Alexandria University
Faculty of Engineering
Computer and Communication Engineering
Specialized Scientific Programs

CC272
Programming II
Fall 2025/2026

Lab 5

Student Management System (GUI + OOP)

1. **Introduction**
   In this lab, you will design and implement a **Student Management System (SMS)** using **Java Swing** for the graphical user interface (GUI) and **Object-Oriented Programming (OOP)** principles for the backend logic.
   The system aims to help educational institutions manage student records efficiently, providing options to add, view, update, and delete student information.

   This project integrates the main concepts of **Encapsulation, Inheritance, Polymorphism, and Abstraction**, while applying GUI design using Java Swing components.

2. **Objectives**
   By the end of this lab, you should be able to:
   - Design and develop a **Java Swing GUI application**.
   - Apply **OOP concepts** to build a modular and reusable system.
   - Implement basic **CRUD operations** (Create, Read, Update, Delete).
   - Manage data validation and error handling.
   - Use **events and listeners** in Swing to handle user actions.
   - Structure their project with clear separation between GUI and backend logic.

Dr. Layla Abou-Hadeed
Eng. Ahmed ElSayed
Eng. Miar Mamdouh
Eng. Mazen Sallam
Eng. Abdelrahman Wael
Eng. Muhannad Bashar
Eng. Ahmed Ashraf
Eng. AbdElaziz Mohamed
Eng. Shams Zayan
Eng. Mohamed Zaytoon

3. **System Overview**
   The **Student Management System** allows an admin or user to manage student records through an interactive GUI interface.
   The system should include features to:
   - Add new students.
   - Display all registered students.
   - Update student details.
   - Delete a student record.
   - Search for a student by ID or name.

   The data will be handled in files (Save and load student data to and from a file).
   A simple list or collection can be used to store student objects during the runtime of the program.
   Using Files is mandatory to Save and load student data.

4. **Functional Requirements**
   4.1 Student Module
   Each student record should contain:
   - **Student ID** (unique integer, automatically generated or entered by the user)
   - **Full Name**
   - **Age**
   - **Gender** (Male/Female)
   - **Department**
   - **GPA or Grade**

   4.2 Core Functionalities
   1. **Add Student**
      - Allow the user to input student details through text fields.
      - Validate input data (e.g., no empty fields, valid age, numeric GPA).
      - Display a confirmation message after successful addition.

2. **View Students**
    - Display all students on a table or list.
    - Allow sorting by name or ID (optional enhancement).
3. **Update Student**
    - Select a student from the table.
    - Load details into the form.
    - Edit and save changes.
4. **Delete Student**
    - Allow deletion of a student by selecting them from the table.
    - Show a confirmation dialog before deletion.
5. **Search Student**
    - Provide a search box to find students by name or ID.
    - Display matching results on the table.

5. GUI Requirements

5.1 Interface Layout

You are required to design a **main window** with navigation options and separate **panels or tabs** for each functionality:

- Add a login screen (Username and Password).
- **Home / Dashboard Panel**
    - A welcome message and buttons to navigate to Add, View, Update, or Delete.
- **Add Student Panel**
    - Labels, TextFields, ComboBoxes, and Buttons.
- **View Students Panel**
    - A JTable to display student data.
- **Search and Update Panel**
    - TextField for search, results table, and editable form.
- **Delete Panel**
    - Table of students with delete button or confirmation dialog.

5.2 Components to Use

- JFrame for the main window
- JPanel for different sections
- JLabel, JTextField, JButton, JComboBox for forms
- JTable for displaying data
- JOptionPane for alerts and confirmations

6. Sample Operations Flow

1. **Open Application → Home Screen**
2. Click **Add Student → Fill Form → Save → Message: "Student Added Successfully!"**
3. Click **View Students → Table of Students Appears**
4. Click **Search → Enter ID → Display Result**
5. Click **Delete → Confirm → Student Removed**

7. Validation and Error Handling

- All text fields must be validated before processing.
- Numeric fields (like Age and GPA) must accept valid numbers only.
- Empty input should show an error message via JOptionPane.

8. Extensions (Optional)

1. Add GPA-based sorting and filtering.

Required:

1. You are required to obey the OOP concepts (inheritance, polymorphism, abstraction, …).
2. A discussion will be made with you at your lab next week on what you delivered.
3. The deadline for the delivery on the form is Sunday 26/10/2025.

What to be delivered

1. On the form, you should deliver a zipped file that contains the .java files of your classes.
2. Your zip file should be named as id1_id2_id3_id4_groupNumber. For example, 4678_4557_4558_4559_G2.

Policies:

1. You should work in groups of four (Same Groups unless you get permission to change).
2. Cheating will be severely penalized, so delivering nothing is so much better than cheating.
3. No late submission is allowed.
4. Each group must use GitHub throughout their lab work.

Dr. Layla Abou-Hadeed

Eng. Ahmed ElSayed
Eng. Miar Mamdouh
Eng. Mazen Sallam
Eng. Abdelrahman Wael
Eng. Muhannad Bashar

Eng. Ahmed Ashraf
Eng. AbdElaziz Mohamed
Eng. Shams Zayan
Eng. Mohamed Zaytoon