**ClassRoom Program Assignment – 9**

1. **Problem Statement:**

Create a Java program that simulates a simple calculator. The calculator should perform basic arithmetic operations: addition, subtraction, multiplication, and division. Your task is to:

* Accept two integers and an operator from the user.
* Use exception handling to catch and manage the following runtime exceptions:
  + ArithmeticException (e.g., division by zero)
  + InputMismatchException (e.g., invalid input type)
  + NullPointerException (simulate a scenario where this might occur)

**Requirements:**

* Use Scanner for input.
* Wrap risky operations in try-catch blocks.
* Print meaningful error messages when exceptions occur.
* Use a finally block to print a closing message.

1. **Problem Statement:**

Design a Java program to model a **University** system. The outer class should be University, and it should contain a **non-static nested class** called Department. Each department should have a name, a head of department, and a list of courses offered.

Additionally, create a **static nested class** called UniversityStats that provides utility methods like counting the total number of departments and listing all department names.

**Requirements:**

**Outer Class: University**

* Fields: universityName, location
* Method: addDepartment(String name, String head, List<String> courses)
* Method: displayDepartments()

**Inner Class: Department (non-static)**

* Fields: name, head, courses
* Method: displayInfo()

**Static Nested Class: UniversityStats**

* Method: getDepartmentCount(List<Department>)
* Method: listDepartmentNames(List<Department>)

**Expected Output Example:**

University Name: TechVille University

Location: Pune

Departments:

- Computer Science (Head: Dr. Mehta)

Courses: Data Structures, Algorithms, AI

- Mechanical Engineering (Head: Dr. Rao)

Courses: Thermodynamics, Fluid Mechanics

**3. Employee Management System**

Create a class Company that contains a non-static nested class Employee. Each Employee should have fields like name, id, and designation. The Company class should maintain a list of employees and provide methods to:

* Add a new employee
* Display all employees
* Search for an employee by ID

Add a static nested class CompanyPolicy that contains static methods to:

* Validate employee designation
* Enforce minimum working hours

**4. Smart Home Simulation**

Design a class SmartHome with a non-static inner class Device. Each device should have a name, type (e.g., light, thermostat), and status (on/off). The outer class should:

* Add and remove devices
* Toggle device status
* Display all devices

Include a static nested class DeviceUtils to:

* Count active devices
* Group devices by type

**5. Online Course Platform**

Create a class CoursePlatform with a nested class Course. Each course should have a title, instructor, and list of enrolled students. Implement methods to:

* Add a course
* Enroll a student
* Display course details

Add a static nested class PlatformStats to:

* Count total courses
* List all instructors

**6. Banking System**

Build a class Bank with a non-static inner class Account. Each account should store account number, holder name, and balance. The Bank class should:

* Create new accounts
* Deposit and withdraw money
* Display account details

Add a static nested class AuditTrail to:

* Log transactions
* Generate a summary of all account activities

**7. Library Catalog**

Create a class Library with a nested class Book. Each book should have a title, author, and genre. The outer class should:

* Add books to the catalog
* Search books by genre
* Display all books

Include a static nested class LibraryUtils to:

* Sort books alphabetically
* Count books by genre

**8. Hospital Management System**

Design a class Hospital with a non-static inner class Patient. Each patient should have a name, age, and illness. The Hospital class should:

* Admit new patients
* Discharge patients
* Display patient records

Add a static nested class HealthStats to:

* Calculate average age
* List common illnesses