Matrix Multiplication:

```
import java.util.Scanner;

public class SimpleMatrixMultiplication {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Input matrix sizes
        System.out.print("Enter rows and columns of first matrix: ");
        int r1 = sc.nextInt();
        int c1 = sc.nextInt();

        System.out.print("Enter rows and columns of second matrix: ");
        int r2 = sc.nextInt();
        int c2 = sc.nextInt();

        // Check if multiplication is possible
        if (c1 != r2) {
            System.out.println("Multiplication not possible. Columns of first matrix must
equal rows of second.");
            sc.close();
            return;
        }

        int[][] a = new int[r1][c1];
        int[][] b = new int[r2][c2];
        int[][] result = new int[r1][c2];

        // Input first matrix
        System.out.println("Enter elements of first matrix:");
```

```
for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c1; j++) {
        a[i][j] = sc.nextInt();
    }
}

// Input second matrix
System.out.println("Enter elements of second matrix:");
for (int i = 0; i < r2; i++) {
    for (int j = 0; j < c2; j++) {
        b[i][j] = sc.nextInt();
    }
}

// Multiply matrices
for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c2; j++) {
        result[i][j] = 0;
        for (int k = 0; k < c1; k++) {
            result[i][j] += a[i][k] * b[k][j];
        }
    }
}

// Display result
System.out.println("Result of multiplication:");
for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c2; j++) {
        System.out.print(result[i][j] + "\t");
    }
```

```
            System.out.println();
        }

        sc.close();
    }
}
```

Calculate Interest :

```
public class Account {
    // Private variables (access specifiers used)
    private String accountHolder;
    private double balance;
    private double annualRate; // Annual interest rate

    // Constructor
    public Account(String accountHolder, double balance, double
annualRate) {
        this.accountHolder = accountHolder;
        this.balance = balance;
        this.annualRate = annualRate;
    }

    // Method 1: Calculate simple interest quarterly
    public void calculateQuarterlyInterest() {
        double quarterlyInterest = (balance * annualRate * 0.25) / 100;  //
0.25 year = 3 months
        System.out.println("Quarterly Interest: ₹" + quarterlyInterest);
    }
```

```
    // Method 2: Display current balance
    public void displayBalance() {
        System.out.println("Account Holder: " + accountHolder);
        System.out.println("Current Balance: ₹" + balance);
    }

    // Method 3: Withdraw money
    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("₹" + amount + " withdrawn successfully.");
        } else {
            System.out.println("Insufficient balance!");
        }
    }

    // Main method for testing
    public static void main(String[] args) {
        Account acc = new Account("Pritam", 10000, 5.0);  // 5% interest
rate

        acc.displayBalance();
        acc.calculateQuarterlyInterest();
        acc.withdraw(2500);
        acc.displayBalance();
    }
}
```

Draw Circle or Triangle:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

/* <applet code="ShapeApplet" width=400 height=300>
   </applet>
*/

public class ShapeApplet extends Applet implements ActionListener {
    Button circleButton, triangleButton;
    String shapeToDraw = "";

    public void init() {
        setLayout(new FlowLayout());

        circleButton = new Button("Draw Circle");
        triangleButton = new Button("Draw Triangle");

        add(circleButton);
        add(triangleButton);

        circleButton.addActionListener(this);
        triangleButton.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
```

```
        if (e.getSource() == circleButton) {
            shapeToDraw = "circle";
        } else if (e.getSource() == triangleButton) {
            shapeToDraw = "triangle";
        }
        repaint();  // Request to redraw
    }

    public void paint(Graphics g) {
        if (shapeToDraw.equals("circle")) {
            g.setColor(Color.BLUE);
            g.drawOval(150, 100, 100, 100);  // x, y, width, height
        } else if (shapeToDraw.equals("triangle")) {
            g.setColor(Color.RED);
            int[] xPoints = {150, 200, 250};
            int[] yPoints = {200, 100, 200};
            g.drawPolygon(xPoints, yPoints, 3);
        }
    }
}
```

Multiplication table of a number:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

/* <applet code="TableApplet" width=300 height=300>
   </applet>
```

```
*/

public class TableApplet extends Applet implements ActionListener {
    TextField numberInput;
    Button showTableButton;
    TextArea resultArea;

    public void init() {
        setLayout(new FlowLayout());

        Label prompt = new Label("Enter a number:");
        numberInput = new TextField(10);
        showTableButton = new Button("Show Table");
        resultArea = new TextArea(10, 25); // For displaying the table

        add(prompt);
        add(numberInput);
        add(showTableButton);
        add(resultArea);

        showTableButton.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        try {
            int number = Integer.parseInt(numberInput.getText());
            StringBuilder table = new StringBuilder();

            for (int i = 1; i <= 10; i++) {
                table.append(number).append(" x ").append(i).append(" =
").append(number * i).append("\n");
```

```
            }

            resultArea.setText(table.toString());
        } catch (NumberFormatException ex) {
            resultArea.setText("Please enter a valid number.");
        }
    }
}
```

Reverse of a String:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

/* <applet code="ReverseNameApplet" width=300 height=200>
  </applet>
*/

public class ReverseNameApplet extends Applet implements ActionListener {
    TextField nameInput;
    Button reverseButton;
    Label resultLabel;

    public void init() {
        setLayout(new FlowLayout());

        Label nameLabel = new Label("Enter your name:");
        nameInput = new TextField(20);
        reverseButton = new Button("Reverse");
        resultLabel = new Label("");
```

```
        add(nameLabel);
        add(nameInput);
        add(reverseButton);
        add(resultLabel);

        reverseButton.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        String name = nameInput.getText();
        String reversed = new StringBuilder(name).reverse().toString();
        resultLabel.setText("Reversed: " + reversed);
    }
}
```

Account Display:

```
// Base class
class Account {
    protected String accountNumber;
    protected String accountHolderName;
    protected double balance;

    // Constructor
    public Account(String accountNumber, String accountHolderName, double
balance) {
        this.accountNumber = accountNumber;
        this.accountHolderName = accountHolderName;
        this.balance = balance;
    }
```

```
    // Method to display account details
    public void Display_Account_Detail() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Holder: " + accountHolderName);
        System.out.println("Balance: ₹" + balance);
    }

    // Method to show account balance
    public double Account_Balance() {
        return balance;
    }
}

// Derived class
class SavingAccount extends Account {
    private double interestRate;

    // Constructor for SavingAccount
    public SavingAccount(String accountNumber, String accountHolderName, double
balance, double interestRate) {
        super(accountNumber, accountHolderName, balance);
        this.interestRate = interestRate;
    }

    // Overriding the Display_Account_Detail method
    @Override
    public void Display_Account_Detail() {
        super.Display_Account_Detail(); // Call base class method
        System.out.println("Interest Rate: " + interestRate + "%");
    }
}
```

```
// Main class to test
public class Main {
    public static void main(String[] args) {
        // Creating base Account
        Account acc = new Account("ACC123", "Pritam Chowdhury", 15000);
        System.out.println("=== Account Details ===");
        acc.Display_Account_Detail();
        System.out.println("Balance: ₹" + acc.Account_Balance());

        // Creating Saving Account
        SavingAccount savAcc = new SavingAccount("SAV456", "Pritam Chowdhury",
25000, 4.5);
        System.out.println("\n=== Saving Account Details ===");
        savAcc.Display_Account_Detail();
        System.out.println("Balance: ₹" + savAcc.Account_Balance());
    }
}
```