

**DBMS LAB ASSIGNMENT - 0**  
**ASSIGNMENT ON USER INTERFACE DEVELOPMENT AND EVENT DRIVEN PROGRAMMING**

INTRODUCTION: This is an assignment on GUI design with event handling. I have done this assignment using the Java Swing Framework. So far in this assignment no actual database is used. It was interesting to complete this assignment.

Here are the two problems given in this assignment.

---

**QUESTION - 1.** *Develop an application as follows.*

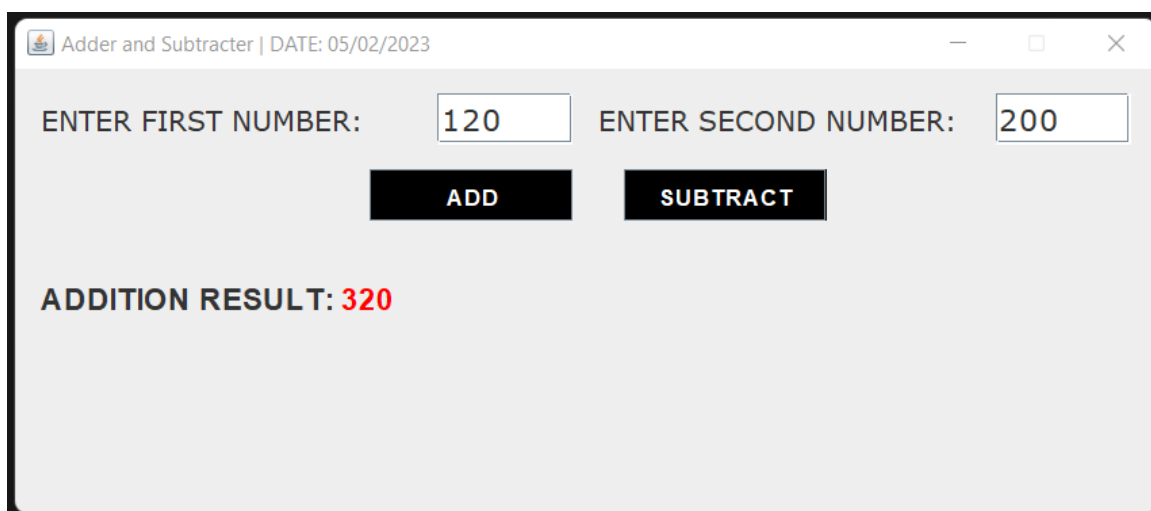
*The user will enter two numbers. Provide button to add or to find the difference. Depending on the option result will be shown in result box. Add a suitable title to the screen. Whenever the form is loaded, also display current date in the title bar.*

**SOLUTION:**

According to the question, a title is given to the Form. Also, whenever the form is loaded the current data is shown in the title bar using Java's `LocalDateTime` class.

**COMPONENTS USED:** `TextField`, `Label`, `Button`.

**EVENT HANDLING:** Whenever the ADD button is clicked an event is fired, as an `ActionListener` is attached to this button. This action first checks whether the inputs are in Integer format. If they are in Integer format the result is shown, if they are not, error is shown. Similarly, the SUBTRACT button is also attached to an `ActionListener` which verifies whether the inputs are Integers and shows corresponding result or error message.



Adder and Subtractor | DATE: 05/02/2023

ENTER FIRST NUMBER:  ENTER SECOND NUMBER:

**SUBTRACTION RESULT: -120**

Adder and Subtractor | DATE: 05/02/2023

ENTER FIRST NUMBER:  ENTER SECOND NUMBER:

**INVALID INPUT..**

### CODE FOR EVENT HANDLING:

```
addbtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try{
            int result = Integer.parseInt(num1field.getText()) +
Integer.parseInt(num2field.getText());
            resultfield.setText("<html><h2>ADDITION RESULT: <font
color='red'>" + result + "</font></h2></html>");
        }catch (NumberFormatException n){
            resultfield.setText("<html><h2>INVALID INPUT..</h2></html>");
        }
    }
});

subbtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try{
            int result = Integer.parseInt(num1field.getText()) -
Integer.parseInt(num2field.getText());
            resultfield.setText("<html><h2>SUBTRACTION RESULT: <font
color='red'>" + result + "</font></h2></html>");
        }catch (NumberFormatException n){
            resultfield.setText("<html><h2>INVALID INPUT..</h2></html>");
        }
    }
});
```

```
}  
});
```

---

**QUESTION - 2.** Consider list of departments (dept code and name), list of students (roll, dept code, name, address and phone) preloaded in array. Now develop an application for the following.

User may add/search/edit/delete/display all student record. While adding, ensure roll must be unique, a list of dept name to be shown from which user selects one and corresponding dept code to be stored. On collecting the data user may choose CANCEL/SAVE button to decide course of action. For searching user provides roll. If it exists details are shown else suitable message to be displayed. To delete user provides roll. If it does not exist then suitable message is to be displayed. To edit also user provides roll. If it exists user may be allowed to edit any field except roll. User may select CANCEL/SAVE to decide course of action. To display all records, at a time five records are to be shown. IT will also have PREV/NEXT button to display previous set and next set respectively. When first set is displayed PREV button must be disabled and at last set NEXT button must be disabled.

**SOLUTION:**

For this problem, we have a GUI with four tabs named – Add, Edit, Delete and Show. For storing Student and Department information, ArrayList is used. We have created three Java classes – Department, StudentData and Student(This one controls the GUI). Some predefined Department objects are added to the array. Also, initially, the StudentData array is empty. This array is manipulated by adding, editing, deleting students. All operations are done on some event-handlers present in the GUI.

**COMPONENTS USED:**

JLabel: for showing various text data onto the GUI.

JTabbedPane: for showing the four tabs onto to application.

JPanel: to hold multiple Components.

TextField: to take inputs from user.

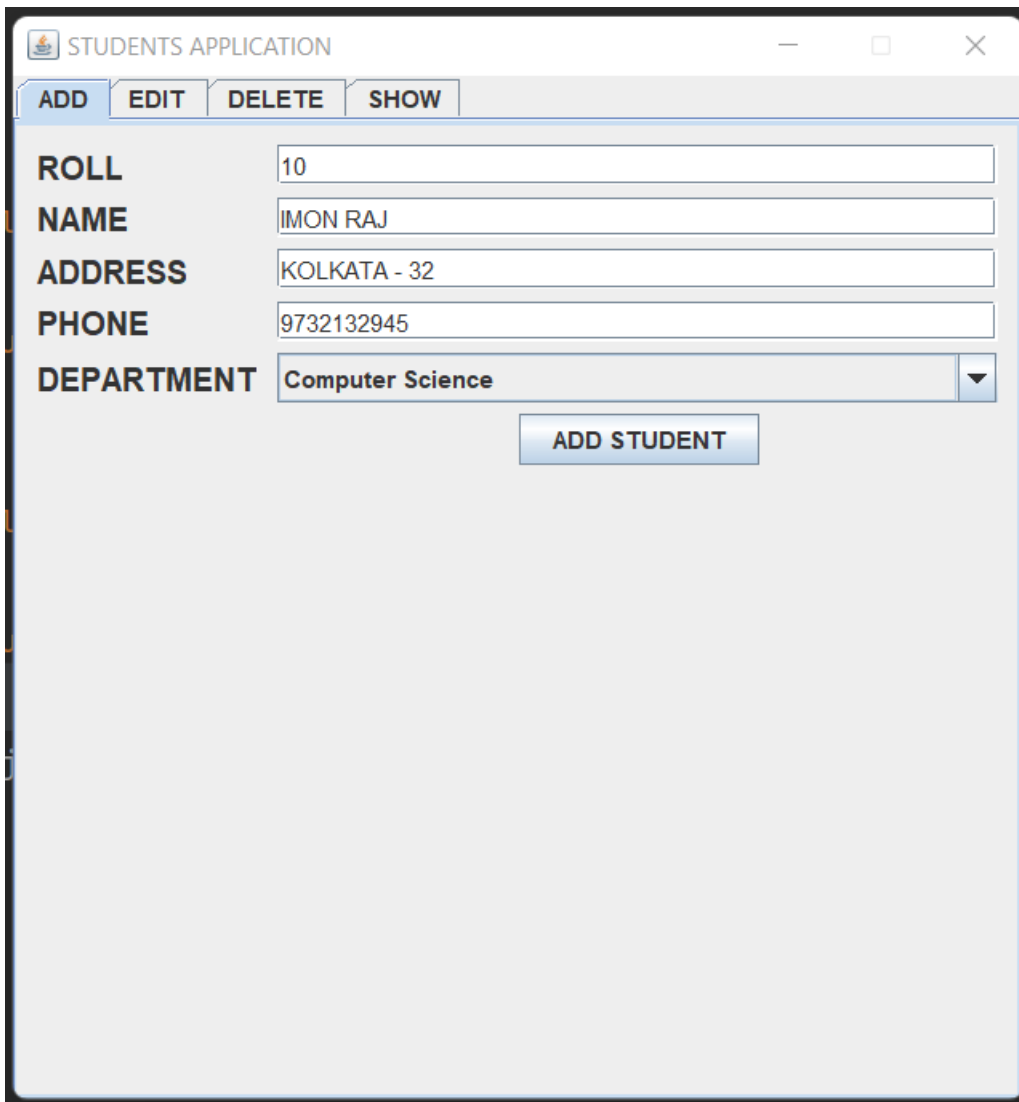
JComboBox: to show the department list as dropdown.

JScrollPane: to show scrollable output.

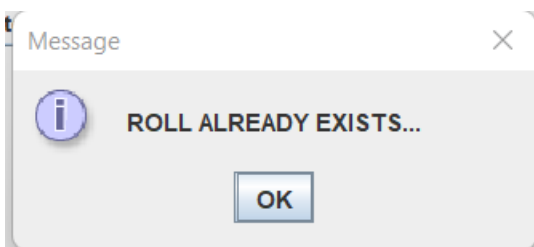
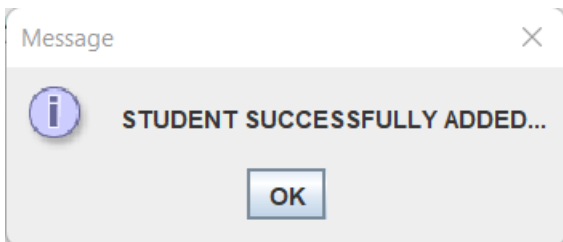
Button: to fire various events.

**EVENT HANDLING:**

**‘ADD A STUDENT’ button:** It first checks in the StudentData array, whether the roll provided already exists or not. If roll is unique, new StudentData object is created with the data from the TextField’s and the department from dropdown. That object is added to the arraylist, with success message. If roll is not unique error message is shown.



The screenshot shows a Java Swing window titled "STUDENTS APPLICATION". It has four tabs: "ADD", "EDIT", "DELETE", and "SHOW". The "ADD" tab is selected. Inside the tab, there are five labels with corresponding input fields: "ROLL" (text field with "10"), "NAME" (text field with "IMON RAJ"), "ADDRESS" (text field with "KOLKATA - 32"), "PHONE" (text field with "9732132945"), and "DEPARTMENT" (dropdown menu with "Computer Science" selected). Below these fields is a blue button labeled "ADD STUDENT".



**‘SAVE’ and ‘CANCEL’ in EDIT tab:** Firstly, whenever the roll is entered, it checks whether it is a valid roll or not. If it is valid, then data is loaded with disabling the EDIT button and ROLL field. After any edit by the user, if SAVE is clicked then, data is updated into the arraylist, if CANCEL is pressed, it ignores the changes. It then hides the fields of values and enables the ROLL field and EDIT button.

STUDENTS APPLICATION

ADD EDIT DELETE SHOW

ROLL 10 EDIT

STUDENTS APPLICATION

ADD EDIT DELETE SHOW

ROLL 10 EDIT

NAME IMON RAJ

ADDRESS KOLKATA - 32

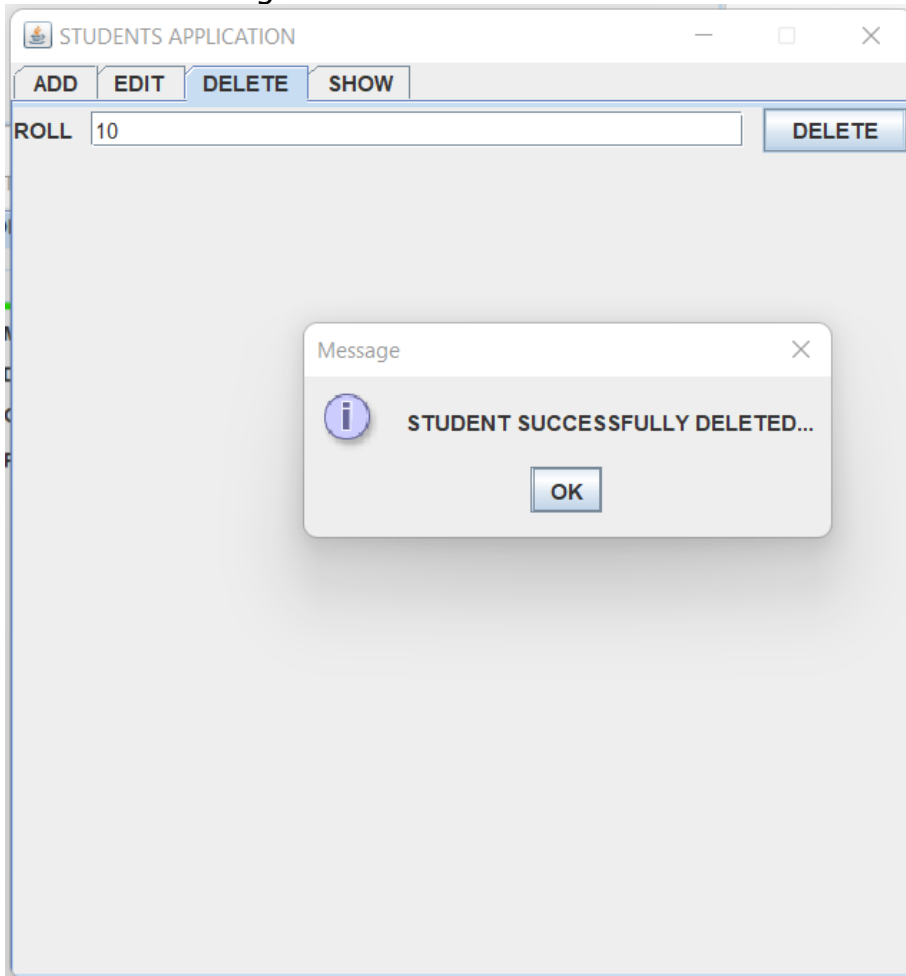
PHONE 9732132945

DEPARTMENT Computer Science

SAVE CANCEL

**‘DELETE’ button in ‘DELETE’ tab:**

Similar to the other two buttons, if first checks validity of the roll. If the roll is valid then that particular student is deleted with success message.



**The ‘SHOW’ tab:** This tab shows five students at a time. Prev and Next button is there. They will be disabled and enabled as needed. Below are some examples.

The prev and next button will fire some events that will show previous or next entries in the students array.

STUDENTS APPLICATION

ADD EDIT DELETE SHOW

NAME= 'IMON RAJ'  
DEPT= Computer Science  
ADDRESS= 'KOLKATA - 32'  
PHONE= '9999999999'  
ROLL= 10  
=====

NAME= 'AMAN RAJ'  
DEPT= Electronics  
ADDRESS= 'KOLKATA - 32'  
PHONE= '888888888888'  
ROLL= 11  
=====

NAME= 'SAHIL SEIKH'  
DEPT= Mechanical  
ADDRESS= 'KOLKATA - 32'  
PHONE= '7777777777'  
ROLL= 12  
=====

NAME= 'RAM ALI'  
DEPT= Civil  
ADDRESS= 'KOLKATA - 32'  
PHONE= '7777777777'  
ROLL= 13  
=====

NAME= 'RAHIM DAS'  
DEPT= Electronics

PREV NEXT

STUDENTS APPLICATION

ADD EDIT DELETE SHOW

NAME= 'KRISNA MONDAL'  
DEPT= Mechanical  
ADDRESS= 'KOLKATA - 32'  
PHONE= '54455434'  
ROLL= 15  
=====

NAME= 'SAGNIK MD.'  
DEPT= Electrical  
ADDRESS= 'KOLKATA - 32'  
PHONE= '6766767679'  
ROLL= 16  
=====

PREV NEXT

CODE FOR EVENT HANDLING:

```

ADDSTUDENTButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        Department d = (Department)
department.getSelectedItemAt();
        int deptCode = d.getDept_code();
//        msg += "\nDept Name: "+(d).getDept_name();
        int roll = Integer.parseInt(textField1.getText());
        String name = textField2.getText();
        String address = textField3.getText();
        String phone = textField4.getText();

        StudentData.deptlist = deptlist;
        if(rollExists(roll)){
            JOptionPane.showMessageDialog(null,"ROLL ALREADY
EXISTS...");
            return;
        }
        studlist.add(new
StudentData(name,deptCode,address,phone,roll));
        JOptionPane.showMessageDialog(null,"STUDENT SUCCESSFULLY
ADDED...");
        showAllStuds();
    }
});
EDITButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        int roll = Integer.parseInt(edit_roll_fld.getText());
        if(!rollExists(roll)){
            JOptionPane.showMessageDialog(null,"ROLL DOESN'T
EXIST...");
            return;
        }
        StudentData editstud = getStudentFromRoll(roll);
        edit_name.setText(editstud.name);
        edit_address.setText(editstud.address);
        edit_phn.setText(editstud.phone);

        depart2.setSelectedItem(getDeptFromCode(editstud.dept_code));
        EDITButton.setEnabled(false);
        edit_roll_fld.setEditable(false);

        edit_panel.setVisible(true);
    }
});
edit_SAVEButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        StudentData editstud =
getStudentFromRoll(Integer.parseInt(edit_roll_fld.getText()));
        editstud.name = edit_name.getText();
        editstud.address = edit_address.getText();
        editstud.phone = edit_phn.getText();
        editstud.dept_code = ((Department)
depart2.getSelectedItemAt()).getDept_code();

```



```

        EDITButton.setEnabled(true);
        edit_roll_fld.setEditable(true);

        edit_panel.setVisible(false);
    }
});
edit_CANCELButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        EDITButton.setEnabled(true);
        edit_roll_fld.setEditable(true);

        edit_panel.setVisible(false);
    }
});
NEXTButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        start += len;
        end += len;
        showAllStuds();
    }
});
PREVButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        start -= len;
        end -= len;
        showAllStuds();
    }
});
DELETEButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if(roll_dlt_field.getText()=="") return;
        int roll = Integer.parseInt(roll_dlt_field.getText());
        dltStudent(roll);
    }
});

```