

# **JAVA PROGRAMMING**

## **SECTION 2-1 PRACTICE**

### **JAVA BANK:**

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;

import javax.swing.border.*;

public class JavaBank extends JFrame {

    /**
     *
     */

    private static final long serialVersionUID = 1L;

    // Make these variables publicly available

    public String Name;

    public int Accountnum;

    public int Balance;

    // JPanel for user inputs

    private JPanel inputDetailJPanel;

    // JLabel and JTextField for account name

    private JLabel NameJLabel;

    private JTextField NameJTextField;

    // JLabel and JTextField for account number

    private JLabel AccountnumJLabel;

    private JTextField AccountnumJTextField;

    // JLabel and JTextField for balance
```

```

private JLabel BalanceJLabel;
private JTextField BalanceJTextField;
// JLabel and JTextField for withdraw
private JLabel DepositJLabel;
private JTextField DepositJTextField;
// JLabel and JTextField for Withdraw
private JLabel WithdrawJLabel;
private JTextField WithdrawJTextField;
// JButton to create account
private JButton CreateAccountJButton;
// JButton to delete account
private JButton DeleteAccountJButton;
// JButton to make transaction
private JButton TransactionJButton;
// JButton to display account
private JButton DisplayJButton;
// JLabel and JTextArea to display account details
private JLabel displayJLabel;
private static JTextArea displayJTextArea;
// constants
//public final static Maximum Accounts that can be created;
public final static int MaxAccounts = 10;
// one-dimensional array to store Account names as Empty or Used
static String AccountNames[] = new String[MaxAccounts];
// two-dimensional array to store Account details
static Account myAccounts[] = new Account[MaxAccounts];
static int noAccounts = 0;
// constructor
public JavaBank() {

```

```

        for (int i=0; i <10; i++) {
            AccountNames[i] = "EMPTY";
            //System.out.println(AccountNames[i]);
        }
    createUserInterface();
}

// create and position GUI components; register event handlers
private void createUserInterface() {
    // get content pane for attaching GUI components
    Container contentPane = getContentPane();

    // enable explicit positioning of GUI components
    contentPane.setLayout(null);

    // set up inputDetailJPanel
    inputDetailJPanel = new JPanel();
    inputDetailJPanel.setBounds(16, 16, 208, 250);
    inputDetailJPanel.setBorder(new TitledBorder("Input Details"));
    inputDetailJPanel.setLayout(null);
    contentPane.add(inputDetailJPanel);

    // set up NameJLabel
    NameJLabel = new JLabel();
    NameJLabel.setBounds(8, 32, 90, 23);
    NameJLabel.setText("Name:");
    inputDetailJPanel.add(NameJLabel);

    // set up NameJTextField
    NameJTextField = new JTextField();
    NameJTextField.setBounds(112, 32, 80, 21);
    NameJTextField.setHorizontalAlignment(JTextField.RIGHT);
    inputDetailJPanel.add(NameJTextField);
}

```

```
// set up AccountnumJLabel
AccountnumJLabel = new JLabel();
AccountnumJLabel.setBounds(8, 56, 100, 23);
AccountnumJLabel.setText("Account Number:");
inputDetailJPanel.add(AccountnumJLabel);

// set up AccountnumTextField
AccountnumJTextField = new JTextField();
AccountnumJTextField.setBounds(112, 56, 80, 21);
AccountnumJTextField.setHorizontalAlignment(JTextField.RIGHT);
inputDetailJPanel.add(AccountnumJTextField);

// set up BalanceJLabel
BalanceJLabel = new JLabel();
BalanceJLabel.setBounds(8, 80, 60, 23);
BalanceJLabel.setText("Balance:");
inputDetailJPanel.add(BalanceJLabel);

// set up BalanceTextField
BalanceJTextField = new JTextField();
BalanceJTextField.setBounds(112, 80, 80, 21);
BalanceJTextField.setHorizontalAlignment(JTextField.RIGHT);
inputDetailJPanel.add(BalanceJTextField);

// set up DepositJLabel
DepositJLabel = new JLabel();
DepositJLabel.setBounds(8, 104, 80, 23);
DepositJLabel.setText("Deposit:");
```

```
inputDetailJPanel.add(DepositJLabel);

// set up DepositJTextField
DepositJTextField = new JTextField();
DepositJTextField.setBounds(112, 104, 80, 21);
DepositJTextField.setHorizontalAlignment(JTextField.RIGHT);
inputDetailJPanel.add(DepositJTextField);

// set up WithdrawJLabel
WithdrawJLabel = new JLabel();
WithdrawJLabel.setBounds(8, 128, 60, 23);
WithdrawJLabel.setText("Withdraw:");
inputDetailJPanel.add(WithdrawJLabel);

// set up WithdrawJTextField
WithdrawJTextField = new JTextField();
WithdrawJTextField.setBounds(112, 128, 80, 21);
WithdrawJTextField.setHorizontalAlignment(JTextField.RIGHT);
inputDetailJPanel.add(WithdrawJTextField);

// set up CreateAccountButton
CreateAccountJButton = new JButton();
CreateAccountJButton.setBounds(112, 152, 80, 24);
CreateAccountJButton.setText("Create");
inputDetailJPanel.add(CreateAccountJButton);
CreateAccountJButton.addActionListener(

new ActionListener() {
    // event handler called when CreateAccountJButton
```

```

        // is clicked

        public void actionPerformed(ActionEvent event) {
            CreateAccountJButtonActionPerformed(event);
        }

    }

); // end call to addActionListener

// set up DeleteAccountButton
DeleteAccountJButton = new JButton();
DeleteAccountJButton.setBounds(16, 152, 80, 24);
DeleteAccountJButton.setText("Delete");
inputDetailJPanel.add(DeleteAccountJButton);
DeleteAccountJButton.addActionListener(

new ActionListener() // anonymous inner class
{
    // event handler called when DeleteAccountJButton
    // is clicked
    public void actionPerformed(ActionEvent event) {
        DeleteAccountJButtonActionPerformed(event);

    }

}

); // end call to addActionListener

```

```
// set up TransactionJButton

TransactionJButton = new JButton();

TransactionJButton.setBounds(16, 180, 176, 24);

TransactionJButton.setText("Make Transaction");

inputDetailJPanel.add(TransactionJButton);

TransactionJButton.addActionListener(

    new ActionListener() // anonymous inner class

    {

        // event handler called when TransactionJButton

        // is clicked

        public void actionPerformed(ActionEvent event) {

            TransactionJButtonActionPerformed(event);

        }

    } // end anonymous inner class

); // end call to addActionListener
```

```
// set up DisplayJButton

DisplayJButton = new JButton();

DisplayJButton.setBounds(16, 208, 176, 24);

DisplayJButton.setText("Display Accounts");

inputDetailJPanel.add(DisplayJButton);

DisplayJButton.addActionListener(

    new ActionListener() // anonymous inner class

    {

        // event handler called when TransactionJButton
```

```
// is clicked  
public void actionPerformed(ActionEvent event) {  
    DisplayJButtonActionPerformed(event);  
}
```

```
} // end anonymous inner class
```

```
); // end call to addActionListener
```

```
// set up displayJLabel
```

```
displayJLabel = new JLabel();  
displayJLabel.setBounds(240, 16, 150, 23);  
displayJLabel.setText("Account Details:");  
contentPane.add(displayJLabel);
```

```
// set up displayJTextArea
```

```
displayJTextArea = new JTextArea();  
JScrollPane scrollPane = new JScrollPane(displayJTextArea);  
scrollPane.setBounds(240, 48, 402, 184);  
scrollPane.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS);  
contentPane.add(scrollPane);  
displayJTextArea.setText("Welcome to Java Bank - There are currently no Accounts created");
```

```
// clear other JTextFields for new data
```

```
NameJTextField.setText(" ");  
AccountnumJTextField.setText("0");  
BalanceJTextField.setText("0");  
DepositJTextField.setText("0");
```



```
WithdrawJTextField.setText("0");

// set properties of application's window
setTitle("Java Bank"); // set title bar string
setSize(670, 308); // set window size
setVisible(true); // display window

} // end method createUserInterface

private void CreateAccountJButtonActionPerformed(ActionEvent event) {
    // System.out.println("Create Account Button Clicked");

    displayJTextArea.setText("");

    Name = "";

    //Get Name from Text Field
    Name = NameJTextField.getText();

    //Get Accountnum from Text Field and convert to int unless blank then set to 0
    if (AccountnumJTextField.getText() == "0") {
        Accountnum = 0;
    }
    else {
        Accountnum = Integer.parseInt(AccountnumJTextField.getText());
    }
}
```

```

//Get Balance from Text Field and convert to int unless blank then set to 0
if (BalanceJTextField.getText() == "0") {
    Balance = 0;
}
else {
    Balance = Integer.parseInt(BalanceJTextField.getText());
}

//int emptyAccount = 11;

if ((noAccounts <= 9) & (Name != "") & (Accountnum != 0)) {
    myAccounts[noAccounts] = new Account(Name,Accountnum,Balance);
    AccountNames[noAccounts] = "USED";
    //System.out.println(myAccounts[noAccounts].getaccountname());
    //emptyAccount = i;

    displayJTextArea.setText(myAccounts[noAccounts].getaccountname() + " " +
myAccounts[noAccounts].getaccountnum() + " " + myAccounts[noAccounts].getbalance());
    noAccounts ++;
    System.out.println(noAccounts);
}
else {
    displayJTextArea.setText("Both the Name field and Account Number must be
completed");
}

```

```

if (noAccounts == 10) {

    // Once account 10 is created. All accounts full.
    displayJTextArea.setText("All Accounts Full!");
}

// clear other JTextFields for new data
NameJTextField.setText(" ");
AccountnumJTextField.setText("0");
BalanceJTextField.setText("0");
DepositJTextField.setText("0");
WithdrawJTextField.setText("0");

}

private void DeleteAccountJButtonActionPerformed(ActionEvent event) {

    displayJTextArea.setText("Oops this isnt coded in this version!");
    //Name = NameJTextField.getText();
    //System.out.println("Delete Account: " + Name);

    // Enter code to delete here

    // clear JTextFields for new data

    NameJTextField.setText(" ");
    AccountnumJTextField.setText("0");
    BalanceJTextField.setText("0");
    DepositJTextField.setText("0");
    WithdrawJTextField.setText("0");

```

```

}

    private void TransactionJButtonActionPerformed(ActionEvent event) {
displayJTextArea.setText("");
        if (noAccounts == 0) {
            displayJTextArea.setText("No Accounts currently created");
        }else {
            // get user input

            int Accountnum = Integer.parseInt(AccountnumJTextField.getText());
            int Deposit = Integer.parseInt(DepositJTextField.getText());
            int Withdraw = Integer.parseInt(WithdrawJTextField.getText());
            for (int i=0; i<noAccounts; i++) {
                if ((myAccounts[i].getaccountnum() == Accountnum) && (Deposit>0)) {
                    myAccounts[i].setbalance(myAccounts[i].getbalance()+Deposit);
                    displayJTextArea.setText(myAccounts[i].getaccountname() + " " +
myAccounts[i].getaccountnum() + " " + myAccounts[i].getbalance());
                }

                if ((myAccounts[i].getaccountnum() == Accountnum) && (Withdraw>0)) {
                    myAccounts[i].setbalance(myAccounts[i].getbalance()-Withdraw);
                    displayJTextArea.setText(myAccounts[i].getaccountname() + " " +
myAccounts[i].getaccountnum() + " " + myAccounts[i].getbalance());
                }
            }
        }

        // clear other JTextFields for new data
        NameJTextField.setText(" ");
        AccountnumJTextField.setText("0");
        BalanceJTextField.setText("0");
        DepositJTextField.setText("0");
    }
}

```

```

        WithdrawJTextField.setText("0");
    }

    private void DisplayJButtonActionPerformed(ActionEvent event) {

        Name = NameJTextField.getText();
        displayJTextArea.setText("");

        if (noAccounts == 0) {
            displayJTextArea.setText("No Accounts currently created");
        }else {
            for (int i=0; i<noAccounts; i++) {

                displayJTextArea.append(myAccounts[i].getaccountname() + " " +
myAccounts[i].getaccountnum() + " " + myAccounts[i].getbalance() + "\n");

            }
        }

        // clear other JTextFields for new data
        NameJTextField.setText(" ");
        AccountnumJTextField.setText("0");
        BalanceJTextField.setText("0");
        DepositJTextField.setText("0");
        WithdrawJTextField.setText("0");
    }

    public static void main(String[] args) {
        // Populate arrays with the word EMPTY
        // so we can check to see if the values are empty later

```

```

JavaBank application = new JavaBank();

application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

}

}

```

## **BIKE:**

```
package bikeproject;
```

```
public class MountainBike extends Bike{
```

```
    private String suspension, type;
```

```
    private int frameSize;
```

```
    public MountainBike()
```

```
{
```

```
        this("Bull Horn", "Hardtail", "Maxxis", "dropper", 27, "RockShox XC32", "Pro", 19);
```

```
    }//end constructor
```

```
    public MountainBike(String handleBars, String frame, String tyres, String seatType, int numGears,
```

```
        String suspension, String type, int frameSize) {
```

```

        super(handleBars, frame, tyres, seatType, numGears);

        this.suspension = suspension;

        this.type = type;

        this.frameSize = frameSize;
    } //end constructor


    public void printDescription()
    {
        super.printDescription();

        System.out.println("This mountain bike is a " + this.type + " bike and has a " +
this.suspension + " suspension and a frame size of " + this.frameSize + "inches.");

    } //end method printDescription
} //end class MountainBike

---

package bikeproject;

public class BikeDriver {

    public static void main(String[] args) {

        RoadBike bike1 = new RoadBike();

        RoadBike bike2 = new RoadBike("drop", "tourer", "semi-grip", "comfort", 14, 25, 18);

        MountainBike bike3 = new MountainBike();

        Bike bike4 = new Bike();

        bike1.printDescription();

        bike2.printDescription();
    }
}

```

```

        bike3.printDescription();

        bike4.printDescription();

    }//end method main
} //end class BikeDriver
----

package bikeproject;

public class MountainBike extends Bike{

    private String suspension, type;
    private int frameSize;

    public MountainBike()
    {
        this("Bull Horn", "Hardtail", "Maxxis", "dropper", 27, "RockShox XC32", "Pro", 19);
    } //end constructor

    public MountainBike(String handleBars, String frame, String tyres, String seatType, int numGears,
        String suspension, String type, int frameSize) {
        super(handleBars, frame, tyres, seatType, numGears);
        this.suspension = suspension;
        this.type = type;
        this.frameSize = frameSize;
    } //end constructor

    public void printDescription()
    {
        super.printDescription();
    }

```



```
        System.out.println("This mountain bike is a " + this.type + " bike and has a " +  
this.suspension + " suspension and a frame size of " + this.frameSize + "inches.");
```

```
    }//end method printDescription
```

```
//end class MountainBike
```

```
----
```

```
package bikeproject;
```

```
public class RoadBike extends Bike{
```

```
    private int  tyreWidth, postHeight;
```

```
    public RoadBike()
```

```
    {
```

```
        this("drop", "racing", "tread less", "razor", 19, 20, 22);
```

```
    }//end constructor
```

```
    public RoadBike(int postHeight)
```

```
    {
```

```
        this("drop", "racing", "tread less", "razor", 19, 20, postHeight);
```

```
    }//end constructor
```

```
    public RoadBike(String handleBars, String frame, String tyres, String seatType, int numGears,
```

```
        int tyreWidth, int postHeight) {
```

```
        super(handleBars, frame, tyres, seatType, numGears);
```

```
        this.tyreWidth = tyreWidth;
```

```
        this.postHeight = postHeight;
```

```
    }//end constructor
```

```

        public void printDescription()
        {
            super.printDescription();

            System.out.println("This Roadbike bike has " + this.tyreWidth + "mm tyres and a post
height of " + this.postHeight + ".");

            }//end method printDescription
    }//end class RoadBike

```

---

```

Oracle Cycles
This bike has drop handlebars on a racing frame with 19 gears.
It has a razor seat with tread less tyres.
This Roadbike bike has 20mm tyres and a post height of 22.

Oracle Cycles
This bike has drop handlebars on a tourer frame with 14 gears.
It has a comfort seat with semi-grip tyres.
This Roadbike bike has 25mm tyres and a post height of 18.

Oracle Cycles
This bike has Bull Horn handlebars on a Hardtail frame with 27 gears.
It has a dropper seat with Maxxis tyres.
This mountain bike is a Pro bike and has a RockShox XC32 suspension and a frame size of 19inches.

Oracle Cycles
This bike has null handlebars on a null frame with 0 gears.
It has a null seat with null tyres.

```

## **CALCULATOR:**

```

import java.awt.Container;

import javax.swing.JFrame;

import javax.swing.JPanel;

import calculator.CalcPanel;

public class CalcMain {

    public static void main(String[] args) {

        JFrame theGUI = new JFrame();

        theGUI.setTitle("My Calculator");

        theGUI.setSize(220,350);
    }
}

```

```

        theGUI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        Container pane = theGUI.getContentPane();

        JPanel myPanel = new CalcPanel();

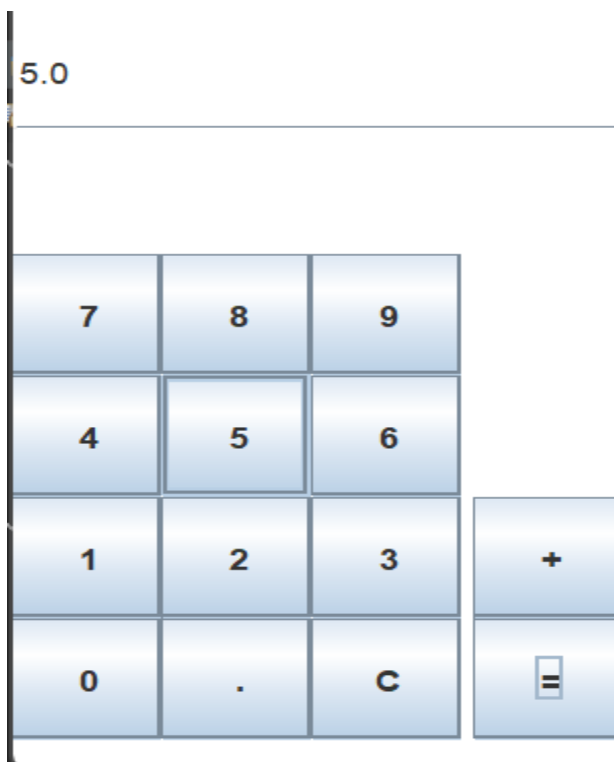
        pane.add(myPanel);

        theGUI.setVisible(true);

    }

}

```



```

package calculator;

import javax.swing.*.*;
import java.awt.Color;
import java.awt.event.*;

public class CalcPanel extends JPanel implements ActionListener {

    String num1="";
    String num2="";

```

```
String operator="";
boolean usingFirst=true;
double total=0;
JTextField display;
JButton b1;
JButton b2;
JButton b3;
JButton b4;
JButton b5;
JButton b6;
JButton b7;
JButton b8;
JButton b9;
JButton b0;
JButton bdec;
JButton bclear;
JButton bequals;
JButton bplus;
public CalcPanel()
{
    this.setBackground(Color.white);
    setLayout(null);
    display=new JTextField();

    b1=new JButton("1");
    b2=new JButton("2");
    b3=new JButton("3");
    b4=new JButton("4");
    b5=new JButton("5");
```

```
b6=new JButton("6");
b7=new JButton("7");
b8=new JButton("8");
b9=new JButton("9");
b0=new JButton("0");
bdec=new JButton(".");
bclear=new JButton("C");
bequals = new JButton( "=");
bplus=new JButton("+");
display.setBounds(0,0,205,50);
b1.setBounds(0,200,50,50);
b2.setBounds(50,200,50,50);
b3.setBounds(100,200,50,50);
bplus.setBounds(154,200,50,50);
b4.setBounds(0,150,50,50);
b5.setBounds(50,150,50,50);
b6.setBounds(100,150,50,50);
b7.setBounds(0,100,50,50);
b8.setBounds(50,100,50,50);
b9.setBounds(100,100,50,50);
b0.setBounds(0,250,50,50);
bdec.setBounds(50,250,50,50);
bclear.setBounds(100,250,50,50);
bequals.setBounds(154,250,50,50);
```

```
add(b1);
```

```
add(b2);
```

```
add(b3);
```

```
add(b4);
```

```
add(b5);  
add(b6);  
add(b7);  
add(b8);  
add(b9);  
add(b0);  
add(bdec);  
add(display);  
add(bclear);  
add(equals);  
add(bplus);
```

```
b1.addActionListener(this);  
b2.addActionListener(this);  
b3.addActionListener(this);  
b4.addActionListener(this);  
b5.addActionListener(this);  
b6.addActionListener(this);  
b7.addActionListener(this);  
b8.addActionListener(this);  
b9.addActionListener(this);  
b0.addActionListener(this);  
equals.addActionListener(this);  
bplus.addActionListener(this);  
bclear.addActionListener(this);  
bdec.addActionListener(this);
```

```
}
```

```
public void actionPerformed(ActionEvent e){
```

```

String s=e.getActionCommand();
if(s.equals("1")||s.equals("2")||s.equals("3")||s.equals("4")||
    s.equals("5")||s.equals("6")||s.equals("7")||s.equals("8")||
    s.equals("9")||s.equals("0")||s.equals("."))
{
    if(usingFirst)
    {

        num1=num1+s;
        display.setText(num1);
    }
    else
    {
        num2=num2+s;
        display.setText(num2);
    }
}
if(s.equals("+"))
{
    usingFirst=false;
    operator="+";
}
if(s.equals("="))
{
    switch(operator){
    case "+":
        total=Double.parseDouble(num1)+Double.parseDouble(num2);
        display.setText( ""+total );
        break;

```

```

        }
        usingFirst=true;
        num1="";
        num2="";
        operator="";
    }
    if(s.equals("C"))
    {
        display.setText( "" );
        usingFirst=true;
        num1="";
        num2="";
        total=0;
    }
}

```

93.0

7	8	9	
4	5	6	
1	2	3	+
0	.	C	=