArrayList<Student>();

    // Adds a new student record to the system
    public void addStudent(Student student) {
```

```
        storage.add(student);
    }

    // Removes a student record from the system
    public boolean removeStudent(Student student) {
        boolean removed = false;
        for (int i = 0; i < storage.size(); i++) {
            Student s = storage.get(i);
            if (s.getPn().equals(student.getPn())) {
                storage.remove(i);
                removed = true;
                break;
            }
        }
        return removed;
    }

    // Prints all student records in the system
    public void printStorage() {
        if (storage.isEmpty()) {
            System.out.println("The storage is empty");
        } else {
            for (Student s : storage) {
                System.out.println(s.getName() + " " + s.getSurname()
+ ", " + s.getPn());
                System.out.println();
            }
        }
    }
}
```

Pay attention to the usage of the ArrayList class, for loops for lists, the break clause, and the String object comparison. It is a good point to understand how interfaces work. The usage of boolean variables can also be observed in this example.

## SMS Tester Class

Now let's test our management system. First, create some student records. Then create an SMS instance and add those student records to the system using the SMS. Then try to remove some of the student records.

```
package midterm.aleksandre_maghlakelidze_1.task3;
```

```
public class SMSTester {
   public static void main(String[] args) {
       Student s1 = new Student();
       s1.setName("Maka");
       s1.setSurname("Kapanadze");
       s1.setPn("12345678912");

       Student s2 = new Student();
       s2.setName("Giorgi");
       s2.setSurname("Giorgadze");
       s2.setPn("111111111111");

       SMS sms = new SMS();
       sms.addStudent(s1);
       sms.addStudent(s1);
       sms.addStudent(s2);
       sms.printStorage();
       sms.removeStudent(s1);
       sms.printStorage(); // Prints twice to demonstrate change in
students
   }
}
```

We print the state of the student management system to check if all the methods are working properly.

This Student Management System example is inspired by my experience as a student at Ilia State University in Georgia. The personal number (pn) field in the Student class is a unique identifier for each student, similar to a student ID number used in many educational institutions. The SMS class structure is similar to the LMS class from the "lms.pdf" document, but with methods tailored to manage student records instead of books. The SMSTester class demonstrates the basic functionality of adding, removing, and printing student records in the system.