Experiment number: 01

Experiment name: Write a JAVA Program to Display Image using JFrame.

Theory: The JFrame class is a fundamental part of Java's GUI toolkit and provides a window for organizing and displaying components, such as images. Java provides the javax.swing.JFrame class for creating and managing a window. By extending JFrame, we can create a window where we can add components like images. To display an image, we need to first load it into memory, which can be done using classes from the javax.imageio package. Once the image is loaded, we can draw it onto the JFrame using methods provided by the Graphics class.

To display an image using JFrame, the following steps are typically followed:

Loading the Image: The image to be displayed is loaded into memory using classes from the javax.imageio package. This usually involves reading the image file from the system into an Image object.

Creating the JFrame: An instance of the JFrame class is created, which serves as the window container for displaying the image.

Rendering the Image: The image loaded into memory is then rendered onto the JFrame using appropriate methods provided by the Graphics class, typically by overriding the paintComponent() method.

Displaying the JFrame: Finally, the JFrame is made visible to the user, displaying the image within its window.

Source code:

```
import java.awt.FlowLayout;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JLabel;

public class ImageDisplay extends JFrame {

    private ImageIcon iceImage;

    private JLabel iceLabel;

    private ImageIcon pustImage
    private JLabel pustLabel;
    public ImageDisplay() {
        setTitle("Image Program");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
        iceImage = new ImageIcon(getClass().getResource("ice.jpg"));
        iceLabel = new JLabel(iceImage);
        add(iceLabel);
        pustImage = new ImageIcon(getClass().getResource("pust.png"));
        pustLabel = new JLabel(pustImage);
        add(pustLabel);
        pack();
        setVisible(true);
    }
    public static void main(String[] args) {
        new ImageDisplay();
    }
}
```

<u>Output:</u>

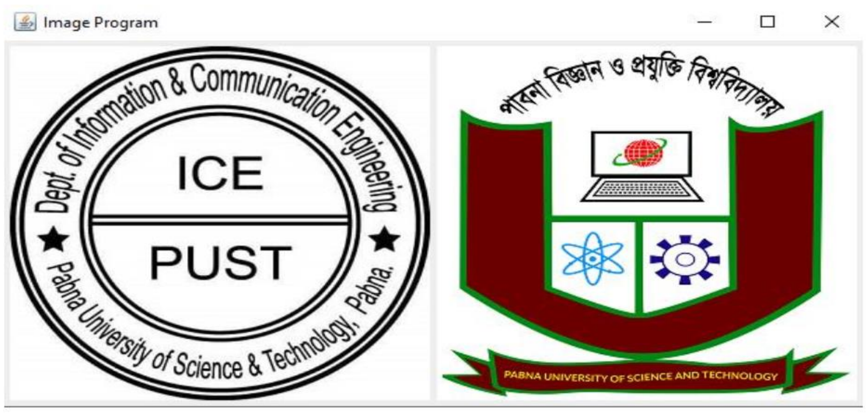


Image Program
Dept. of Information & Communication Engineering
ICE
PUST
Pabna University of Science & Technology, Pabna.
পাবনা বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয়
PABNA UNIVERSITY OF SCIENCE AND TECHNOLOGY

Experiment number: 02

Experiment name: Write a JAVA Program for generating Restaurant Bill.

Objectives:

- To write a Java program that can generate restaurant bills after ordering from a customer.
- To understand the basic concepts of Java programming, such as variables, arrays, loops, and functions.
- To be able to apply these concepts to solve real-world problems.

Theory:

A restaurant bill is a document that lists the items ordered by a customer and their corresponding prices. It also includes the total amount due, including tax and tip. The Java program that we will write will take the customer's order as input and generate a restaurant bill as output. The program will use the following concepts:

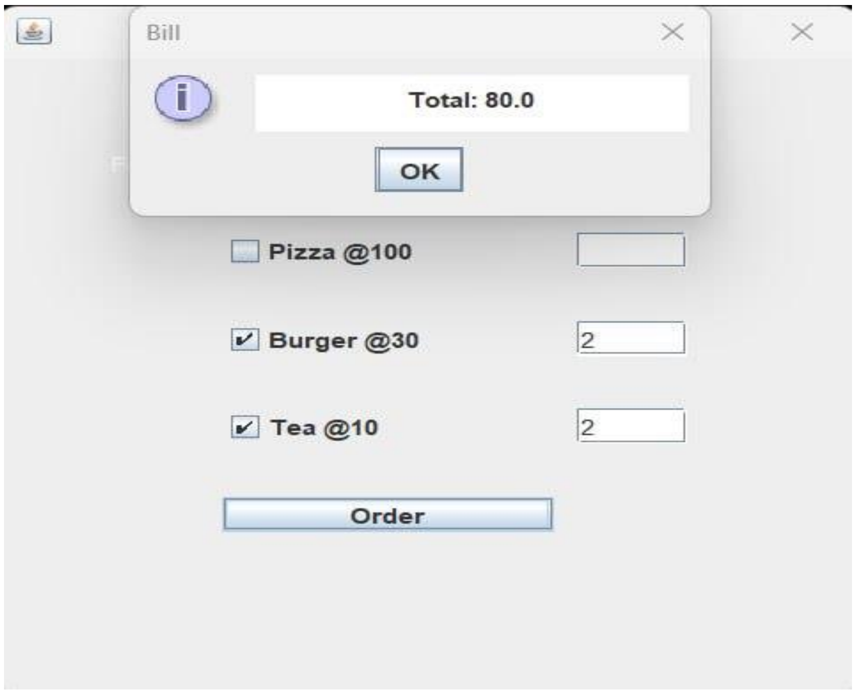
- Variables: Variables are used to store data. In our program, we will use variables to store the customer's order, the prices of the items, and the tax and tip rates.
- Arrays: Arrays are used to store a collection of data. In our program, we will use an array to store the items in the customer's order.
- Loops: Loops are used to execute a block of code repeatedly. In our program, we will use a loop to iterate through the items in the customer's order and calculate the total price.
- Functions: Functions are used to group together related code. In our program, we will define a function to calculate the total price of the customer's order.

Source code:

```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class BillGeneration1 extends JFrame implements ActionListener {
   JCheckBox cb1, cb2, cb3;
```

```
    JLabel label;
    JTextField quantityField1, quantityField2, quantityField3;
    public BillGeneration1() {
        setSize(400, 400);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(null);
        label = new JLabel("<html><font color=\"#0000FF\">Food Ordering
System</font></html>");
        label.setBounds(50, 50, 300, 20);
        cb1 = new JCheckBox("Pizza @100");
        cb1.setBounds(100, 100, 150, 20);
        cb2 = new JCheckBox("Burger @30");
        cb2.setBounds(100, 150, 150, 20);
        cb3 = new JCheckBox("Tea @10");
        cb3.setBounds(100, 200, 150, 20);
        quantityField1 = new JTextField();
        quantityField1.setBounds(260, 100, 50, 20);
        quantityField2 = new JTextField();
        quantityField2.setBounds(260, 150, 50, 20);
        quantityField3 = new JTextField();
        quantityField3.setBounds(260, 200, 50, 20);
        JButton btn = new JButton("Order");
        btn.setBounds(100, 250, 150, 20);
        btn.addActionListener(this);
        add(cb1);
        add(cb2);
        add(cb3);
        add(quantityField1);
        add(quantityField2);
        add(quantityField3);
        add(btn);
        add(label);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {
        double amount = 0;
        if (cb1.isSelected()) {
            int quantity1 = Integer.parseInt(quantityField1.getText());
            amount += 100 * quantity1;
        }
if (cb2.isSelected()) {
            int quantity2 = Integer.parseInt(quantityField2.getText());
            amount += 30 * quantity2;
        }
        if (cb3.isSelected()) {
            int quantity3 = Integer.parseInt(quantityField3.getText());
            amount += 10 * quantity3;
        }
        JPanel panel = new JPanel();
        panel.setBackground(Color.BLUE);
        JLabel messageLabel = new JLabel("Total: " + amount);
        panel.add(messageLabel);
        JOptionPane.showMessageDialog(this, panel, "Bill",
JOptionPane.INFORMATION_MESSAGE);
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new BillGeneration1());
    }
}
```

Output:



Experiment number: 03

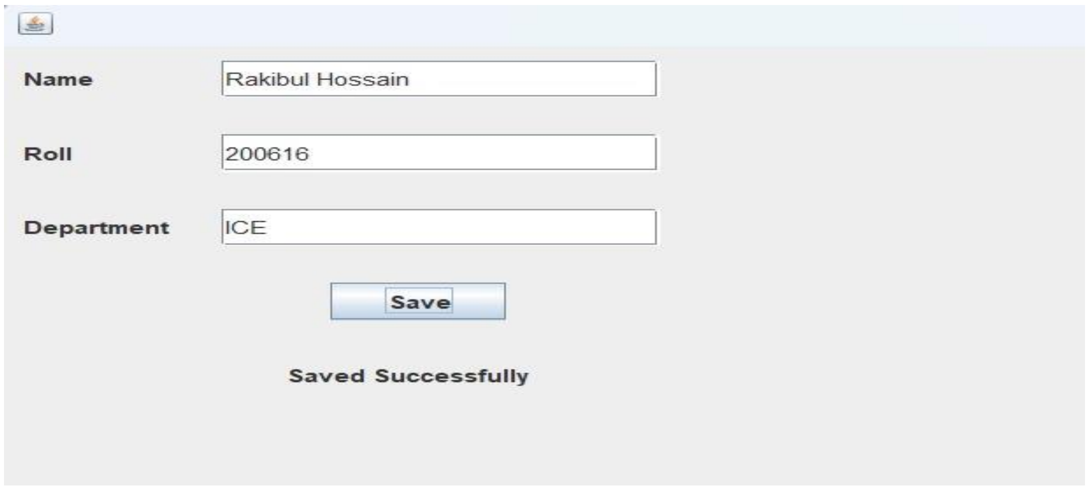
Experiment name: Write a JAVA Program to Create a Student form in GUI.

Theory: A GUI (Graphical User Interface) is a type of user interface that uses graphical elements, such as buttons, text fields, and labels, to interact with the user. GUI programming is a complex topic, but it is essential for creating user-friendly applications.
The Java Swing library provides a set of classes that can be used to create GUI applications in Java. In this experiment, we will use the Swing library to create a student registration form for the ICE department.
The form will have three text fields for the student's name, roll number, and department. It will also have a button to submit the form. When the user clicks the submit button, the program will validate the input and then print a message to the console confirming the registration.

Source code:

```
package javaapplication3;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
public class form implements ActionListener {
private static JLabel success;
private static JFrame frame;
private static JLabel label1,label2,label3;
private static JPanel panel;
private static JButton button;
private static JTextField userText1, userText2, userText3;
public static void main(String[] args) {
frame = new JFrame();
panel = new JPanel();
frame.setSize(400, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.add(panel);
panel.setLayout(null);
//Setting all Three Lebels
label1= new JLabel("Name");
label1.setBounds(10,10,80,25);
panel.add(label1);label2 = new JLabel("Roll");
label2.setBounds(10,60,80,25);
panel.add(label2);
label3 = new JLabel("Department");
label3.setBounds(10,110,80,25);
panel.add(label3);
//Creating all Textfields
userText1 = new JTextField("Enter Your Name");
userText1.setBounds(100,10,200,25);
panel.add(userText1);
JTextField userText2 = new JTextField("Enter Your Name");
userText2.setBounds(100,60,200,25);
panel.add(userText2);
JTextField userText3 = new JTextField("Enter Your Name");
userText3.setBounds(100,110,200,25);
panel.add(userText3);
button = new JButton("Save");
button.setBounds(150, 160, 80, 25);
button.addActionListener(new form());
panel.add(button);
success = new JLabel("");
success.setBounds(130,210,300,25);
panel.add(success);
frame.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent e) {
// TODO Auto-generated method stub
success.setText("Saved Successfully");
}
}
```

Output:



```
Name          Rakibul Hossain
Roll          200616
Department    ICE
              Save
        Saved Successfully
```

Experiment number: 04

Experiment name: Write a JAVA Program to develop a simple calculator in GUI

Objectives:

- To write a Java program that can create a simple calculator with a graphical user interface (GUI).
- To understand the basic concepts of Java GUI programming, such as labels, text fields, buttons, and event handlers.
- To be able to apply these concepts to create a useful and user-friendly application.

Theory:

A GUI (Graphical User Interface) is a type of user interface that uses graphical elements, such as buttons, text fields, and labels, to interact with the user. GUI programming is a complex topic, but it is essential for creating user-friendly applications.

The Java Swing library provides a set of classes that can be used to create GUI applications in Java. In this experiment, we will use the Swing library to create a simple calculator with a GUI.

The calculator will have two text fields for the first and second numbers, and it will have four buttons for addition, subtraction, multiplication, and division. When the user clicks a button, the program will perform the corresponding operation on the two numbers and display the result in the text field.

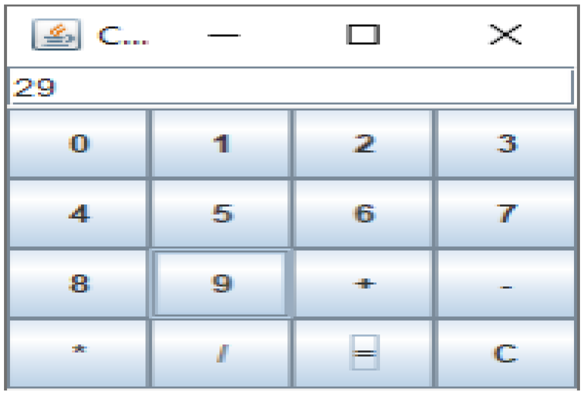
Source code:

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class Calculator extends JFrame implements ActionListener {
   JButton b10, b11, b12, b13, b14, b15;
   JButton b[] = new JButton[10];
   int i, r, n1, n2;
   JTextField res;
```

```
char op;
public Calculator() {
    super("Calculator");
    setLayout(new BorderLayout());
    JPanel p = new JPanel();
    p.setLayout(new GridLayout(4, 4));
    for (int i = 0; i <= 9; i++) {
        b[i] = new JButton(i + "");
        p.add(b[i]);
        b[i].addActionListener(this);
    }
    b10 = new JButton("+");
    p.add(b10);
    b10.addActionListener(this);
    b11 = new JButton("-");
    p.add(b11);
    b11.addActionListener(this);
    b12 = new JButton("*");
    p.add(b12);
    b12.addActionListener(this);
    b13 = new JButton("/");
    p.add(b13);
    b13.addActionListener(this);
    b14 = new JButton("=");
    p.add(b14);
    b14.addActionListener(this);
    b15 = new JButton("C");
    p.add(b15);
```

```
        b15.addActionListener(this);
        res = new JTextField(10);
        add(p, BorderLayout.CENTER);
        add(res, BorderLayout.NORTH);
        setVisible(true);
        setSize(200, 200);
    }
 public void actionPerformed(ActionEvent ae)
JButton pb = (JButton) ae.getSource();
        if (pb == b15) {
            r = n1 = n2 = 0;
            res.setText("");
        } else if (pb == b14) {
            n2 = Integer.parseInt(res.getText());
            eval();
            res.setText("" + r);
        } else {
            boolean opf = false;
            if (pb == b10) {
                op = '+';
                opf = true;
            }
            if (pb == b11) {
                op = '-';
                opf = true;
            }
            if (pb == b12) {
                op = '*';
            opf = true;
        }
        if (pb == b13) {
            op = '/';
            opf = true;
        }
        if (!opf) {
            for (i = 0; i < 10; i++) {
                if (pb == b[i]) {
                    String t = res.getText();
                    t += i;
                    res.setText(t);
                }
            }
        } else {
            n1 = Integer.parseInt(res.getText());
            res.setText("");
        }
    }
}
int eval() {
    switch (op) {
        case '+':
            r = n1 + n2;
            break;
        case '-':
            r = n1 - n2;
            break;
            case '*':
                r = n1 * n2;
                break;
            case '/':
                r = n1 / n2;
                break;
        }
        return 0;  }
    public static void main(String arg[]) {
        new Calculator();
    }
}
```

Output:


C...
29
0
1
2
3
4
5
6
7
8
9
+
-
*
/
=
C