Assignment No. 2

Module 2: Inheritance & Exception Handling

1) Code:

assignment2a.java

```java
// Interface to define salary-related behavior
interface Payroll {
    double calculateSalary();
}

// User-defined exception
class InvalidSalaryException extends Exception {
    public InvalidSalaryException(String message) {
        super(message);
    }
}

// Abstract class for common employee attributes
abstract class Employee {
    protected String name;
    protected double salary;

    // Constructor
    public Employee(String name, double salary) throws InvalidSalaryException {
        if (name == null || name.trim().isEmpty()) {
            throw new IllegalArgumentException(s:"Employee name cannot be null or empty.");
        }
        if (salary < 0) {
            throw new InvalidSalaryException(message:"Salary cannot be negative.");
        }
        this.name = name;
        this.salary = salary;
    }

    // Abstract method to display employee info
    public abstract void displayInfo();

    // Final method that cannot be overridden
    public final void companyPolicy() {
        System.out.println(x:"All employees must adhere to company policies.");
    }
}

// Full-time employee implementation
class FullTimeEmployee extends Employee implements Payroll {

    private double bonus;

    public FullTimeEmployee(String name, double salary, double bonus) throws InvalidSalaryException {
        super(name, salary);
        if (bonus < 0) {
            throw new InvalidSalaryException(message:"Bonus cannot be negative.");
```

```java
        if (bonus < 0) {
            throw new InvalidSalaryException(message:"Bonus cannot be negative.");
        }
        this.bonus = bonus;
    }

    @Override
    public double calculateSalary() {
        return salary + bonus;
    }

    @Override
    public void displayInfo() {
        System.out.println("Full-Time Employee: " + name);
        System.out.println("Salary: " + salary);
        System.out.println("Bonus: " + bonus);
        System.out.println("Total Salary: " + calculateSalary());
    }
}

// Part-time employee implementation
class PartTimeEmployee extends Employee implements Payroll {

    private int hoursWorked;
    private double hourlyRate;

    public PartTimeEmployee(String name, double salary, int hoursWorked, double hourlyRate) throws InvalidSalaryExcep
        super(name, salary);
        if (hoursWorked < 0 || hourlyRate < 0) {
            throw new InvalidSalaryException(message:"Hours worked and hourly rate cannot be negative.");
        }
        this.hoursWorked = hoursWorked;
        this.hourlyRate = hourlyRate;
    }

    @Override
    public double calculateSalary() {
        return hoursWorked * hourlyRate;
    }

    @Override
    public void displayInfo() {
        System.out.println("Part-Time Employee: " + name);
        System.out.println("Hourly Rate: " + hourlyRate);
        System.out.println("Hours Worked: " + hoursWorked);
        System.out.println("Total Salary: " + calculateSalary());
    }
}

// Final class that cannot be extended
final class Company {
    public void showWelcomeMessage() {
        System.out.println(x:"Welcome to the Employee Management System");
    }
}

// Main class – must match the file name assignment2a.java
public class assignment2a {

    Run | Debug
    public static void main(String[] args) {
        Company company = new Company();
        company.showWelcomeMessage();

        try {
            Employee fullTime = new FullTimeEmployee(name:"Alice", salary:50000, bonus:5000);
            Employee partTime = new PartTimeEmployee(name:"Bob", salary:0, hoursWorked:120, hourlyRate:300);
```

```
            Employee partTime = new PartTimeEmployee(name:"Bob", salary:0, hoursWorked:120, hourlyRate:300);

            fullTime.displayInfo();
            fullTime.companyPolicy(); // Final method

            partTime.displayInfo();
            partTime.companyPolicy(); // Final method

        } catch (InvalidSalaryException e) {
            System.out.println("Invalid Salary Error: " + e.getMessage());
        } catch (IllegalArgumentException e) {
            System.out.println("Invalid Input: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("General Error: " + e.getMessage());
        } finally {
            System.out.println(x:"Execution completed. Cleaning up resources.");
        }
    }
}
```

Output:

```
PROBLEMS 13  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                Run: assignment2a

PS C:\Users\91788\Downloads\java>  & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\91788\AppData\Roaming\Code\User\workspaceStorage\f5fa97c5fc5359ad7a9244a856664d07\redhat.java\jdt_ws\java_deb43c15\bin' 'assignment2a'
Welcome to the Employee Management System
Full-Time Employee: Alice
Salary: 50000.0
Bonus: 5000.0
Total Salary: 55000.0
All employees must adhere to company policies.
Part-Time Employee: Bob
Hourly Rate: 300.0
Hours Worked: 120
Total Salary: 36000.0
All employees must adhere to company policies.
Execution completed. Cleaning up resources.
PS C:\Users\91788\Downloads\java>
```

2) Code:

assignment2b.java

```java
// Print task using Runnable interface
class PrintTask implements Runnable {
    @Override
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println("Printing document " + i);
            try {
                Thread.sleep(millis:1000); // simulate time taken to print
            } catch (InterruptedException e) {
                System.out.println(x:"Printing interrupted.");
            }
        }
    }
}

// Scan task using Thread class
class ScanTask extends Thread {
    @Override
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println("Scanning document " + i);
            try {
                Thread.sleep(millis:1200); // simulate time taken to scan
            } catch (InterruptedException e) {
                System.out.println(x:"Scanning interrupted.");
```

```java
                System.out.println(x:"Scanning interrupted.");
            }
        }
    }
}

// Main class — must match the file name assignment2b.java
public class assignment2b {
    Run | Debug
    public static void main(String[] args) {
        // Create print task thread
        Thread printThread = new Thread(new PrintTask());

        // Create scan task thread
        ScanTask scanThread = new ScanTask();

        // Start both threads
        printThread.start();
        scanThread.start();

        System.out.println(x:"Printing and scanning tasks started...");
    }
}
```

Output:

```
PROBLEMS 13  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS        Run: assignment2b

PS C:\Users\91788\Downloads\java>  & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetail
' '-cp' 'C:\Users\91788\AppData\Roaming\Code\User\workspaceStorage\f5fa97c5fc5359ad7a9244a856664d07\redhat.java\jdt_ws\java_deb43c15
ssignment2b'
Printing and scanning tasks started...
Printing document 1
Scanning document 1
Printing document 2
Scanning document 2
Printing document 3
Scanning document 3
Printing document 4
Scanning document 4
Printing document 5
Scanning document 5
PS C:\Users\91788\Downloads\java>
```