Experiment - 2

Student Name: Aryan Srivastav   **UID:** 23BCS10135
Branch: BE-CSE   **Section/Group:** KRG-2B
Semester: 5ᵗʰ   **Date of Performance:**13/8/25
Subject Name: Project Based Learning in Java
Subject Code: 23CSH-304

Aim: To develop Java programs to manage product details, library systems, and student information using classes, inheritance, and abstraction.

Easy-level Problem-

Aim: To write a Java program to create a Product class with attributes id,name and price.The program should:
   - Demonstrate the use of constructors and methods to display product details.

Objective: To understand use of classes, constructors and methods in Java using concepts like Java class definition, constructor and method usage.

Procedure:
      1. Define a class named 'Product' with attributes 'id', 'name' and 'price'.
      2. Use a parameterized constructor to initialize these attributes.
      3. Define a method 'displayDetails()' to print product information.
      4. In the main method, create an object and display its details.

Sample Input -
Product ID: 101
Name: Laptop
Price: 75000

Sample Output -
Product Details:
ID: 101
Name: Laptop
Price: 75000

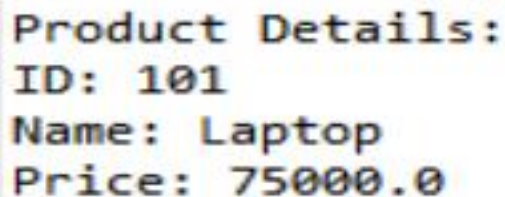## Code -

```java
package exp2;

class Product {
int id;
String name;
double price;

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

// Method to display details
void displayDetails() {
System.out.println("Product Details:");
System.out.println("ID: " + id);
System.out.println("Name: " + name);
System.out.println("Price: " + price);
}
}

public class easy {
public static void main(String[] args) {
// Creating product object
Product p1 = new Product(101, "Laptop", 75000);
p1.displayDetails();
}
}
```

Output -

```
Product Details:
ID: 101
Name: Laptop
Price: 75000.0
```

Medium- Level Problem -

Aim : To write a Java program to implement a library management system. The program should :
  - Use a base class Book and derived classse Fiction and NonFiction.

Objective: Understand inheritance and dynamic method invocation in Java using concepts of Java inheritance using base and derived classes.

Procedure:
      1. Define a base class 'Book' with common attributes like title,author and price.
      2. Create two derived classes: 'Fiction' and 'NonFiction' extending the 'Book' class.
      3. Override method in each subclass to display respective book details.
      4. Instantiate objects of each subclass and invoke their display methods.

Sample Input :
Book 1:
Type: Fiction
Title: Harry Potter
Author: J.K. Rowling
Price: 500

Book 2:
Type: Non-Fiction
Title: Sapiens
Author: Yuval Noah Harari
Price: 700

Sample Output:
Fiction Book Details:
Title: Harry Potter
Author: J.K. Rowling
Price: 500

Non-Fiction Book Details:
Title: Sapiens

Author: Yuval Noah Harari

Price: 700

Code :

```java
// Base class
package exp2;

class Book {
String title, author;
double price;

Book(String title, String author, double price) {
this.title = title;
this.author = author;
this.price = price;
}

void displayDetails() {
System.out.println("Book Details:");
System.out.println("Title: " + title);
System.out.println("Author: " + author);
System.out.println("Price: " + price);
}
}

// Derived class Fiction
class Fiction extends Book {
Fiction(String title, String author, double price) {
super(title, author, price);
}

@Override
void displayDetails() {
System.out.println("Fiction Book Details:");
System.out.println("Title: " + title);
System.out.println("Author: " + author);
System.out.println("Price: " + price);
}
}

// Derived class NonFiction
class NonFiction extends Book {
NonFiction(String title, String author, double price) {
super(title, author, price);
}

@Override
void displayDetails() {
```

```java
System.out.println("Non-Fiction Book Details:");
System.out.println("Title: " + title);
System.out.println("Author: " + author);
System.out.println("Price: " + price);
}
}

public class medium {
public static void main(String[] args) {
Fiction f1 = new Fiction("Harry Potter", "J.K. Rowling", 500);
NonFiction nf1 = new NonFiction("Sapiens", "Yuval Noah Harari", 700);

f1.displayDetails();
System.out.println();
nf1.displayDetails();
}
}
```

Output:

```
Fiction Book Details:
Title: Harry Potter
Author: J.K. Rowling
Price: 500.0

Non-Fiction Book Details:
Title: Sapiens
Author: Yuval Noah Harari
Price: 700.0
```

Hard -level Problem-

Aim : To design a student information system using Java with following features:
 - Use an abstract class Person with attributes name,age and methods like displayDetails().
 - Create derived classes Student and Teacher to override displayDetails() and add unique attributes like rollNumber for students and subject for teachers.

Objective: Demonstrate abstraction and polymorphism using abstract classes and derived classes using Java concepts of abstract classes, inheritance and overriding .

Procedure:

1. Define an abstract class 'Person' with attributes 'name' and 'age', and an abstract method 'displayDetails()'.
2. Create a 'Student' class extending 'Person' , with an additional attribute 'rollNumber', and implement 'displayDetails()'.
3. Create a 'Teacher' class extending 'Person' with an additional attribute 'subject' and implement 'displayDetails()'.
4. In the main method, create objects of 'Student' and 'Teacher' and invoke 'displayDetails()' on each.

Sample Input:
Add Student:
Name: Alice
Age: 20
Roll Number: 101

Add Teacher:
Name: Mr. Smith
Age: 40
Subject: Mathematics

Sample Output:
Student Details:
Name: Alice
Age: 20
Roll Number: 101

Teacher Details:
Name: Mr. Smith
Age: 40
Subject: Mathematics

Code :

```java
// Abstract class
package exp2;

abstract class Person {
String name;
int age;

Person(String name, int age) {
this.name = name;
this.age = age;
}

// Abstract method
abstract void displayDetails();
}

// Student class
class Student extends Person {
int rollNumber;

Student(String name, int age, int rollNumber) {
super(name, age);
this.rollNumber = rollNumber;
}

@Override
void displayDetails() {
System.out.println("Student Details:");
System.out.println("Name: " + name);
System.out.println("Age: " + age);
System.out.println("Roll Number: " + rollNumber);
}
}

// Teacher class
class Teacher extends Person {
String subject;

Teacher(String name, int age, String subject) {
super(name, age);
this.subject = subject;
}
```

```java
@Override
void displayDetails() {
System.out.println("Teacher Details:");
System.out.println("Name: " + name);
System.out.println("Age: " + age);
System.out.println("Subject: " + subject);
}
}

public class hard {
public static void main(String[] args) {
Student s1 = new Student("Alice", 20, 101);
Teacher t1 = new Teacher("Mr. Smith", 40, "Mathematics");

s1.displayDetails();
System.out.println();
t1.displayDetails();
}
}
```

Output:

```
<terminated> hard (1) Java Applica
Student Details:
Name: Alice
Age: 20
Roll Number: 101

Teacher Details:
Name: Mr. Smith
Age: 40
Subject: Mathematics
```