Task 5

STUDENT COURSE REGISTRATION SYSTEM

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
class Student
{
 private String name;
 private String rollNumber;
 private String grade;
 public Student(String name, String rollNumber, String grade)
{
 this.name = name;
 this.rollNumber = rollNumber;
 this.grade = grade;
 }
 public String getName()
 {
 return name;
 }
 public String getRollNumber()
 {
 return rollNumber;
 }
 public String getGrade()
 {
 return grade;
 }
```

```
 @Override
 public String toString()
{
 return "Name: " + name + ", Roll Number: " + rollNumber + ", Grade: " + grade;
 }
}
class StudentManagementSystem
{
 private List<Student> students;
 public StudentManagementSystem()
{
 students = new ArrayList<>();
 loadStudents();
 }
 public void addStudent(Student student)
{
 students.add(student);
 saveStudents();
 System.out.println("Student added successfully.");
 }
 public void removeStudent(String rollNumber)
{
 students.removeIf(student -> student.getRollNumber().equals(rollNumber));
 saveStudents();
 System.out.println("Student removed successfully.");
 }
 public Student searchStudent(String rollNumber)
{
 for (Student student : students)
{
 if (student.getRollNumber().equals(rollNumber))
```

```
{
 return student;
 }
 }
 return null;
 }
 public void displayAllStudents()
{
 if (students.isEmpty())
{
 System.out.println("No students available.");
 return;
 }
 for (Student student : students)
{
 System.out.println(student);
 }
 }
 private void loadStudents()
{

 try (BufferedReader reader = new BufferedReader(new FileReader("students.txt")))
 {

 String line;
 while ((line = reader.readLine()) != null)
 {
 String[] data = line.split(",");
 if (data.length == 3)
{
 students.add(new Student(data[0], data[1], data[2]));
```

```
 }
 }
 } catch (IOException e)
{
 System.out.println("Could not load students: " + e.getMessage());
 }
 }
 private void saveStudents()
{
 try (BufferedWriter writer = new BufferedWriter(new FileWriter("students.txt")))
 {
 for (Student student : students)
{
 writer.write(student.getName() + "," + student.getRollNumber() + "," + student.getGrade());
 writer.newLine();
 }
 } catch (IOException e)
{
 System.out.println("Could not save students: " + e.getMessage());
 }
 }
}
public class Main
{
 public static void main(String[] args)
{
 Scanner scanner = new Scanner(System.in);
 StudentManagementSystem sms = new StudentManagementSystem();
 int choice;

 do {
```

```
System.out.println("\nStudent Management System Menu:");
System.out.println("1. Add Student");
System.out.println("2. Remove Student");
System.out.println("3. Search for Student");
System.out.println("4. Display All Students");
System.out.println("5. Exit");
System.out.print("Please choose an option: ");
choice = scanner.nextInt();
scanner.nextLine(); // Consume newline
switch (choice)
{
case 1:
System.out.print("Enter name: ");
String name = scanner.nextLine();
System.out.print("Enter roll number: ");
String rollNumber = scanner.nextLine();
System.out.print("Enter grade: ");
String grade = scanner.nextLine();
if (!name.isEmpty() && !rollNumber.isEmpty() && !grade.isEmpty())
{
sms.addStudent(new Student(name, rollNumber, grade));
} else
{
System.out.println("All fields are required.");
}
break;
case 2:
System.out.print("Enter roll number to remove: ");
String rollToRemove = scanner.nextLine();
sms.removeStudent(rollToRemove);
```

```
 break;

 case 3:

 System.out.print("Enter roll number to search: ");
 String rollToSearch = scanner.nextLine();
 Student foundStudent = sms.searchStudent(rollToSearch);
 if (foundStudent != null)
{
 System.out.println("Student found: " + foundStudent);
 } else
 {
 System.out.println("Student not found.");
 }
 break;
 case 4:
 sms.displayAllStudents();
 break;
 case 5:
 System.out.println("Exiting the application. Goodbye!");
 break;
 default:
 System.out.println("Invalid option. Please try again.");
 }
 } while (choice != 5);
 scanner.close();
 }
}
```