

# Slip 1 java

## Slip 1.1

```
class PrimeArray {
    // Method to check if a number is prime
    static boolean isPrime(int num) {
        if (num <= 1)
            return false;
        for (int i = 2; i <= num / 2; i++) {
            if (num % i == 0)
                return false;
        }
        return true;
    }

    public static void main(String args[]) {
        if (args.length == 0) {
            System.out.println("Please provide numbers as command line arguments.");
            return;
        }

        System.out.println("Prime numbers in the array:");
        for (int i = 0; i < args.length; i++) {
            int num = Integer.parseInt(args[i]); // Convert string to integer
            if (isPrime(num)) {
                System.out.print(num + " ");
            }
        }
    }
}
```

java PrimeArray 10 11 12 13 14 15 17

## Slip 1.2

```
import java.util.Scanner;

// Abstract class Staff
abstract class Staff {
    protected int id;
```

```

protected String name;

// Parameterized constructor
Staff(int id, String name) {
    this.id = id;
    this.name = name;
}

// Abstract method to display details
abstract void display();
}

// Subclass OfficeStaff
class OfficeStaff extends Staff {
    private String department;

    // Parameterized constructor
    OfficeStaff(int id, String name, String department) {
        super(id, name); // Call parent constructor
        this.department = department;
    }

    // Implement display method
    void display() {
        System.out.println("ID: " + id + ", Name: " + name + ", Department: " + department);
    }
}

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

        System.out.print("Enter number of staff members: ");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline

        // Create array of OfficeStaff objects
        OfficeStaff staffArray[] = new OfficeStaff[n];

        // Input details for each staff
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Staff " + (i + 1));
            System.out.print("Enter ID: ");
            int id = sc.nextInt();
            sc.nextLine(); // consume newline
            System.out.print("Enter Name: ");
            String name = sc.nextLine();

```

```

        System.out.print("Enter Department: ");
        String dept = sc.nextLine();

        staffArray[i] = new OfficeStaff(id, name, dept);
    }

    // Display details
    System.out.println("\n--- Staff Details ---");
    for (int i = 0; i < n; i++) {
        staffArray[i].display();
    }

    sc.close();
}
}

```

## Output

Enter number of staff members: 2

Enter details for Staff 1

Enter ID: 101

Enter Name: Aditya

Enter Department: Accounts

Enter details for Staff 2

Enter ID: 102

Enter Name: Neha

Enter Department: HR

## SLIP 2.1

```

class BMI_Calculator {
    public static void main(String args[]) {
        if (args.length != 4) {
            System.out.println("Usage: java BMI_Calculator <FirstName> <LastName>
<Weight(kg)> <Height(m)>");
            return;
        }

        // Reading command line arguments
        String firstName = args[0];
        String lastName = args[1];
        double weight = Double.parseDouble(args[2]); // in kilograms
    }
}

```

```

double height = Double.parseDouble(args[3]); // in meters

// BMI Calculation
double bmi = weight / (height * height);

// Output
System.out.println("Name: " + firstName + " " + lastName);
System.out.println("Weight: " + weight + " kg");
System.out.println("Height: " + height + " m");
System.out.printf("BMI: %.2f\n", bmi);
}
}

```

## OutPUT

Name: John Doe

Weight: 70.0 kg

Height: 1.75 m

BMI: 22.86

## Slip 2.2

```

import java.util.Scanner;

class CricketPlayer {
    String name;
    int no_of_innings;
    int no_of_times_notout;
    int total_runs;
    double bat_avg;

    // Constructor
    CricketPlayer(String name, int no_of_innings, int no_of_times_notout, int total_runs) {
        this.name = name;
        this.no_of_innings = no_of_innings;
        this.no_of_times_notout = no_of_times_notout;
        this.total_runs = total_runs;
        this.bat_avg = 0.0;
    }

    // Static method to calculate batting average
    public static void avg(CricketPlayer p) {
        int timesOut = p.no_of_innings - p.no_of_times_notout;
    }
}

```

```

        if (timesOut != 0) {
            p.bat_avg = (double) p.total_runs / timesOut;
        } else {
            p.bat_avg = p.total_runs; // If never out, average = runs
        }
    }
}

// Static method to sort by batting average (descending order)
public static void sort(CricketPlayer players[]) {
    int n = players.length;
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (players[j].bat_avg < players[j + 1].bat_avg) {
                // swap
                CricketPlayer temp = players[j];
                players[j] = players[j + 1];
                players[j + 1] = temp;
            }
        }
    }
}

// Display player details
public void display() {
    System.out.printf("%-10s Innings: %-3d NotOut: %-3d Runs: %-5d Average: %.2f\n",
        name, no_of_innings, no_of_times_notout, total_runs, bat_avg);
}
}

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

        System.out.print("Enter number of players: ");
        int n = sc.nextInt();
        CricketPlayer players[] = new CricketPlayer[n];

        // Input player details
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for player " + (i + 1));
            System.out.print("Name: ");
            String name = sc.next();
            System.out.print("No. of Innings: ");
            int innings = sc.nextInt();
            System.out.print("No. of times Not Out: ");
            int notout = sc.nextInt();
            System.out.print("Total Runs: ");
            int runs = sc.nextInt();

```

```

        players[i] = new CricketPlayer(name, innings, notout, runs);
        CricketPlayer.avg(players[i]); // calculate average
    }

    // Sort players by average
    CricketPlayer.sort(players);

    // Display in sorted order
    System.out.println("\n--- Player Details Sorted by Batting Average ---");
    for (CricketPlayer p : players) {
        p.display();
    }

    sc.close();
}
}

```

## Output

Enter number of players: 3

Enter details for player 1

Name: Virat

No. of Innings: 250

No. of times Not Out: 40

Total Runs: 12000

Enter details for player 2

Name: Rohit

No. of Innings: 200

No. of times Not Out: 30

Total Runs: 9000

Enter details for player 3

Name: Dhoni

No. of Innings: 300

No. of times Not Out: 100

Total Runs: 10500

--- Player Details Sorted by Batting Average ---

Virat    Innings: 250 NotOut: 40 Runs: 12000 Average: 57.14

Dhoni    Innings: 300 NotOut: 100 Runs: 10500 Average: 52.50

Rohit    Innings: 200 NotOut: 30 Runs: 9000 Average: 47.37

## Slip 3.1

```
import java.util.Scanner;
```

```
import java.util.Arrays;
```

```
public class CitySort {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter number of cities: ");
```

```
        int n = sc.nextInt();
```

```
        sc.nextLine(); // consume newline
```

```
        String cities[] = new String[n];
```

```
        // Input city names
```

```
        for (int i = 0; i < n; i++) {
```

```
            System.out.print("Enter city name " + (i + 1) + ": ");
```

```
            cities[i] = sc.nextLine();
```

```
}

// Sort cities alphabetically

Arrays.sort(cities);

// Display sorted cities

System.out.println("\nCities in Ascending Order:");

for (String city : cities) {

    System.out.println(city);

}

sc.close();

}
```

## output

Enter number of cities: 5

Enter city name 1: Mumbai

Enter city name 2: Delhi



Enter city name 3: Kolkata

Enter city name 4: Chennai

Enter city name 5: Bangalore

Cities in Ascending Order:

Bangalore

Chennai

Delhi

Kolkata

Mumbai

## Slip 3.2

```
import java.util.Scanner;

// User-defined Exception
class CovidPositiveException extends Exception {
    CovidPositiveException(String msg) {
        super(msg);
    }
}

// Patient class
class Patient {
    String patient_name;
    int patient_age;
    int patient_oxy_level;
    int patient_HRCT_report;

    // Constructor
    Patient(String name, int age, int oxy, int hrct) {
        this.patient_name = name;
        this.patient_age = age;
        this.patient_oxy_level = oxy;
    }
}
```

```

        this.patient_HRCT_report = hrct;
    }

    // Method to check patient condition
    void checkPatient() throws CovidPositiveException {
        if (patient_oxy_level < 95 && patient_HRCT_report > 10) {
            throw new CovidPositiveException("Patient is Covid Positive(+) and Needs to be
Hospitalized!");
        } else {
            display();
        }
    }

    // Display method
    void display() {
        System.out.println("\n--- Patient Information ---");
        System.out.println("Name: " + patient_name);
        System.out.println("Age: " + patient_age);
        System.out.println("Oxygen Level: " + patient_oxy_level + "%");
        System.out.println("HRCT Report: " + patient_HRCT_report);
    }
}

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

        System.out.print("Enter Patient Name: ");
        String name = sc.nextLine();
        System.out.print("Enter Patient Age: ");
        int age = sc.nextInt();
        System.out.print("Enter Oxygen Level (%): ");
        int oxy = sc.nextInt();
        System.out.print("Enter HRCT Report value: ");
        int hrct = sc.nextInt();

        Patient p = new Patient(name, age, oxy, hrct);

        try {
            p.checkPatient();
        } catch (CovidPositiveException e) {
            System.out.println("\n*** ALERT: " + e.getMessage() + " ***");
        }

        sc.close();
    }
}

```

# Output

Enter Patient Name: Rahul  
Enter Patient Age: 35  
Enter Oxygen Level (%): 97  
Enter HRCT Report value: 8

--- Patient Information ---

Name: Rahul  
Age: 35  
Oxygen Level: 97%  
HRCT Report: 8

## Slip 4.1

```
import java.util.Scanner;

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

        // Input rows and columns
        System.out.print("Enter number of rows: ");
        int rows = sc.nextInt();
        System.out.print("Enter number of columns: ");
        int cols = sc.nextInt();

        int[][] arr = new int[rows][cols];

        // Input array elements
        System.out.println("Enter elements of the array:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                arr[i][j] = sc.nextInt();
            }
        }

        // Display original array
        System.out.println("\nOriginal Array:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(arr[i][j] + " ");
            }
        }
    }
}
```

```

        System.out.println();
    }

    // Transpose logic (swap rows and columns)
    int[][] transpose = new int[cols][rows];
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            transpose[j][i] = arr[i][j];
        }
    }

    // Display transposed array
    System.out.println("\nArray after changing rows and columns (Transpose):");
    for (int i = 0; i < cols; i++) {
        for (int j = 0; j < rows; j++) {
            System.out.print(transpose[j][i] + " ");
        }
        System.out.println();
    }

    sc.close();
}
}

```

## Output

```

Enter number of rows: 2
Enter number of columns: 3
Enter elements of the array:
1 2 3
4 5 6

```

```

Original Array:
1 2 3
4 5 6

```

```

Array after changing rows and columns (Transpose):
1 4
2 5
3 6

```

## Slip 4.2

```

import java.awt.*;
import java.awt.event.*;

```

```

// User-defined Exception
class LoginException extends Exception {
    LoginException(String msg) {
        super(msg);
    }
}

public class LoginScreen extends Frame implements ActionListener {
    Label l1, l2, l3;
    TextField t1, t2;
    Button b1, b2;
    int attempts = 0;

    LoginScreen() {
        setTitle("Login Screen");
        setSize(400, 250);
        setLayout(null);

        l1 = new Label("User Name:");
        l2 = new Label("Password:");
        l3 = new Label("");

        t1 = new TextField();
        t2 = new TextField();
        t2.setEchoChar('*'); // hide password

        b1 = new Button("Login");
        b2 = new Button("Clear");

        // set positions
        l1.setBounds(50, 50, 100, 30);
        l2.setBounds(50, 100, 100, 30);
        t1.setBounds(160, 50, 150, 30);
        t2.setBounds(160, 100, 150, 30);
        b1.setBounds(80, 160, 80, 30);
        b2.setBounds(200, 160, 80, 30);
        l3.setBounds(50, 200, 300, 30);

        // add components
        add(l1); add(t1);
        add(l2); add(t2);
        add(b1); add(b2);
        add(l3);

        b1.addActionListener(this);
        b2.addActionListener(this);

        setVisible(true);
    }
}

```

```

// close window
addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        System.exit(0);
    }
});
}

public void actionPerformed(ActionEvent ae) {
    if (ae.getSource() == b1) { // Login button
        try {
            String user = t1.getText();
            String pass = t2.getText();

            if (!user.equals(pass)) {
                attempts++;
                throw new LoginException("Invalid login! Attempts left: " + (3 - attempts));
            } else {
                l3.setText("Login Successful!");
            }
        } catch (LoginException e) {
            l3.setText(e.getMessage());
            if (attempts >= 3) {
                l3.setText("Max attempts reached! Login Disabled.");
                b1.setEnabled(false);
            }
        }
    }

    if (ae.getSource() == b2) { // Clear button
        t1.setText("");
        t2.setText("");
        l3.setText("");
    }
}

public static void main(String[] args) {
    new LoginScreen();
}
}

```

# Output

## Slip 5.1

```
import java.util.Scanner;
```

```
// Base class: Continent
```

```
class Continent {  
    String continentName;  
  
    Continent(String continentName) {  
        this.continentName = continentName;  
    }  
}
```

```
// Country class inherits Continent
```

```
class Country extends Continent {  
    String countryName;  
  
    Country(String continentName, String countryName) {  
        super(continentName);  
        this.countryName = countryName;  
    }  
}
```

```
// State class inherits Country
```

```
class State extends Country {  
    String stateName;  
    String placeName;  
  
    State(String continentName, String countryName, String stateName, String placeName) {  
        super(continentName, countryName);  
        this.stateName = stateName;  
        this.placeName = placeName;  
    }  
}
```

```
// Display details
```

```
void display() {  
    System.out.println("\n--- Place Details ---");  
    System.out.println("Place : " + placeName);  
    System.out.println("State : " + stateName);  
    System.out.println("Country : " + countryName);  
    System.out.println("Continent : " + continentName);  
}  
}
```

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

        System.out.print("Enter Continent Name: ");
        String continent = sc.nextLine();

        System.out.print("Enter Country Name: ");
        String country = sc.nextLine();

        System.out.print("Enter State Name: ");
        String state = sc.nextLine();

        System.out.print("Enter Place Name: ");
        String place = sc.nextLine();

        // Create State object
        State s = new State(continent, country, state, place);

        // Display details
        s.display();

        sc.close();
    }
}
```

## Output

```
Enter Continent Name: Asia
Enter Country Name: India
Enter State Name: Maharashtra
Enter Place Name: Mumbai
```

```
--- Place Details ---
Place : Mumbai
State : Maharashtra
Country : India
Continent : Asia
```

## Slip 5.2

```
import java.util.Scanner;

public class MatrixOperations {
    public static void main(String[] args) {
```



```

Scanner sc = new Scanner(System.in);

System.out.print("Enter number of rows: ");
int rows = sc.nextInt();
System.out.print("Enter number of columns: ");
int cols = sc.nextInt();

int[][] matrix1 = new int[rows][cols];
int[][] matrix2 = new int[rows][cols];

// Input first matrix
System.out.println("Enter elements of first matrix:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        matrix1[i][j] = sc.nextInt();
    }
}

// Input second matrix
System.out.println("Enter elements of second matrix:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        matrix2[i][j] = sc.nextInt();
    }
}

while (true) {
    System.out.println("\n--- Menu ---");
    System.out.println("1. Addition");
    System.out.println("2. Multiplication");
    System.out.println("3. Exit");
    System.out.print("Enter your choice: ");
    int choice = sc.nextInt();

    if (choice == 1) {
        int[][] sum = new int[rows][cols];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                sum[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        System.out.println("Sum of matrices:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(sum[i][j] + " ");
            }
            System.out.println();
        }
    }
}

```

```

    }

    } else if (choice == 2) {
        if (rows != cols) {
            System.out.println("Matrix multiplication requires square matrices!");
            continue;
        }
        int[][] product = new int[rows][cols];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                product[i][j] = 0;
                for (int k = 0; k < cols; k++) {
                    product[i][j] += matrix1[i][k] * matrix2[k][j];
                }
            }
        }

        System.out.println("Product of matrices:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(product[i][j] + " ");
            }
            System.out.println();
        }

    } else if (choice == 3) {
        System.out.println("Exiting program...");
        break;
    } else {
        System.out.println("Invalid choice! Try again.");
    }
}

sc.close();
}
}

```

## OutPut

Enter number of rows: 2

Enter number of columns: 2

Enter elements of first matrix:

1 2

3 4

Enter elements of second matrix:

5 6  
7 8

--- Menu ---

1. Addition
2. Multiplication
3. Exit

Enter your choice: 1

Sum of matrices:

6 8  
10 12

--- Menu ---

1. Addition
2. Multiplication
3. Exit

Enter your choice: 2

Product of matrices:

19 22  
43 50

--- Menu ---

1. Addition
2. Multiplication
3. Exit

Enter your choice: 3

Exiting program...

## Slip 6.1

// Employee class

class Employee {

int empId;

String empName;

String empDesignation;

double empSal;

// Constructor

Employee(int empId, String empName, String empDesignation, double empSal) {

this.empId = empId;

this.empName = empName;

this.empDesignation = empDesignation;

this.empSal = empSal;

}

```

// Override toString() method
@Override
public String toString() {
    return "Employee ID: " + empld +
        "\nName: " + empName +
        "\nDesignation: " + empDesignation +
        "\nSalary: " + empSal;
}
}

// Main class
public class EmployeeMain {
    public static void main(String[] args) {
        // Create Employee object
        Employee e1 = new Employee(101, "Rahul Sharma", "Software Engineer", 75000.0);

        // Display employee details using toString()
        System.out.println("--- Employee Details ---");
        System.out.println(e1);
    }
}

```

## Output

```

--- Employee Details ---
Employee ID: 101
Name: Rahul Sharma
Designation: Software Engineer
Salary: 75000.0

```

## Slip 6.2

```

import java.util.Scanner;

// Abstract class Order
abstract class Order {
    int id;
    String description;

    abstract void accept();
    abstract void display();
}

// PurchaseOrder class
class PurchaseOrder extends Order {
    String customerName;
}

```

```

Scanner sc = new Scanner(System.in);

@Override
void accept() {
    System.out.print("Enter Purchase Order ID: ");
    id = sc.nextInt();
    sc.nextLine(); // consume newline
    System.out.print("Enter Description: ");
    description = sc.nextLine();
    System.out.print("Enter Customer Name: ");
    customerName = sc.nextLine();
}

@Override
void display() {
    System.out.println("\n--- Purchase Order ---");
    System.out.println("Order ID: " + id);
    System.out.println("Description: " + description);
    System.out.println("Customer Name: " + customerName);
}
}

// SalesOrder class
class SalesOrder extends Order {
    String vendorName;
    Scanner sc = new Scanner(System.in);

    @Override
    void accept() {
        System.out.print("Enter Sales Order ID: ");
        id = sc.nextInt();
        sc.nextLine(); // consume newline
        System.out.print("Enter Description: ");
        description = sc.nextLine();
        System.out.print("Enter Vendor Name: ");
        vendorName = sc.nextLine();
    }

    @Override
    void display() {
        System.out.println("\n--- Sales Order ---");
        System.out.println("Order ID: " + id);
        System.out.println("Description: " + description);
        System.out.println("Vendor Name: " + vendorName);
    }
}

// Main class

```

```

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

        PurchaseOrder[] po = new PurchaseOrder[3];
        SalesOrder[] so = new SalesOrder[3];

        System.out.println("Enter details for 3 Purchase Orders:");
        for (int i = 0; i < 3; i++) {
            po[i] = new PurchaseOrder();
            po[i].accept();
        }

        System.out.println("\nEnter details for 3 Sales Orders:");
        for (int i = 0; i < 3; i++) {
            so[i] = new SalesOrder();
            so[i].accept();
        }

        System.out.println("\n--- Displaying Purchase Orders ---");
        for (int i = 0; i < 3; i++) {
            po[i].display();
        }

        System.out.println("\n--- Displaying Sales Orders ---");
        for (int i = 0; i < 3; i++) {
            so[i].display();
        }

        sc.close();
    }
}

```

# Output

Enter details for 3 Purchase Orders:

Enter Purchase Order ID: 101

Enter Description: Laptop

Enter Customer Name: Rahul

Enter Purchase Order ID: 102

Enter Description: Monitor

Enter Customer Name: Priya

Enter Purchase Order ID: 103

Enter Description: Keyboard

Enter Customer Name: Ankit

Enter details for 3 Sales Orders:

Enter Sales Order ID: 201

Enter Description: Hard Disk

Enter Vendor Name: Dell

Enter Sales Order ID: 202

Enter Description: Mouse

Enter Vendor Name: HP

Enter Sales Order ID: 203

Enter Description: Printer

Enter Vendor Name: Canon

--- Displaying Purchase Orders ---

--- Purchase Order ---

Order ID: 101

Description: Laptop

Customer Name: Rahul

--- Purchase Order ---

Order ID: 102

Description: Monitor

Customer Name: Priya

--- Purchase Order ---

Order ID: 103

Description: Keyboard

Customer Name: Ankit



--- Displaying Sales Orders ---

--- Sales Order ---

Order ID: 201

Description: Hard Disk

Vendor Name: Dell

--- Sales Order ---

Order ID: 202

Description: Mouse

Vendor Name: HP

--- Sales Order ---

Order ID: 203

Description: Printer

Vendor Name: Canon

## Slip 7.1

```
import java.util.Scanner;
```

```
class Bank {  
    private String accountNumber;  
    private String accountHolder;  
    private double balance;
```

```
    // Constructor
```

```

Bank(String accountNumber, String accountHolder, double balance) {
    this.accountNumber = accountNumber;
    this.accountHolder = accountHolder;
    this.balance = balance;
}

// Deposit method
void deposit(double amount) {
    if (amount > 0) {
        balance += amount;
        System.out.println("₹" + amount + " deposited successfully.");
    } else {
        System.out.println("Invalid deposit amount!");
    }
}

// Withdraw method
void withdraw(double amount) {
    if (amount > 0) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("₹" + amount + " withdrawn successfully.");
        } else {
            System.out.println("Insufficient balance!");
        }
    } else {
        System.out.println("Invalid withdrawal amount!");
    }
}

// Get balance and details
void getBalance() {
    System.out.println("\n--- Account Details ---");
    System.out.println("Account Number: " + accountNumber);
    System.out.println("Account Holder: " + accountHolder);
    System.out.println("Balance: ₹" + balance);
}

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

        // Create Bank object
        System.out.print("Enter Account Number: ");
        String accNo = sc.nextLine();
        System.out.print("Enter Account Holder Name: ");
    }
}

```

```

String accHolder = sc.nextLine();
System.out.print("Enter Initial Balance: ₹");
double initialBalance = sc.nextDouble();

Bank b = new Bank(accNo, accHolder, initialBalance);

int choice;
do {
    System.out.println("\n--- Bank Menu ---");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Check Balance");
    System.out.println("4. Exit");
    System.out.print("Enter your choice: ");
    choice = sc.nextInt();

    switch (choice) {
        case 1:
            System.out.print("Enter amount to deposit: ₹");
            double dep = sc.nextDouble();
            b.deposit(dep);
            break;
        case 2:
            System.out.print("Enter amount to withdraw: ₹");
            double wit = sc.nextDouble();
            b.withdraw(wit);
            break;
        case 3:
            b.getBalance();
            break;
        case 4:
            System.out.println("Exiting...");
            break;
        default:
            System.out.println("Invalid choice! Try again.");
    }

} while (choice != 4);

sc.close();
}
}

```

## Output

```

Enter Account Number: 12345
Enter Account Holder Name: Rahul

```

Enter Initial Balance: ₹5000

--- Bank Menu ---

1. Deposit
2. Withdraw
3. Check Balance
4. Exit

Enter your choice: 1

Enter amount to deposit: ₹2000

₹2000 deposited successfully.

--- Bank Menu ---

Enter your choice: 2

Enter amount to withdraw: ₹1000

₹1000 withdrawn successfully.

--- Bank Menu ---

Enter your choice: 3

--- Account Details ---

Account Number: 12345

Account Holder: Rahul

Balance: ₹6000.0

--- Bank Menu ---

Enter your choice: 4

Exiting...

## Slip 7.2

```
import java.io.*;
```

```
import java.util.Scanner;
```

```
public class FileReverseCase {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter the path of the text file: ");
```

```
        String filePath = sc.nextLine();
```

```
        try {
```

```
            // Read the file
```

```
            BufferedReader br = new BufferedReader(new FileReader(filePath));
```

```
            StringBuilder content = new StringBuilder();
```

```
            String line;
```

```
            while ((line = br.readLine()) != null) {
```

```

        content.append(line).append("\n");
    }

    br.close();

    // Reverse the content
    content.reverse();

    // Change case
    StringBuilder finalOutput = new StringBuilder();
    for (int i = 0; i < content.length(); i++) {
        char ch = content.charAt(i);
        if (Character.isUpperCase(ch)) {
            finalOutput.append(Character.toLowerCase(ch));
        } else if (Character.isLowerCase(ch)) {
            finalOutput.append(Character.toUpperCase(ch));
        } else {
            finalOutput.append(ch);
        }
    }

    // Display final output
    System.out.println("\n--- Reversed Content with Changed Case ---");
    System.out.println(finalOutput.toString());

    } catch (FileNotFoundException e) {
        System.out.println("File not found! Please check the path.");
    } catch (IOException e) {
        System.out.println("Error reading the file.");
    }
}

sc.close();
}
}

```

## Slip 8.1

```

import java.util.Scanner;

class Sphere {
    double radius;

    // Constructor
    Sphere(double radius) {
        this.radius = radius;
    }
}

```

```

// Method to calculate surface area
double surfaceArea() {
    return 4 * 3.14 * radius * radius;
}

// Method to calculate volume
double volume() {
    return (4.0 / 3.0) * 3.14 * radius * radius * radius;
}
}

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

        System.out.print("Enter radius of the sphere: ");
        double r = sc.nextDouble();

        Sphere s = new Sphere(r);

        System.out.println("\n--- Sphere Details ---");
        System.out.printf("Radius: %.2f\n", r);
        System.out.printf("Surface Area: %.2f\n", s.surfaceArea());
        System.out.printf("Volume: %.2f\n", s.volume());

        sc.close();
    }
}

```

## Output

Enter radius of the sphere: 5

--- Sphere Details ---

Radius: 5.00

Surface Area: 314.00

Volume: 523.33

## Slip 8.2

```
import java.awt.*;
```

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

```

public class MouseEventDemo extends Frame implements MouseListener,
MouseListener {
    Label l1;

```

TextField tf;

```
MouseEventDemo() {
    setTitle("Mouse Event Demo");
    setSize(400, 300);
    setLayout(null);

    l1 = new Label("Mouse Click Position:");
    l1.setBounds(50, 50, 150, 30);
    add(l1);

    tf = new TextField();
    tf.setBounds(200, 50, 150, 30);
    tf.setEditable(false);
    add(tf);

    // Add mouse listeners
    addMouseListener(this);
    addMouseMotionListener(this);

    setVisible(true);

    // Close window
    addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent we) {
            System.exit(0);
        }
    });
}

// MouseListener methods
public void mouseClicked(MouseEvent me) {
    int x = me.getX();
    int y = me.getY();
    tf.setText("X: " + x + ", Y: " + y);
}

public void mousePressed(MouseEvent me) {}
public void mouseReleased(MouseEvent me) {}
public void mouseEntered(MouseEvent me) {}
public void mouseExited(MouseEvent me) {}

// MouseMotionListener methods
public void mouseMoved(MouseEvent me) {
    setTitle("Mouse Moved at X: " + me.getX() + ", Y: " + me.getY());
}

public void mouseDragged(MouseEvent me) {}
```

```
        public static void main(String[] args) {
            new MouseEventDemo();
        }
    }
```

# Output

## Slip 9.1

```
import java.util.Scanner;
```

```
class Clock {
    private int hours;
    private int minutes;
    private int seconds;

    // Constructor
    Clock(int h, int m, int s) {
        if (isValidTime(h, m, s)) {
            this.hours = h;
            this.minutes = m;
            this.seconds = s;
        } else {
            System.out.println("Invalid time! Setting to 00:00:00");
            this.hours = 0;
            this.minutes = 0;
            this.seconds = 0;
        }
    }

    // Method to check validity of time
    boolean isValidTime(int h, int m, int s) {
        return (h >= 0 && h < 24) && (m >= 0 && m < 60) && (s >= 0 && s < 60);
    }

    // Method to display time in 24-hour format
    void display24HourFormat() {
        System.out.printf("Time (24-hour): %02d:%02d:%02d\n", hours, minutes, seconds);
    }

    // Method to display time in AM/PM format
    void displayAMPM() {
        String period = (hours >= 12) ? "PM" : "AM";
        int displayHour = (hours % 12 == 0) ? 12 : hours % 12;
    }
}
```



```
        System.out.printf("Time (AM/PM): %02d:%02d:%02d %s\n", displayHour, minutes,
seconds, period);
    }
}
```

```
// Method to accept time from user
```

```
void acceptTime() {
```

```
    Scanner sc = new Scanner(System.in);
```

```
    System.out.print("Enter hours (0-23): ");
```

```
    int h = sc.nextInt();
```

```
    System.out.print("Enter minutes (0-59): ");
```

```
    int m = sc.nextInt();
```

```
    System.out.print("Enter seconds (0-59): ");
```

```
    int s = sc.nextInt();
```

```
    if (isValidTime(h, m, s)) {
```

```
        hours = h;
```

```
        minutes = m;
```

```
        seconds = s;
```

```
    } else {
```

```
        System.out.println("Invalid time entered! Keeping previous time.");
```

```
    }
```

```
}
```

```
}
```

```
// Main class
```

```
public class ClockMain {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.println("Enter initial time:");
```

```
        System.out.print("Hours: ");
```

```
        int h = sc.nextInt();
```

```
        System.out.print("Minutes: ");
```

```
        int m = sc.nextInt();
```

```
        System.out.print("Seconds: ");
```

```
        int s = sc.nextInt();
```

```
        Clock c = new Clock(h, m, s);
```

```
        // Display time
```

```
        c.display24HourFormat();
```

```
        c.displayAMPM();
```

```
        // Optionally allow user to update time
```

```
        System.out.println("\nUpdate Time:");
```

```
        c.acceptTime();
```

```
        // Display updated time
```

```

        c.display24HourFormat();
        c.displayAMPM();

        sc.close();
    }
}

```

## Output

Enter initial time:  
Hours: 14  
Minutes: 30  
Seconds: 45  
Time (24-hour): 14:30:45  
Time (AM/PM): 02:30:45 PM

Update Time:  
Enter hours (0-23): 9  
Enter minutes (0-59): 15  
Enter seconds (0-59): 5  
Time (24-hour): 09:15:05  
Time (AM/PM): 09:15:05 AM

## Slip 9.2

```

// Marker Interface
interface ProductMarker {
    // No methods, just a marker
}

```

```

// Product class implementing marker interface
class Product implements ProductMarker {
    int product_id;
    String product_name;
    double product_cost;
    int product_quantity;

    static int count = 0; // Static variable to track object count

    // Default constructor
    Product() {
        this.product_id = 0;
        this.product_name = "Unknown";
        this.product_cost = 0.0;
        this.product_quantity = 0;
        count++;
    }
}

```

```

    }

    // Parameterized constructor
    Product(int id, String name, double cost, int quantity) {
        this.product_id = id;
        this.product_name = name;
        this.product_cost = cost;
        this.product_quantity = quantity;
        count++;
    }

    // Method to display product details
    void display() {
        System.out.println("\n--- Product Details ---");
        System.out.println("Product ID: " + product_id);
        System.out.println("Product Name: " + product_name);
        System.out.println("Product Cost: " + product_cost);
        System.out.println("Product Quantity: " + product_quantity);
    }
}

// Main class
public class ProductMain {
    public static void main(String[] args) {
        // Creating objects
        Product p1 = new Product(); // default constructor
        Product p2 = new Product(101, "Laptop", 75000, 5);
        Product p3 = new Product(102, "Mouse", 500, 50);

        // Display objects
        p1.display();
        p2.display();
        p3.display();

        // Display object count
        System.out.println("\nTotal Product objects created: " + Product.count);
    }
}

```

## Output

```

--- Product Details ---
Product ID: 0
Product Name: Unknown
Product Cost: 0.0
Product Quantity: 0

```

--- Product Details ---  
Product ID: 101  
Product Name: Laptop  
Product Cost: 75000.0  
Product Quantity: 5

--- Product Details ---  
Product ID: 102  
Product Name: Mouse  
Product Cost: 500.0  
Product Quantity: 50

Total Product objects created: 3

## Slip 10.1

```
import java.util.Scanner;

// Functional interface
@FunctionalInterface
interface CubeCalculator {
    int calculate(int x);
}

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

        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        // Lambda expression to calculate cube
        CubeCalculator cube = (x) -> x * x * x;

        int result = cube.calculate(num);

        System.out.println("Cube of " + num + " is: " + result);

        sc.close();
    }
}
```

## Output

Enter a number: 5  
Cube of 5 is: 125

# Slip 10.2

file: `StudentInfo.java`

```
package student;

public class StudentInfo {
    int rollNo;
    String name;
    String studentClass;

    // Constructor
    public StudentInfo(int rollNo, String name, String studentClass) {
        this.rollNo = rollNo;
        this.name = name;
        this.studentClass = studentClass;
    }

    // Method to display student info
    public void displayInfo() {
        System.out.println("\n--- Student Information ---");
        System.out.println("Roll No: " + rollNo);
        System.out.println("Name: " + name);
        System.out.println("Class: " + studentClass);
    }
}
```

File: `StudentPer.java`

```
package student;

public class StudentPer {
    int[] marks;

    // Constructor
    public StudentPer(int[] marks) {
        this.marks = marks;
    }

    // Method to calculate percentage
    public double calculatePercentage() {
        int total = 0;
        for (int m : marks) {
```

```

        total += m;
    }
    return (total / (marks.length * 100.0)) * 100; // assuming
each subject is out of 100
}

// Display percentage
public void displayPercentage() {
    System.out.printf("Percentage: %.2f%%\n",
calculatePercentage());
}
}

```

**File:** StudentMain.java

```

import java.util.Scanner;
import student.StudentInfo;
import student.StudentPer;

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

        System.out.print("Enter Roll No: ");
        int roll = sc.nextInt();
        sc.nextLine(); // consume newline

        System.out.print("Enter Name: ");
        String name = sc.nextLine();

        System.out.print("Enter Class: ");
        String cls = sc.nextLine();

        int[] marks = new int[6];
        System.out.println("Enter marks for 6 subjects (out of
100):");
        for (int i = 0; i < 6; i++) {
            System.out.print("Subject " + (i+1) + ": ");
            marks[i] = sc.nextInt();
        }
    }
}

```

```
        // Create StudentInfo object
        StudentInfo sInfo = new StudentInfo(roll, name, cls);
        sInfo.displayInfo();

        // Create StudentPer object
        StudentPer sPer = new StudentPer(marks);
        sPer.displayPercentage();

        sc.close();
    }
}
```

## Output

```
Enter Roll No: 101
Enter Name: Rahul Sharma
Enter Class: 10th
Enter marks for 6 subjects (out of 100):
Subject 1: 85
Subject 2: 90
Subject 3: 78
Subject 4: 88
Subject 5: 92
Subject 6: 80
```

```
--- Student Information ---
Roll No: 101
Name: Rahul Sharma
Class: 10th
Percentage: 85.50%
```

## Slip 11.1

```
// Interface Operation
interface Operation {
    double PI = 3.142; // Constant

    // Abstract method to calculate volume
    double volume();
}
```

```

}

// Cylinder class implementing Operation
class Cylinder implements Operation {
    double radius;
    double height;

    // Constructor
    Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }

    // Implement volume method
    @Override
    public double volume() {
        return PI * radius * radius * height; // Volume =  $\pi * r^2 * h$ 
    }
}

// Main class
public class CylinderMain {
    public static void main(String[] args) {
        // Example: Create a cylinder object
        Cylinder c = new Cylinder(5, 10); // radius = 5, height = 10

        System.out.println("--- Cylinder Volume ---");
        System.out.println("Radius: " + c.radius);
        System.out.println("Height: " + c.height);
        System.out.printf("Volume: %.2f\n", c.volume());
    }
}

```

## Output

```

--- Cylinder Volume ---
Radius: 5.0
Height: 10.0
Volume: 785.50

```

## Slip 11.2

```
import java.util.Scanner;
```



```
// Custom Exception class
class InvalidPasswordException extends Exception {
    InvalidPasswordException(String msg) {
        super(msg);
    }
}

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

        System.out.print("Enter Username: ");
        String username = sc.nextLine();

        System.out.print("Enter Password: ");
        String password = sc.nextLine();

        try {
            if (!username.equals(password)) {
                throw new InvalidPasswordException("Invalid Password! Username and Password
do not match.");
            } else {
                System.out.println("Login Successful!");
            }
        } catch (InvalidPasswordException e) {
            System.out.println(e.getMessage());
        }

        sc.close();
    }
}
```

## Output

```
Enter Username: admin
Enter Password: admin
Login Successful!
```

```
Enter Username: admin
Enter Password: 12345
Invalid Password! Username and Password do not match.
```

## Slip 12.1

```
// Parent class College
```

```

class College {
    int cno;
    String cname;
    String caddr;

    // Constructor
    College(int cno, String cname, String caddr) {
        this.cno = cno;
        this.cname = cname;
        this.caddr = caddr;
    }

    // Method to display college details
    void displayCollegeDetails() {
        System.out.println("\n--- College Details ---");
        System.out.println("College No: " + cno);
        System.out.println("College Name: " + cname);
        System.out.println("College Address: " + caddr);
    }
}

// Derived class Department
class Department extends College {
    int dno;
    String dname;

    // Constructor
    Department(int cno, String cname, String caddr, int dno, String dname) {
        super(cno, cname, caddr); // Call parent constructor
        this.dno = dno;
        this.dname = dname;
    }

    // Method to display department details along with college details
    void displayDepartmentDetails() {
        displayCollegeDetails(); // Display college info
        System.out.println("\n--- Department Details ---");
        System.out.println("Department No: " + dno);
        System.out.println("Department Name: " + dname);
    }
}

// Main class
public class CollegeMain {
    public static void main(String[] args) {
        // Create Department object
        Department dept = new Department(101, "ABC College", "Delhi", 1, "Computer
Science");
    }
}

```

```
        // Display details
        dept.displayDepartmentDetails();
    }
}
```

## Output

--- College Details ---

College No: 101

College Name: ABC College

College Address: Delhi

--- Department Details ---

Department No: 1

Department Name: Computer Science

## Slip 12.2

```
import java.awt.*;
```

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

```
public class SimpleCalculator extends Frame implements ActionListener {
```

```
    TextField tf;
```

```
    Button[] numButtons = new Button[10];
```

```
    Button add, sub, mul, mod, eq, clr;
```

```
    double num1 = 0, num2 = 0, result = 0;
```

```
    char operator;
```

```
    SimpleCalculator() {
```

```
        setTitle("Simple Calculator");
```

```
        setSize(300, 400);
```

```
        setLayout(new BorderLayout());
```

```
        // TextField to display result
```

```
        tf = new TextField();
```

```
        tf.setEditable(false);
```

```
        add(tf, BorderLayout.NORTH);
```

```
        // Panel for buttons
```

```
        Panel panel = new Panel();
```

```
        panel.setLayout(new GridLayout(4, 4, 5, 5)); // 4x4 grid
```

```
        // Number buttons
```

```

for (int i = 1; i <= 9; i++) {
    numButtons[i] = new Button(String.valueOf(i));
    numButtons[i].addActionListener(this);
    panel.add(numButtons[i]);
}

// Operation buttons
add = new Button("+");
sub = new Button("-");
mul = new Button("*");
mod = new Button("%");
eq = new Button("=");
clr = new Button("C");
numButtons[0] = new Button("0");

// Add buttons to panel
panel.add(add);
panel.add(numButtons[0]);
panel.add(sub);
panel.add(mul);
panel.add(mod);
panel.add(eq);
panel.add(clr);

add(panel, BorderLayout.CENTER);

// Add action listeners
add.addActionListener(this);
sub.addActionListener(this);
mul.addActionListener(this);
mod.addActionListener(this);
eq.addActionListener(this);
clr.addActionListener(this);

setVisible(true);

// Close window
addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        System.exit(0);
    }
});
}

@Override
public void actionPerformed(ActionEvent ae) {
    String command = ae.getActionCommand();

```

```

// If digit pressed
if (command.charAt(0) >= '0' && command.charAt(0) <= '9') {
    tf.setText(tf.getText() + command);
}
// If operation pressed
else if (command.equals("+") || command.equals("-") || command.equals("*") ||
command.equals("%")) {
    num1 = Double.parseDouble(tf.getText());
    operator = command.charAt(0);
    tf.setText("");
}
// If equal pressed
else if (command.equals("=")) {
    num2 = Double.parseDouble(tf.getText());
    switch (operator) {
        case '+':
            result = num1 + num2;
            break;
        case '-':
            result = num1 - num2;
            break;
        case '*':
            result = num1 * num2;
            break;
        case '%':
            result = num1 % num2;
            break;
    }
    tf.setText(String.valueOf(result));
}
// Clear
else if (command.equals("C")) {
    tf.setText("");
    num1 = num2 = result = 0;
}
}

public static void main(String[] args) {
    new SimpleCalculator();
}
}

```

## Slip 13.1

```
import java.io.*;
```

```

public class FileWordLineCount {
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Usage: java FileWordLineCount <filename>");
            return;
        }

        String fileName = args[0];
        File file = new File(fileName);

        if (!file.exists()) {
            System.out.println("File does not exist: " + fileName);
            return;
        }

        int lineCount = 0;
        int wordCount = 0;

        try (BufferedReader br = new BufferedReader(new FileReader(file))) {
            String line;

            while ((line = br.readLine()) != null) {
                lineCount++;

                // Split line into words
                String[] words = line.trim().split("\\s+");
                if (words.length == 1 && words[0].equals("")) {
                    // Empty line, ignore
                    continue;
                }
                wordCount += words.length;
            }

            System.out.println("File: " + fileName);
            System.out.println("Number of lines: " + lineCount);
            System.out.println("Number of words: " + wordCount);

        } catch (IOException e) {
            System.out.println("Error reading the file: " + e.getMessage());
        }
    }
}

```

## Output

File: sample.txt

Number of lines: 5  
Number of words: 42

## Slip 13.2

```
import java.text.SimpleDateFormat;
import java.util.Date;

public class DateTimeFormats {
    public static void main(String[] args) {
        Date now = new Date();

        // Format 1: dd/MM/yyyy
        SimpleDateFormat format1 = new SimpleDateFormat("dd/MM/yyyy");
        System.out.println("Current date is : " + format1.format(now));

        // Format 2: MM-dd-yyyy
        SimpleDateFormat format2 = new SimpleDateFormat("MM-dd-yyyy");
        System.out.println("Current date is : " + format2.format(now));

        // Format 3: EEEE MMMM dd yyyy (Tuesday August 31 2021)
        SimpleDateFormat format3 = new SimpleDateFormat("EEEE MMMM dd yyyy");
        System.out.println("Current date is : " + format3.format(now));

        // Format 4: EEE MMMM dd HH:mm:ss z yyyy (Fri August 31 15:25:59 IST 2021)
        SimpleDateFormat format4 = new SimpleDateFormat("EEE MMMM dd HH:mm:ss z
yyyy");
        System.out.println("Current date and time is : " + format4.format(now));

        // Format 5: dd/MM/yy hh:mm:ss a Z (31/08/21 15:25:59 PM +0530)
        SimpleDateFormat format5 = new SimpleDateFormat("dd/MM/yy hh:mm:ss a Z");
        System.out.println("Current date and time is : " + format5.format(now));
    }
}
```

## Output

Current date is : 29/09/2025  
Current date is : 09-29-2025  
Current date is : Monday September 29 2025  
Current date and time is : Mon September 29 14:42:30 IST 2025  
Current date and time is : 29/09/25 02:42:30 PM +0530

# Slip 14.1

```
import java.util.Scanner;

// User-defined Exception
class ZeroNumberException extends Exception {
    ZeroNumberException(String msg) {
        super(msg);
    }
}

public class PrimeChecker {

    // Static method to check prime
    static boolean isPrime(int n) {
        if (n <= 1) return false;
        for (int i = 2; i <= Math.sqrt(n); i++) {
            if (n % i == 0) return false;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        try {
            if (num == 0) {
                throw new ZeroNumberException("Number is 0");
            }

            if (isPrime(num)) {
                System.out.println(num + " is a prime number.");
            } else {
                System.out.println(num + " is not a prime number.");
            }
        } catch (ZeroNumberException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        sc.close();
    }
}
```



# Slip 14.2

## Step 1: Package SY

File: SYMarks.java

```
package SY;

public class SYMarks {
    public int computerTotal;
    public int mathsTotal;
    public int electronicsTotal;

    public SYMarks(int computerTotal, int mathsTotal, int
electronicsTotal) {
        this.computerTotal = computerTotal;
        this.mathsTotal = mathsTotal;
        this.electronicsTotal = electronicsTotal;
    }

    // Method to get total marks in SY
    public int totalSYMarks() {
        return computerTotal + mathsTotal + electronicsTotal;
    }
}
```

## Step 2: Package TY

File: TYMarks.java

```
package TY;

public class TYMarks {
    public int theory;
    public int practicals;

    public TYMarks(int theory, int practicals) {
        this.theory = theory;
        this.practicals = practicals;
    }
}
```

```
        // Method to get total marks in TY
        public int totalTYMarks() {
            return theory + practicals;
        }
    }
}
```

---

### Step 3: Student Class

**File:** Student.java

```
import SY.SYMarks;
import TY.TYMarks;

public class Student {
    int rollNumber;
    String name;
    SYMarks syMarks;
    TYMarks tyMarks;

    public Student(int rollNumber, String name, SYMarks syMarks,
TYMarks tyMarks) {
        this.rollNumber = rollNumber;
        this.name = name;
        this.syMarks = syMarks;
        this.tyMarks = tyMarks;
    }

    // Calculate grade based on Computer total marks (SY + TY)
    public void displayResult() {
        int computerTotal = syMarks.computerTotal + tyMarks.theory;
// Assuming TY theory as computer marks
        String grade;

        if (computerTotal >= 70) grade = "A";
        else if (computerTotal >= 60) grade = "B";
        else if (computerTotal >= 50) grade = "C";
        else if (computerTotal >= 40) grade = "Pass Class";
        else grade = "FAIL";
    }
}
```

```

        System.out.println("\n--- Student Result ---");
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Name: " + name);
        System.out.println("SY Computer: " + syMarks.computerTotal);
        System.out.println("TY Computer Theory: " + tyMarks.theory);
        System.out.println("Total Computer Marks: " +
computerTotal);
        System.out.println("Grade: " + grade);
    }
}

```

---

## Step 4: Main Class to Test

**File:** Main.java

```

import java.util.Scanner;
import SY.SYMarks;
import TY.TYMarks;

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

        System.out.print("Enter number of students: ");
        int n = sc.nextInt();
        sc.nextLine();

        Student[] students = new Student[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Student " +
(i+1));

            System.out.print("Roll Number: ");
            int roll = sc.nextInt();
            sc.nextLine();

            System.out.print("Name: ");
            String name = sc.nextLine();

```

```

        System.out.print("SY Computer Marks: ");
        int syComp = sc.nextInt();
        System.out.print("SY Maths Marks: ");
        int syMath = sc.nextInt();
        System.out.print("SY Electronics Marks: ");
        int syElec = sc.nextInt();

        System.out.print("TY Computer Theory Marks: ");
        int tyTheory = sc.nextInt();
        System.out.print("TY Computer Practicals Marks: ");
        int tyPract = sc.nextInt();
        sc.nextLine();

        SYMarks sy = new SYMarks(syComp, syMath, syElec);
        TYMarks ty = new TYMarks(tyTheory, tyPract);

        students[i] = new Student(roll, name, sy, ty);
    }

    // Display results
    for (Student s : students) {
        s.displayResult();
    }

    sc.close();
}
}

```

---

## Sample Input & Output

### Input:

Enter number of students: 1

Roll Number: 101

Name: Rahul

SY Computer Marks: 35

SY Maths Marks: 40

SY Electronics Marks: 38

TY Computer Theory Marks: 40

TY Computer Practicals Marks: 35

**Output:**

```
--- Student Result ---  
Roll Number: 101  
Name: Rahul  
SY Computer: 35  
TY Computer Theory: 40  
Total Computer Marks: 75  
Grade: A
```

---

## Slip 15.1

```
import java.io.*;  
import java.util.Scanner;  
  
public class FileCopyBooks {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
  
        System.out.print("Enter the source file name: ");  
        String sourceFile = sc.nextLine();  
  
        System.out.print("Enter the destination file name: ");  
        String destFile = sc.nextLine();  
  
        File src = new File(sourceFile);  
        File dest = new File(destFile);  
  
        if (!src.exists()) {  
            System.out.println("Source file does not exist: " + sourceFile);  
            sc.close();  
            return;  
        }  
  
        try (BufferedReader br = new BufferedReader(new FileReader(src));  
            BufferedWriter bw = new BufferedWriter(new FileWriter(dest))) {  
  
            String line;  
            while ((line = br.readLine()) != null) {  
                bw.write(line);  
                bw.newLine(); // preserve line breaks  
            }  
        }  
    }  
}
```

```

        System.out.println("Contents copied from " + sourceFile + " to " + destFile);

    } catch (IOException e) {
        System.out.println("Error: " + e.getMessage());
    }

    sc.close();
}
}

```

## ✓ Sample Run

**Source file (`books.txt`):**

The Alchemist, Paulo Coelho  
Harry Potter, J.K. Rowling

**Destination file (`copy.txt`) after running the program:**

The Alchemist, Paulo Coelho  
Harry Potter, J.K. Rowling

# Slip 15.2

```

// Parent class Account
class Account {
    String custName;
    int accNo;

    // Default constructor
    Account() {
        custName = "Unknown";
        accNo = 0;
    }

    // Parameterized constructor
    Account(String custName, int accNo) {
        this.custName = custName;
        this.accNo = accNo;
    }
}

// Subclass SavingAccount

```

```

class SavingAccount extends Account {
    double savingBal;
    double minBal;

    // Default constructor
    SavingAccount() {
        super();
        savingBal = 0.0;
        minBal = 0.0;
    }

    // Parameterized constructor
    SavingAccount(String custName, int accNo, double savingBal, double minBal) {
        super(custName, accNo);
        this.savingBal = savingBal;
        this.minBal = minBal;
    }
}

// Derived class AccountDetail
class AccountDetail extends SavingAccount {
    double depositAmt;
    double withdrawalAmt;

    // Parameterized constructor
    AccountDetail(String custName, int accNo, double savingBal, double minBal, double
depositAmt, double withdrawalAmt) {
        super(custName, accNo, savingBal, minBal);
        this.depositAmt = depositAmt;
        this.withdrawalAmt = withdrawalAmt;
    }

    // Method to display customer details
    void displayDetails() {
        System.out.println("\n--- Customer Account Details ---");
        System.out.println("Customer Name: " + custName);
        System.out.println("Account Number: " + accNo);
        System.out.println("Saving Balance: " + savingBal);
        System.out.println("Minimum Balance: " + minBal);
        System.out.println("Deposit Amount: " + depositAmt);
        System.out.println("Withdrawal Amount: " + withdrawalAmt);
        double finalBal = savingBal + depositAmt - withdrawalAmt;
        System.out.println("Final Balance: " + finalBal);
    }
}

// Main class
public class AccountMain {

```

```

public static void main(String[] args) {
    // Create AccountDetail object
    AccountDetail customer = new AccountDetail("Rahul Sharma", 101, 5000, 1000, 2000,
1500);

    // Display customer details
    customer.displayDetails();
}
}

```

## Output

```

--- Customer Account Details ---
Customer Name: Rahul Sharma
Account Number: 101
Saving Balance: 5000.0
Minimum Balance: 1000.0
Deposit Amount: 2000.0
Withdrawal Amount: 1500.0
Final Balance: 5500.0

```

## Slip 16.1

```

import java.util.Scanner;

// Functional Interface
@FunctionalInterface
interface SquareCalculator {
    int calculate(int x);
}

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

        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        // Lambda expression to calculate square
        SquareCalculator square = (x) -> x * x;

        int result = square.calculate(num);

        System.out.println("Square of " + num + " is: " + result);
    }
}

```



```
        sc.close();
    }
}
```

## Sample Run

```
Enter a number: 7
Square of 7 is: 49
```

# Slip 16.2

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

```
public class MenuExample extends Frame {
```

```
    MenuExample() {
        // Create MenuBar
        MenuBar mb = new MenuBar();

        // File Menu
        Menu file = new Menu("File");
        MenuItem newFile = new MenuItem("New");
        MenuItem open = new MenuItem("Open");
        MenuItem save = new MenuItem("Save");
        file.add(newFile);
        file.add(open);
        file.add(save);

        // Edit Menu
        Menu edit = new Menu("Edit");
        MenuItem editItem = new MenuItem("Edit");
        edit.add(editItem);

        // About Menu
        Menu about = new Menu("About");
        CheckboxMenuItem showAbout = new CheckboxMenuItem("Show About");
        about.add(showAbout);

        // Add Menus to MenuBar
        mb.add(file);
        mb.add(edit);
        mb.add(about);

        // Set MenuBar
```

```

setMenuBar(mb);

// Frame settings
setTitle("Java AWT Examples");
setSize(400, 300);
setLayout(null);
setVisible(true);

// Closing window
addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        System.exit(0);
    }
});
}

public static void main(String[] args) {
    new MenuExample();
}
}

```

## Slip 17.1

```

import java.util.Scanner;

// Superclass
class Customer {
    protected String name;
    protected String phoneNumber;

    public void readCustomer() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter customer name: ");
        name = sc.nextLine();
        System.out.print("Enter phone number: ");
        phoneNumber = sc.nextLine();
    }

    public void displayCustomer() {
        System.out.println("Customer Name: " + name);
        System.out.println("Phone Number: " + phoneNumber);
    }
}

// Derived class from Customer

```

```

class Depositor extends Customer {
    protected String accNo;
    protected double balance;

    public void readDepositor() {
        readCustomer(); // Read Customer details
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account number: ");
        accNo = sc.nextLine();
        System.out.print("Enter balance: ");
        balance = sc.nextDouble();
    }

    public void displayDepositor() {
        displayCustomer(); // Display Customer details
        System.out.println("Account Number: " + accNo);
        System.out.println("Balance: " + balance);
    }
}

// Derived class from Depositor
class Borrower extends Depositor {
    private String loanNo;
    private double loanAmt;

    public void readBorrower() {
        readDepositor(); // Read Depositor (and Customer) details
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter loan number: ");
        loanNo = sc.nextLine();
        System.out.print("Enter loan amount: ");
        loanAmt = sc.nextDouble();
    }

    public void displayBorrower() {
        displayDepositor(); // Display Depositor (and Customer) details
        System.out.println("Loan Number: " + loanNo);
        System.out.println("Loan Amount: " + loanAmt);
        System.out.println("-----");
    }
}

// Main class
public class BankDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of customers: ");
        int n = sc.nextInt();
    }
}

```

```

Borrower[] customers = new Borrower[n];

// Reading details
for (int i = 0; i < n; i++) {
    System.out.println("\nEnter details for Customer " + (i + 1) + ":");
    customers[i] = new Borrower();
    customers[i].readBorrower();
}

// Displaying details
System.out.println("\nCustomer Details:");
for (int i = 0; i < n; i++) {
    customers[i].displayBorrower();
}
}
}

```

## Output

Enter number of customers: 2

Enter details for Customer 1:

Enter customer name: Aditya

Enter phone number: 9876543210

Enter account number: ACC1001

Enter balance: 50000

Enter loan number: L001

Enter loan amount: 20000

Enter details for Customer 2:

Enter customer name: Priya

Enter phone number: 9123456780

Enter account number: ACC1002

Enter balance: 75000

Enter loan number: L002

Enter loan amount: 30000

Customer Details:

Customer Name: Aditya

Phone Number: 9876543210

Account Number: ACC1001

Balance: 50000.0

Loan Number: L001

Loan Amount: 20000.0

-----

Customer Name: Priya

Phone Number: 9123456780

Account Number: ACC1002  
Balance: 75000.0  
Loan Number: L002  
Loan Amount: 30000.0  
-----

## Slip 17.2

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

public class StringManipulation extends JFrame implements ActionListener {

    // Text fields
    JTextField tf1, tf2, tf3;
    // Buttons
    JButton btnConcat, btnReverse;

    public StringManipulation() {
        // Set Frame properties
        setTitle("String Manipulation");
        setSize(400, 200);
        setLayout(new GridLayout(4, 2, 10, 10));
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Initialize components
        JLabel lbl1 = new JLabel("String 1:");
        JLabel lbl2 = new JLabel("String 2:");
        JLabel lbl3 = new JLabel("Result:");

        tf1 = new JTextField();
        tf2 = new JTextField();
        tf3 = new JTextField();
        tf3.setEditable(false); // Result box should not be editable

        btnConcat = new JButton("Concatenate");
        btnReverse = new JButton("Reverse");

        // Add action listeners
        btnConcat.addActionListener(this);
        btnReverse.addActionListener(this);

        // Add components to frame
        add(lbl1);
        add(tf1);
        add(lbl2);
```

```

        add(tf2);
        add(lbl3);
        add(tf3);
        add(btnConcat);
        add(btnReverse);

        setVisible(true);
    }

    // Handle button clicks
    public void actionPerformed(ActionEvent e) {
        String str1 = tf1.getText();
        String str2 = tf2.getText();

        if (e.getSource() == btnConcat) {
            tf3.setText(str1 + str2); // Concatenate
        } else if (e.getSource() == btnReverse) {
            String concat = str1 + str2;
            String reversed = new StringBuilder(concat).reverse().toString();
            tf3.setText(reversed); // Reverse
        }
    }

    public static void main(String[] args) {
        new StringManipulation();
    }
}

```

## Slip 18.1

```

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

public class BorderLayoutDemo extends JFrame {

    public BorderLayoutDemo() {
        // Set frame title and size
        setTitle("BorderLayout Example");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Set BorderLayout for the frame
        setLayout(new BorderLayout(10, 10)); // gaps between components

        // Create buttons for each region
        JButton btnNorth = new JButton("North");
        JButton btnSouth = new JButton("South");
    }
}

```

```

        JButton btnEast = new JButton("East");
        JButton btnWest = new JButton("West");
        JButton btnCenter = new JButton("Center");

        // Add buttons to different regions
        add(btnNorth, BorderLayout.NORTH);
        add(btnSouth, BorderLayout.SOUTH);
        add(btnEast, BorderLayout.EAST);
        add(btnWest, BorderLayout.WEST);
        add(btnCenter, BorderLayout.CENTER);

        // Make the frame visible
        setVisible(true);
    }

    public static void main(String[] args) {
        new BorderLayoutDemo();
    }
}

```

## Output

```

-----
|   North   |
-----
| West |  Center  | East |
-----
|   South   |
-----

```

## Slip 18.2

```

import java.util.Scanner;
import java.util.Arrays;

// Define the CricketPlayer class
class CricketPlayer {
    String name;
    int no_of_innings;
    int no_of_times_notout;
    int total_runs;
    double bat_avg;

    // Constructor

```

```

    public CricketPlayer(String name, int no_of_innings, int no_of_times_notout, int
total_runs) {
        this.name = name;
        this.no_of_innings = no_of_innings;
        this.no_of_times_notout = no_of_times_notout;
        this.total_runs = total_runs;
        this.bat_avg = 0.0; // Initially 0, will be calculated later
    }

    // Static method to calculate batting average
    public static void avg(CricketPlayer player) {
        if (player.no_of_innings - player.no_of_times_notout != 0) {
            player.bat_avg = (double) player.total_runs / (player.no_of_innings -
player.no_of_times_notout);
        } else {
            player.bat_avg = player.total_runs; // Avoid division by zero
        }
    }

    // Static method to sort players by batting average in descending order
    public static void sort(CricketPlayer[] players) {
        Arrays.sort(players, (p1, p2) -> Double.compare(p2.bat_avg, p1.bat_avg));
    }

    // Method to display player details
    public void display() {
        System.out.println("Name: " + name);
        System.out.println("Innings: " + no_of_innings);
        System.out.println("Not Out: " + no_of_times_notout);
        System.out.println("Total Runs: " + total_runs);
        System.out.println("Batting Average: " + String.format("%.2f", bat_avg));
        System.out.println("-----");
    }
}

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

        System.out.print("Enter number of players: ");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline

        CricketPlayer[] players = new CricketPlayer[n];

        // Input player details
        for (int i = 0; i < n; i++) {

```



```

        System.out.println("Enter details for player " + (i + 1));
        System.out.print("Name: ");
        String name = sc.nextLine();
        System.out.print("Number of Innings: ");
        int innings = sc.nextInt();
        System.out.print("Number of Times Not Out: ");
        int notOut = sc.nextInt();
        System.out.print("Total Runs: ");
        int runs = sc.nextInt();
        sc.nextLine(); // Consume newline

        players[i] = new CricketPlayer(name, innings, notOut, runs);
        CricketPlayer.avg(players[i]); // Calculate batting average
    }

    // Sort players by batting average
    CricketPlayer.sort(players);

    // Display sorted player details
    System.out.println("\nPlayers sorted by batting average (highest to lowest):");
    for (CricketPlayer player : players) {
        player.display();
    }

    sc.close();
}
}

```

## Output

Players sorted by batting average (highest to lowest):

Name: Kohli

Innings: 12

Not Out: 3

Total Runs: 600

Batting Average: 66.67

-----

Name: Virat

Innings: 10

Not Out: 2

Total Runs: 500

Batting Average: 62.50

-----

Name: Rohit

Innings: 8

Not Out: 1

Total Runs: 400

Batting Average: 57.14

---

## Slip 19.1

```
import java.util.Scanner;

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

        // Input number of rows and columns
        System.out.print("Enter number of rows: ");
        int rows = sc.nextInt();
        System.out.print("Enter number of columns: ");
        int cols = sc.nextInt();

        if (rows != cols) {
            System.out.println("Matrix must be square to have a diagonal!");
            return;
        }

        int[][] matrix = new int[rows][cols];

        // Input array elements
        System.out.println("Enter elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }

        // Calculate sum of diagonal elements
        int sum = 0;
        for (int i = 0; i < rows; i++) {
            sum += matrix[i][i]; // Main diagonal: row index = column index
        }

        // Display sum
        System.out.println("Sum of diagonal elements: " + sum);

        sc.close();
    }
}
```

# Output

## Input:

```
Enter number of rows: 3
Enter number of columns: 3
Enter elements of the matrix:
1 2 3
4 5 6
7 8 9
```

## Output:

```
Sum of diagonal elements: 15
```

# Slip 19.2

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

public class SubjectComboBox extends JFrame implements ActionListener {
    JComboBox<String> subjectCombo;
    JTextField selectedSubjectField;

    public SubjectComboBox() {
        // Frame settings
        setTitle("T.Y.B.Sc. Computer Science Subjects");
        setSize(400, 150);
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Subject list
        String[] subjects = {
            "Data Mining",
            "Artificial Intelligence",
            "Software Engineering",
            "Computer Networks",
            "Database Management System",
            "Operating Systems"
        };

        // Create combo box
        subjectCombo = new JComboBox<>(subjects);
        subjectCombo.addActionListener(this); // Handle selection
    }
}
```

```

// Create text field
selectedSubjectField = new JTextField(20);
selectedSubjectField.setEditable(false);

// Add components to frame
add(new JLabel("Select Subject:"));
add(subjectCombo);
add(new JLabel("Selected Subject:"));
add(selectedSubjectField);

setVisible(true);
}

// Handle combo box selection
public void actionPerformed(ActionEvent e) {
    String selectedSubject = (String) subjectCombo.getSelectedItem();
    selectedSubjectField.setText(selectedSubject);
}

public static void main(String[] args) {
    new SubjectComboBox();
}
}

```

## Slip 20.1

```

// Base class Continent
class Continent {
    String continentName;

    public void setContinent(String name) {
        this.continentName = name;
    }

    public void displayContinent() {
        System.out.println("Continent: " + continentName);
    }
}

// Country class inherits from Continent
class Country extends Continent {
    String countryName;

    public void setCountry(String name) {
        this.countryName = name;
    }
}

```

```
        public void displayCountry() {  
            System.out.println("Country: " + countryName);  
        }  
    }  
}
```

// State class inherits from Country

```
class State extends Country {  
    String stateName;  
  
    public void setState(String name) {  
        this.stateName = name;  
    }  
  
    public void displayState() {  
        System.out.println("State: " + stateName);  
    }  
}
```

// Place class inherits from State

```
class Place extends State {  
    String placeName;  
  
    public void setPlace(String name) {  
        this.placeName = name;  
    }  
  
    public void displayPlace() {  
        System.out.println("Place: " + placeName);  
    }  
}
```

// Main class

```
public class MultilevelInheritanceDemo {  
    public static void main(String[] args) {  
        Place p = new Place();  
  
        // Set details  
        p.setContinent("Asia");  
        p.setCountry("India");  
        p.setState("Maharashtra");  
        p.setPlace("Pune");  
  
        // Display details  
        System.out.println("Displaying Place Details:");  
        p.displayPlace();  
        p.displayState();  
        p.displayCountry();  
    }  
}
```

```
        p.displayContinent();
    }
}
```

## Output

**Displaying Place Details:**

**Place:** Pune

**State:** Maharashtra

**Country:** India

**Continent:** Asia

## Slip 20.2

### . Package: **Operation**

Create a folder named **Operation** (this will be your package).

#### **Addition.java**

```
package Operation;

public class Addition {

    // Method to add two integers
    public int add(int a, int b) {
        return a + b;
    }

    // Method to subtract two float values
    public float subtract(float a, float b) {
        return a - b;
    }
}
```

#### **Maximum.java**

```
package Operation;

public class Maximum {

    // Method to return the maximum of two integers
    public int max(int a, int b) {
```

```
        if(a > b) {
            return a;
        } else {
            return b;
        }
    }
}
```

---

## 2. Main Class to Test the Package

```
import Operation.Addition;
import Operation.Maximum;

public class TestOperation {
    public static void main(String[] args) {
        // Create objects
        Addition addObj = new Addition();
        Maximum maxObj = new Maximum();

        // Test Addition methods
        int sum = addObj.add(10, 20);
        float difference = addObj.subtract(15.5f, 5.5f);

        System.out.println("Sum of integers: " + sum);
        System.out.println("Difference of floats: " + difference);

        // Test Maximum method
        int maximum = maxObj.max(10, 25);
        System.out.println("Maximum of integers: " + maximum);
    }
}
```

If you run the `TestOperation` class provided above, the output will be:

```
Sum of integers: 30
Difference of floats: 10.0
Maximum of integers: 25
```

# Slip 21.1

## 1. Define the User-Defined Exception

```
// InvalidDateException.java
```

```
public class InvalidDateException extends Exception {
    public InvalidDateException(String message) {
        super(message);
    }
}
```

---

## 2. Define the **MyDate** Class

```
// MyDate.java
import java.util.Scanner;

public class MyDate {
    private int day;
    private int month;
    private int year;

    // Method to accept date
    public void acceptDate() throws InvalidDateException {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter day (dd): ");
        int d = sc.nextInt();
        System.out.print("Enter month (mm): ");
        int m = sc.nextInt();
        System.out.print("Enter year (yyyy): ");
        int y = sc.nextInt();

        if (!isValidDate(d, m, y)) {
            throw new InvalidDateException("Invalid date entered: " + d + "/" + m + "/" + y);
        }

        this.day = d;
        this.month = m;
        this.year = y;
    }

    // Method to display date
    public void displayDate() {
        System.out.println("Date: " + String.format("%02d/%02d/%04d", day, month, year));
    }

    // Method to check if date is valid
    private boolean isValidDate(int d, int m, int y) {
        if (y < 1 || m < 1 || m > 12 || d < 1) return false;
    }
}
```



```

int[] daysInMonth = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

// Check for leap year
if (m == 2 && isLeapYear(y)) {
    return d <= 29;
}

return d <= daysInMonth[m - 1];
}

// Check for leap year
private boolean isLeapYear(int y) {
    return (y % 4 == 0 && y % 100 != 0) || (y % 400 == 0);
}
}

```

---

### 3. Main Class to Test **MyDate**

```

// TestMyDate.java
public class TestMyDate {
    public static void main(String[] args) {
        MyDate date = new MyDate();

        try {
            date.acceptDate(); // Accept date from user
            date.displayDate(); // Display the date
        } catch (InvalidDateException e) {
            System.out.println(e.getMessage());
        }
    }
}

```

### 1. Valid Date Input

User Input:

Enter day (dd): 29

Enter month (mm): 2

Enter year (yyyy): 2024

Output:

Date: 29/02/2024

*Explanation:* 2024 is a leap year, so 29th February is valid.

---

## 2. Invalid Date Input

User Input:

Enter day (dd): 31

Enter month (mm): 4

Enter year (yyyy): 2023

Output:

Invalid date entered: 31/4/2023

# Slip 21.2

## Employee.java

```
public class Employee {
    // Instance variables
    private int id;
    private String name;
    private String deptName;
    private double salary;

    // Static variable to keep count of objects
    private static int objectCount = 0;

    // Default constructor
    public Employee() {
        this.id = 0;
        this.name = "Unknown";
        this.deptName = "None";
        this.salary = 0.0;
        objectCount++;
    }

    // Parameterized constructor
    public Employee(int id, String name, String deptName, double salary) {
```

```
        this.id = id;
        this.name = name;
        this.deptName = deptName;
        this.salary = salary;
        objectCount++;
    }

    // Static method to get object count
    public static int getObjectCount() {
        return objectCount;
    }

    // Method to display employee details
    public void display() {
        System.out.println("Employee ID: " + this.id);
        System.out.println("Name: " + this.name);
        System.out.println("Department: " + this.deptName);
        System.out.println("Salary: " + this.salary);
        System.out.println("-----");
    }
}
```

---

### **TestEmployee.java**

```
public class TestEmployee {
    public static void main(String[] args) {
        // Creating employee objects using parameterized constructor
        Employee e1 = new Employee(101, "Alice", "HR", 50000);
        System.out.println("Object count: " +
Employee.getObjectCount());
        e1.display();

        Employee e2 = new Employee(102, "Bob", "IT", 60000);
        System.out.println("Object count: " +
Employee.getObjectCount());
        e2.display();

        Employee e3 = new Employee(103, "Charlie", "Finance",
55000);
        System.out.println("Object count: " +
Employee.getObjectCount());
    }
}
```

```
        e3.display();
    }
}
```

---

### ✓ Output

```
Object count: 1
Employee ID: 101
Name: Alice
Department: HR
Salary: 50000.0
```

-----

```
Object count: 2
Employee ID: 102
Name: Bob
Department: IT
Salary: 60000.0
```

-----

```
Object count: 3
Employee ID: 103
Name: Charlie
Department: Finance
Salary: 55000.0
```

-----

## Slip 22.1

### Shape.java

```
// Abstract class Shape
abstract class Shape {
    protected double a, b; // Two integers/doubles to store
    dimensions

    // Constructor
    public Shape(double a, double b) {
        this.a = a;
        this.b = b;
    }

    // Abstract method to print area
```

```
        abstract void printArea();
    }
}
```

---

### Rectangle.java

```
// Rectangle class extending Shape
class Rectangle extends Shape {
    public Rectangle(double length, double width) {
        super(length, width);
    }

    // Overriding printArea method
    @Override
    void printArea() {
        double area = a * b;
        System.out.println("Area of Rectangle: " + area);
    }
}
```

---

### Triangle.java

```
// Triangle class extending Shape
class Triangle extends Shape {
    public Triangle(double base, double height) {
        super(base, height);
    }

    // Overriding printArea method
    @Override
    void printArea() {
        double area = 0.5 * a * b;
        System.out.println("Area of Triangle: " + area);
    }
}
```

---

### Circle.java

```
// Circle class extending Shape
class Circle extends Shape {
```

```
public Circle(double radius) {
    super(radius, 0); // Only radius is needed, b is unused
}

// Overriding printArea method
@Override
void printArea() {
    double area = Math.PI * a * a;
    System.out.println("Area of Circle: " + area);
}
}
```

---

### TestShape.java

```
public class TestShape {
    public static void main(String[] args) {
        // Create objects for each shape
        Shape rect = new Rectangle(10, 5);
        Shape tri = new Triangle(8, 6);
        Shape circ = new Circle(7);

        // Print areas
        rect.printArea();
        tri.printArea();
        circ.printArea();
    }
}
```

---

### ✓ Output

```
Area of Rectangle: 50.0
Area of Triangle: 24.0
Area of Circle: 153.93804002589985
```

---

## Slip 22.2

### MouseEventDemo.java

```
import javax.swing.*;
```

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

public class MouseEventDemo extends JFrame {

    private String eventName = ""; // To store the name of the
current mouse event

    public MouseEventDemo() {
        setTitle("Mouse Event Demo");
        setSize(500, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Add MouseListener and MouseMotionListener using adapters
        addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                eventName = "Mouse Clicked";
                repaint();
            }

            @Override
            public void mousePressed(MouseEvent e) {
                eventName = "Mouse Pressed";
                repaint();
            }

            @Override
            public void mouseReleased(MouseEvent e) {
                eventName = "Mouse Released";
                repaint();
            }

            @Override
            public void mouseEntered(MouseEvent e) {
                eventName = "Mouse Entered";
                repaint();
            }

            @Override
```

```

        public void mouseExited(MouseEvent e) {
            eventName = "Mouse Exited";
            repaint();
        }
    });

    addMouseListener(new MouseMotionAdapter() {
        @Override
        public void mouseDragged(MouseEvent e) {
            eventName = "Mouse Dragged";
            repaint();
        }

        @Override
        public void mouseMoved(MouseEvent e) {
            eventName = "Mouse Moved";
            repaint();
        }
    });
}

// Paint method to display event name
@Override
public void paint(Graphics g) {
    super.paint(g);
    g.setColor(Color.RED);
    g.setFont(new Font("Arial", Font.BOLD, 24));

    // Draw the event name at the center of the window
    FontMetrics fm = g.getFontMetrics();
    int x = (getWidth() - fm.stringWidth(eventName)) / 2;
    int y = (getHeight() / 2);
    g.drawString(eventName, x, y);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        MouseEventDemo frame = new MouseEventDemo();
        frame.setVisible(true);
    });
}

```



```
}
```

---

## Slip 23.1

### **MyNumber.java**

```
public class MyNumber {

    private int number; // Private data member

    // Default constructor
    public MyNumber() {
        this.number = 0;
    }

    // Parameterized constructor
    public MyNumber(int number) {
        this.number = number;
    }

    // Method to check if the number is negative
    public boolean isNegative() {
        return number < 0;
    }

    // Method to check if the number is positive
    public boolean isPositive() {
        return number > 0;
    }

    // Method to check if the number is zero
    public boolean isZero() {
        return number == 0;
    }

    // Method to check if the number is odd
    public boolean isOdd() {
        return number % 2 != 0;
    }

    // Method to check if the number is even
```

```

    public boolean isEven() {
        return number % 2 == 0;
    }

    // Method to display all properties
    public void displayProperties() {
        System.out.println("Number: " + number);
        System.out.println("Positive? " + isPositive());
        System.out.println("Negative? " + isNegative());
        System.out.println("Zero? " + isZero());
        System.out.println("Odd? " + isOdd());
        System.out.println("Even? " + isEven());
    }

    // Main method
    public static void main(String[] args) {
        int value = 0;

        if (args.length > 0) {
            try {
                value = Integer.parseInt(args[0]); // Parse
command-line argument
            } catch (NumberFormatException e) {
                System.out.println("Invalid input. Using default
value 0.");
            }
        }

        MyNumber num = new MyNumber(value); // Create object using
parameterized constructor
        num.displayProperties();              // Display properties
    }
}

```

## ample Output

### Command:

```
java MyNumber 15
```

### Output:

Number: 15  
Positive? true  
Negative? false  
Zero? false  
Odd? true  
Even? false

## Slip 23.2

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

public class CurrencyConverter extends Frame implements ActionListener {
    Label l1, l2, l3;
    TextField t1, t2, t3;
    Button convert;

    public CurrencyConverter() {
        // Labels
        l1 = new Label("Singapore Dollars");
        l2 = new Label("US Dollars");
        l3 = new Label("Euros");

        // TextFields
        t1 = new TextField();
        t2 = new TextField();
        t3 = new TextField();

        // Button
        convert = new Button("Convert");
        convert.addActionListener(this);

        // Layout
        setLayout(new GridLayout(4, 2, 10, 10));
        add(l1); add(t1);
        add(l2); add(t2);
        add(l3); add(t3);
        add(new Label("")); add(convert);

        // Frame settings
        setTitle("Currency Converter");
        setSize(300, 200);
        setVisible(true);

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
```

```

        System.exit(0);
    }
});
}

public void actionPerformed(ActionEvent e) {
    try {
        double sgd = Double.parseDouble(t1.getText());

        // Conversion
        double usd = sgd / 1.41;    // SGD → USD
        double euro = sgd * 0.65;    // SGD → EUR

        // Show result with 2 decimal places
        t2.setText(String.format("%.2f", usd));
        t3.setText(String.format("%.2f", euro));

    } catch (Exception ex) {
        t2.setText("Error");
        t3.setText("Error");
    }
}

public static void main(String[] args) {
    new CurrencyConverter();
}
}

```

Output

```

+-----+
| Currency Converter      |
| Singapore Dollars [ 100 ] |
| US Dollars      [ 70.92 ] |
| Euros          [ 65.00 ] |
|                  [Convert] |
+-----+

```

## Slip 24.1

```

// Abstract class Bank
abstract class Bank {
    // Abstract method
    abstract int getBalance();
}

```

```

// Subclass BankA
class BankA extends Bank {

```

```

private int balance = 100;

@Override
int getBalance() {
    return balance;
}
}

// Subclass BankB
class BankB extends Bank {
    private int balance = 150;

    @Override
    int getBalance() {
        return balance;
    }
}

// Subclass BankC
class BankC extends Bank {
    private int balance = 200;

    @Override
    int getBalance() {
        return balance;
    }
}

// Main class to test
public class Main {
    public static void main(String[] args) {
        Bank bankA = new BankA();
        Bank bankB = new BankB();
        Bank bankC = new BankC();

        System.out.println("Balance in Bank A: Rs." + bankA.getBalance());
        System.out.println("Balance in Bank B: Rs." + bankB.getBalance());
        System.out.println("Balance in Bank C: Rs." + bankC.getBalance());
    }
}

```

### ✓ Output:

```

Balance in Bank A: Rs.100
Balance in Bank B: Rs.150
Balance in Bank C: Rs.200

```

# Slip 24.2

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

public class ConcentricCircles extends Frame {
    int x = -100, y = -100; // Initial position (outside screen so nothing shows at start)

    ConcentricCircles() {
        setSize(400, 400);
        setTitle("Concentric Circles");
        setVisible(true);

        // Handle window close
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });

        // Mouse listener to get click position
        addMouseListener(new MouseAdapter() {
            public void mouseClicked(MouseEvent me) {
                x = me.getX();
                y = me.getY();
                repaint();
            }
        });
    }

    public void paint(Graphics g) {
        // Draw 3 concentric circles centered at (x, y)
        g.setColor(Color.BLACK);
        g.drawOval(x - 30, y - 30, 60, 60); // outer
        g.drawOval(x - 60, y - 60, 120, 120); // middle
        g.drawOval(x - 90, y - 90, 180, 180); // largest
    }

    public static void main(String[] args) {
        new ConcentricCircles();
    }
}
```

Output

 <- outer circle



## Slip 25.1

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

// Student class
class Student {
    int rollno;
    String name;
    String studentClass;
    double per;

    // Constructor
    Student(int rollno, String name, String studentClass, double per) {
        this.rollno = rollno;
        this.name = name;
        this.studentClass = studentClass;
        this.per = per;
    }

    // Method to display student details
    void display() {
        System.out.println("\n--- Student Details ---");
        System.out.println("Roll No: " + rollno);
        System.out.println("Name: " + name);
        System.out.println("Class: " + studentClass);
        System.out.println("Percentage: " + per + "%");
    }
}

// Main class
public class Main {
    public static void main(String[] args) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter Roll No: ");
        int rollno = Integer.parseInt(br.readLine());

        System.out.print("Enter Name: ");
        String name = br.readLine();

        System.out.print("Enter Class: ");
        String studentClass = br.readLine();
    }
}
```

```

        System.out.print("Enter Percentage: ");
        double per = Double.parseDouble(br.readLine());

        // Create Student object
        Student s = new Student(rollno, name, studentClass, per);

        // Display student details
        s.display();
    }
}

```

### ✓ Sample Output:

```

Enter Roll No: 101
Enter Name: Aditya
Enter Class: BSc
Enter Percentage: 85.5

--- Student Details ---
Roll No: 101
Name: Aditya
Class: BSc
Percentage: 85.5%

```

## Slip 25.2

```

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

class StudentForm extends JFrame implements ActionListener {
    JTextField nameField;
    JRadioButton fy, sy, ty;
    JCheckBox music, dance, sports;
    JTextArea output;
    JButton submit;
    ButtonGroup classGroup;

    StudentForm() {
        setTitle("Student Form");
        setSize(400, 400);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
    }
}

```



```

// Name
add(new JLabel("Your Name:"));
nameField = new JTextField(15);
add(nameField);

// Class
add(new JLabel("Your Class:"));
fy = new JRadioButton("FY");
sy = new JRadioButton("SY");
ty = new JRadioButton("TY");
classGroup = new ButtonGroup();
classGroup.add(fy);
classGroup.add(sy);
classGroup.add(ty);
add(fy); add(sy); add(ty);

// Hobbies
add(new JLabel("Your Hobbies:"));
music = new JCheckBox("Music");
dance = new JCheckBox("Dance");
sports = new JCheckBox("Sports");
add(music); add(dance); add(sports);

// Submit button
submit = new JButton("Submit");
add(submit);
submit.addActionListener(this);

// Output area
output = new JTextArea(5, 30);
add(output);

setVisible(true);
}

public void actionPerformed(ActionEvent e) {
    String name = nameField.getText();
    String cls = "";
    if (fy.isSelected()) cls = "FY";
    else if (sy.isSelected()) cls = "SY";
    else if (ty.isSelected()) cls = "TY";

    String hobbies = "";
    if (music.isSelected()) hobbies += "Music ";
    if (dance.isSelected()) hobbies += "Dance ";
    if (sports.isSelected()) hobbies += "Sports ";

    output.setText("Name: " + name + "\nClass: " + cls + "\nHobbies: " + hobbies);
}

```

```

    }

    public static void main(String[] args) {
        new StudentForm();
    }
}

```

## Slip 26.1

```

// Item class
class Item {
    int item_number;
    String item_name;
    double item_price;

    // Static member to keep count of objects
    static int count = 0;

    // Default constructor
    Item() {
        item_number = 0;
        item_name = "Unknown";
        item_price = 0.0;
        count++; // Increment object count
    }

    // Parameterized constructor
    Item(int item_number, String item_name, double item_price) {
        this.item_number = item_number;
        this.item_name = item_name;
        this.item_price = item_price;
        count++; // Increment object count
    }

    // Static method to get object count
    static int getCount() {
        return count;
    }

    // Method to display object details
    void display() {
        System.out.println("Item Number: " + item_number);
        System.out.println("Item Name: " + item_name);
        System.out.println("Item Price: Rs." + item_price);
        System.out.println("-----");
    }
}

```

```
// Main class
public class Main {
    public static void main(String[] args) {
        // Create objects using parameterized constructor
        Item item1 = new Item(101, "Laptop", 50000);
        item1.display();
        System.out.println("Objects created: " + Item.getCount() + "\n");

        Item item2 = new Item(102, "Mouse", 500);
        item2.display();
        System.out.println("Objects created: " + Item.getCount() + "\n");

        Item item3 = new Item(103, "Keyboard", 1500);
        item3.display();
        System.out.println("Objects created: " + Item.getCount() + "\n");
    }
}
```

### ✓ Sample Output:

```
Item Number: 101
Item Name: Laptop
Item Price: Rs.50000.0
-----
Objects created: 1

Item Number: 102
Item Name: Mouse
Item Price: Rs.500.0
-----
Objects created: 2

Item Number: 103
Item Name: Keyboard
Item Price: Rs.1500.0
-----
Objects created: 3
```

## Slip 26.2

```
import java.io.*;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
```

```

import java.time.temporal.ChronoUnit;
import java.util.*;

// Donor class implementing Serializable
class Donor implements Serializable {
    String name;
    int age;
    String address;
    String contactNumber;
    String bloodGroup;
    LocalDate lastDonationDate;

    // Constructor
    Donor(String name, int age, String address, String contactNumber, String bloodGroup,
LocalDate lastDonationDate) {
        this.name = name;
        this.age = age;
        this.address = address;
        this.contactNumber = contactNumber;
        this.bloodGroup = bloodGroup;
        this.lastDonationDate = lastDonationDate;
    }

    // Display method
    void display() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Address: " + address);
        System.out.println("Contact Number: " + contactNumber);
        System.out.println("Blood Group: " + bloodGroup);
        System.out.println("Last Donation Date: " + lastDonationDate);
        System.out.println("-----");
    }
}

// Main class
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd-MM-yyyy");

        System.out.print("Enter number of donors: ");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline

        List<Donor> donors = new ArrayList<>();

        // Read donor details from user

```

```

for (int i = 0; i < n; i++) {
    System.out.println("\nEnter details for Donor " + (i + 1));
    System.out.print("Name: ");
    String name = sc.nextLine();

    System.out.print("Age: ");
    int age = sc.nextInt();
    sc.nextLine();

    System.out.print("Address: ");
    String address = sc.nextLine();

    System.out.print("Contact Number: ");
    String contactNumber = sc.nextLine();

    System.out.print("Blood Group: ");
    String bloodGroup = sc.nextLine();

    System.out.print("Last Donation Date (dd-MM-yyyy): ");
    String dateStr = sc.nextLine();
    LocalDate lastDonationDate = LocalDate.parse(dateStr, formatter);

    donors.add(new Donor(name, age, address, contactNumber, bloodGroup,
lastDonationDate));
}

// Write donors to file
try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream("donors.dat"))) {
    oos.writeObject(donors);
    System.out.println("\nDonor details saved to file successfully!");
} catch (IOException e) {
    System.out.println("Error writing to file: " + e.getMessage());
}

// Read donors from file and display filtered donors
System.out.println("\nDonors with Blood Group A+ve and not donated in last 6
months:\n");
try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream("donors.dat"))) {
    List<Donor> readDonors = (List<Donor>) ois.readObject();
    LocalDate sixMonthsAgo = LocalDate.now().minus(6, ChronoUnit.MONTHS);

    for (Donor d : readDonors) {
        if (d.bloodGroup.equalsIgnoreCase("A+ve") &&
d.lastDonationDate.isBefore(sixMonthsAgo)) {
            d.display();
        }
    }
}

```

```
    }  
    } catch (IOException | ClassNotFoundException e) {  
        System.out.println("Error reading from file: " + e.getMessage());  
    }  
  
    sc.close();  
}  
}
```

### Input (example):

Enter number of donors: 3

Enter details for Donor 1

Name: Aditya

Age: 25

Address: Pune

Contact Number: 9876543210

Blood Group: A+ve

Last Donation Date (dd-MM-yyyy): 01-01-2025

Enter details for Donor 2

Name: Priya

Age: 30

Address: Pune

Contact Number: 9123456780

Blood Group: B+ve

Last Donation Date (dd-MM-yyyy): 01-03-2025

Enter details for Donor 3

Name: Rahul

Age: 28

Address: Pune

Contact Number: 9988776655

Blood Group: A+ve

Last Donation Date (dd-MM-yyyy): 01-01-2024

---

### Output:

Donor details saved to file successfully!

Donors with Blood Group A+ve and not donated in last 6 months:

Name: Rahul  
Age: 28  
Address: Pune  
Contact Number: 9988776655  
Blood Group: A+ve  
Last Donation Date: 2024-01-01  
-----

## Slip 27.1

```
// Employee class
class Employee {
    String name;
    int empId;
    double basicSalary;

    // Constructor
    Employee(String name, int empId, double basicSalary) {
        this.name = name;
        this.empId = empId;
        this.basicSalary = basicSalary;
    }

    // Method to get salary
    double getSalary() {
        return basicSalary;
    }

    // Method to display employee details
    void display() {
        System.out.println("Employee ID: " + empId);
        System.out.println("Name: " + name);
        System.out.println("Salary: Rs." + getSalary());
        System.out.println("-----");
    }
}

// Manager class extending Employee
class Manager extends Employee {
    double travelAllowance;
    double houseRentAllowance;

    // Constructor
    Manager(String name, int empId, double basicSalary, double travelAllowance, double
houseRentAllowance) {
```

```

        super(name, empId, basicSalary);
        this.travelAllowance = travelAllowance;
        this.houseRentAllowance = houseRentAllowance;
    }

    // Overriding getSalary() method
    @Override
    double getSalary() {
        return basicSalary + travelAllowance + houseRentAllowance;
    }
}

// Main class
public class Main {
    public static void main(String[] args) {
        // Employee object
        Employee emp = new Employee("Aditya", 101, 50000);
        emp.display();

        // Manager object
        Manager mgr = new Manager("Priya", 102, 60000, 10000, 15000);
        mgr.display();
    }
}

```

### ✓ Sample Output:

```

Employee ID: 101
Name: Aditya
Salary: Rs.50000.0
-----

```

```

Employee ID: 102
Name: Priya
Salary: Rs.85000.0
-----

```

## Slip 27.2

```

import java.io.File;
import java.util.Scanner;

public class FileDirectoryOperations {
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Usage: java FileDirectoryOperations <path>");
            return;
        }
    }
}

```



```

}

String path = args[0];
File f = new File(path);
Scanner sc = new Scanner(System.in);

if (!f.exists()) {
    System.out.println("The path does not exist.");
    sc.close();
    return;
}

if (f.isDirectory()) {
    // Directory operations
    File[] files = f.listFiles((dir, name) -> name.endsWith(".txt"));
    if (files == null || files.length == 0) {
        System.out.println("No text files found in the directory.");
    } else {
        System.out.println("Text files found in the directory:");
        for (File file : files) {
            System.out.println(file.getName());
        }

        System.out.print("Do you want to delete all these text files? (yes/no): ");
        String confirm = sc.nextLine();

        if (confirm.equalsIgnoreCase("yes")) {
            int count = 0;
            for (File file : files) {
                if (file.delete()) {
                    count++;
                }
            }
            System.out.println(count + " text file(s) deleted successfully.");
        } else {
            System.out.println("Delete operation cancelled.");
        }
    }
} else if (f.isFile()) {
    // File operations
    System.out.println("File Details:");
    System.out.println("Name: " + f.getName());
    System.out.println("Absolute Path: " + f.getAbsolutePath());
    System.out.println("Size: " + f.length() + " bytes");
    System.out.println("Readable: " + f.canRead());
    System.out.println("Writable: " + f.canWrite());
    System.out.println("Executable: " + f.canExecute());
    System.out.println("Last Modified: " + new java.util.Date(f.lastModified()));
}

```

```

    } else {
        System.out.println("The path is neither a file nor a directory.");
    }

    sc.close();
}
}

```

### Sample Output 1 (Directory):

```

Text files found in the directory:
file1.txt
file2.txt
Do you want to delete all these text files? (yes/no): yes
2 text file(s) deleted successfully.

```

### Sample Output 2 (File):

```

File Details:
Name: file1.txt
Absolute Path: C:\Users\Aditya\Documents\TestFolder\file1.txt
Size: 1024 bytes
Readable: true
Writable: true
Executable: false
Last Modified: Mon Sep 29 22:00:00 IST 2025

```

## Slip 28.1

```

import java.io.File;
import java.util.Scanner;

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

        // Read file name from user
        System.out.print("Enter the file name with path: ");
        String fileName = sc.nextLine();

        File file = new File(fileName);

        // Check if file exists
        if (file.exists()) {

```

```
System.out.println("File exists: Yes");

// Check readability
System.out.println("Readable: " + file.canRead());

// Check writability
System.out.println("Writable: " + file.canWrite());

// Check if file or directory
if (file.isFile()) {
    System.out.println("Type: File");
} else if (file.isDirectory()) {
    System.out.println("Type: Directory");
} else {
    System.out.println("Type: Unknown");
}

// File length in bytes
System.out.println("Length: " + file.length() + " bytes");
} else {
    System.out.println("File exists: No");
}

sc.close();
}
}
```

### **Sample Run:**

#### **Input:**

Enter the file name with path: C:\Users\Aditya\Documents\example.txt

#### **Output:**

File exists: Yes  
Readable: true  
Writable: true  
Type: File  
Length: 1024 bytes

#### **If the file does not exist:**

File exists: No

# Slip 28.2

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

public class SwingTemperatureConverter extends JFrame implements ActionListener {
    private JTextField celsiusField, fahrenheitField;

    SwingTemperatureConverter() {
        setTitle("Temperature Converter");
        setSize(300, 150);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new GridLayout(2, 2, 5, 5));

        // Labels
        JLabel celsiusLabel = new JLabel("Celsius:");
        JLabel fahrenheitLabel = new JLabel("Fahrenheit:");

        // TextFields
        celsiusField = new JTextField();
        fahrenheitField = new JTextField();

        add(celsiusLabel);
        add(celsiusField);
        add(fahrenheitLabel);
        add(fahrenheitField);

        // Event handling
        celsiusField.addActionListener(this);
        fahrenheitField.addActionListener(this);

        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            if (e.getSource() == celsiusField) {
                double c = Double.parseDouble(celsiusField.getText());
                double f = (c * 9 / 5) + 32;
                fahrenheitField.setText(String.format("%.2f", f));
            }
            else if (e.getSource() == fahrenheitField) {
                double f = Double.parseDouble(fahrenheitField.getText());
                double c = (f - 32) * 5 / 9;
            }
        }
    }
}
```

```

        celsiusField.setText(String.format("%.2f", c));
    }
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(this, "Please enter a valid number!");
}
}

public static void main(String[] args) {
    new SwingTemperatureConverter();
}
}

```

Output

The GUI looks just like your image:

```

+-----+
| Temperature Converter |
| Celsius:  [37.4  ] |
| Fahrenheit: [99.5  ] |
+-----+

```

## Slip 29.1

```

import java.util.ArrayList;
import java.util.Scanner;

```

// Customer class

```
class Customer {
```

```
    int custNo;
```

```
    String custName;
```

```
    String contactNumber;
```

```
    String custAddr;
```

// Constructor

```
Customer(int custNo, String custName, String contactNumber, String custAddr) {
```

```
    this.custNo = custNo;
```

```
    this.custName = custName;
```

```
    this.contactNumber = contactNumber;
```

```
    this.custAddr = custAddr;
```

```
}
```

// Method to display customer details

```
void display() {
```

```
    System.out.println("Customer Number: " + custNo);
```

```
    System.out.println("Customer Name: " + custName);
```

```
    System.out.println("Contact Number: " + contactNumber);
```

```

        System.out.println("Customer Address: " + custAddr);
        System.out.println("-----");
    }
}

// Main class
public class CustomerSearch {
    // Method to search customer by contact number
    static void searchCustomerByContact(ArrayList<Customer> customers, String
contactNumber) {
        boolean found = false;
        for (Customer c : customers) {
            if (c.contactNumber.equals(contactNumber)) {
                System.out.println("Customer found:");
                c.display();
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println("No customer found with contact number: " + contactNumber);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Customer> customers = new ArrayList<>();

        // Example: Read n customers from user
        System.out.print("Enter number of customers: ");
        int n = sc.nextInt();
        sc.nextLine(); // consume newline

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Customer " + (i + 1));
            System.out.print("Customer Number: ");
            int custNo = sc.nextInt();
            sc.nextLine(); // consume newline

            System.out.print("Customer Name: ");
            String custName = sc.nextLine();

            System.out.print("Contact Number: ");
            String contactNumber = sc.nextLine();

            System.out.print("Customer Address: ");
            String custAddr = sc.nextLine();

```

```
        customers.add(new Customer(custNo, custName, contactNumber, custAddr));
    }

    // Search for a customer by contact number
    System.out.print("\nEnter contact number to search: ");
    String searchContact = sc.nextLine();

    searchCustomerByContact(customers, searchContact);

    sc.close();
}
}
```

### ✓ Sample Run:

#### Input:

Enter number of customers: 2

Enter details for Customer 1

Customer Number: 101

Customer Name: Aditya

Contact Number: 9876543210

Customer Address: Pune

Enter details for Customer 2

Customer Number: 102

Customer Name: Priya

Contact Number: 9123456780

Customer Address: Mumbai

Enter contact number to search: 9876543210

#### Output:

Customer found:

Customer Number: 101

Customer Name: Aditya

Contact Number: 9876543210

Customer Address: Pune

-----

# Slip 29.2

```
import java.util.ArrayList;
import java.util.Scanner;
```

```
// Superclass Vehicle
```

```
class Vehicle {
```

```
    String company;
```

```
    double price;
```

```
    Vehicle(String company, double price) {
```

```
        this.company = company;
```

```
        this.price = price;
```

```
    }
```

```
    void display() {
```

```
        System.out.println("Company: " + company);
```

```
        System.out.println("Price: Rs." + price);
```

```
    }
```

```
}
```

```
// Subclass LightMotorVehicle
```

```
class LightMotorVehicle extends Vehicle {
```

```
    double mileage;
```

```
    LightMotorVehicle(String company, double price, double mileage) {
```

```
        super(company, price);
```

```
        this.mileage = mileage;
```

```
    }
```

```
    @Override
```

```
    void display() {
```

```
        System.out.println("\n--- Light Motor Vehicle ---");
```

```
        super.display();
```

```
        System.out.println("Mileage: " + mileage + " km/l");
```

```
    }
```

```
}
```

```
// Subclass HeavyMotorVehicle
```

```
class HeavyMotorVehicle extends Vehicle {
```

```
    double capacityInTons;
```

```
    HeavyMotorVehicle(String company, double price, double capacityInTons) {
```

```
        super(company, price);
```

```
        this.capacityInTons = capacityInTons;
```

```
    }
```



```

@Override
void display() {
    System.out.println("\n--- Heavy Motor Vehicle ---");
    super.display();
    System.out.println("Capacity: " + capacityInTons + " tons");
}
}

// Main class
public class VehicleInfo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Vehicle> vehicles = new ArrayList<>();

        System.out.print("Enter number of vehicles: ");
        int n = sc.nextInt();
        sc.nextLine(); // consume newline

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Vehicle " + (i + 1));

            System.out.print("Type of vehicle (L for Light, H for Heavy): ");
            String type = sc.nextLine();

            System.out.print("Company: ");
            String company = sc.nextLine();

            System.out.print("Price: ");
            double price = sc.nextDouble();

            if (type.equalsIgnoreCase("L")) {
                System.out.print("Mileage (km/l): ");
                double mileage = sc.nextDouble();
                vehicles.add(new LightMotorVehicle(company, price, mileage));
            } else if (type.equalsIgnoreCase("H")) {
                System.out.print("Capacity in tons: ");
                double capacity = sc.nextDouble();
                vehicles.add(new HeavyMotorVehicle(company, price, capacity));
            } else {
                System.out.println("Invalid vehicle type! Skipping this entry.");
            }

            sc.nextLine(); // consume newline
        }

        // Display all vehicle information
        System.out.println("\n--- Vehicle Information ---");
        for (Vehicle v : vehicles) {

```

```
        v.display();
    }

    sc.close();
}
}
```

### ✓ Sample Run:

#### Input:

Enter number of vehicles: 2

Enter details for Vehicle 1

Type of vehicle (L for Light, H for Heavy): L

Company: Honda

Price: 80000

Mileage (km/l): 18

Enter details for Vehicle 2

Type of vehicle (L for Light, H for Heavy): H

Company: Tata

Price: 500000

Capacity in tons: 10

#### Output:

--- Vehicle Information ---

--- Light Motor Vehicle ---

Company: Honda

Price: Rs.80000.0

Mileage: 18.0 km/l

--- Heavy Motor Vehicle ---

Company: Tata

Price: Rs.500000.0

Capacity: 10.0 tons

## Slip 30.1

```
import java.util.Scanner;
```

```

// Person class
class Person {
    String personName;
    String aadharNo;
    String panNo;

    // Constructor using 'this' keyword
    Person(String personName, String aadharNo, String panNo) {
        this.personName = personName;
        this.aadharNo = aadharNo;
        this.panNo = panNo;
    }

    // Method to display person information
    void display() {
        System.out.println("\nPerson Name: " + this.personName);
        System.out.println("Aadhar No: " + this.aadharNo);
        System.out.println("PAN No: " + this.panNo);
        System.out.println("-----");
    }
}

// Main class
public class PersonInfo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Person[] persons = new Person[5];

        // Accept information for 5 persons
        for (int i = 0; i < 5; i++) {
            System.out.println("\nEnter details for Person " + (i + 1));

            System.out.print("Person Name: ");
            String name = sc.nextLine();

            System.out.print("Aadhar No: ");
            String aadhar = sc.nextLine();

            System.out.print("PAN No: ");
            String pan = sc.nextLine();

            // Create Person object using 'this' keyword in constructor
            persons[i] = new Person(name, aadhar, pan);
        }

        // Display information of all persons
        System.out.println("\n--- Person Details ---");
    }
}

```

```

        for (Person p : persons) {
            p.display();
        }

        sc.close();
    }
}

```

### ✓ Sample Run:

#### Input:

```

Enter details for Person 1
Person Name: Aditya
Aadhar No: 123456789012
PAN No: ABCDE1234F

```

```

Enter details for Person 2
Person Name: Priya
Aadhar No: 987654321098
PAN No: XYZAB5678C

```

*(continues for 5 persons)*

#### Output:

```

--- Person Details ---

```

```

Person Name: Aditya
Aadhar No: 123456789012
PAN No: ABCDE1234F

```

```

-----

```

```

Person Name: Priya
Aadhar No: 987654321098
PAN No: XYZAB5678C

```

## Slip 30.2

```

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

```

```

public class IntegerDivisionGUI extends JFrame implements ActionListener {

```

```

// GUI Components
JTextField number1Field, number2Field, resultField;
JButton divideButton;

public IntegerDivisionGUI() {
    // Set up frame
    setTitle("Integer Division");
    setSize(400, 200);
    setLayout(new GridLayout(4, 2, 10, 10));
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // Labels
    add(new JLabel("Number 1:"));
    number1Field = new JTextField();
    add(number1Field);

    add(new JLabel("Number 2:"));
    number2Field = new JTextField();
    add(number2Field);

    add(new JLabel("Result:"));
    resultField = new JTextField();
    resultField.setEditable(false);
    add(resultField);

    // Divide button
    divideButton = new JButton("Divide");
    divideButton.addActionListener(this);
    add(divideButton);

    setVisible(true);
}

// Handle button click
@Override
public void actionPerformed(ActionEvent e) {
    try {
        int num1 = Integer.parseInt(number1Field.getText());
        int num2 = Integer.parseInt(number2Field.getText());

        // Perform division
        int result = num1 / num2;
        resultField.setText(String.valueOf(result));

    } catch (NumberFormatException nfe) {
        JOptionPane.showMessageDialog(this,
            "Invalid input! Please enter integers only.",

```

```
        "Number Format Exception",
        JOptionPane.ERROR_MESSAGE);
    } catch (ArithmeticException ae) {
        JOptionPane.showMessageDialog(this,
            "Division by zero is not allowed!",
            "Arithmetic Exception",
            JOptionPane.ERROR_MESSAGE);
    }
}

public static void main(String[] args) {
    // Run GUI in Event Dispatch Thread
    SwingUtilities.invokeLater(() -> new IntegerDivisionGUI());
}
}
```