Southeast University

Department of Computer Science and Engineering (CSE)

School of Sciences and Engineering

Semester: (Summer, Year: 2025)

LAB REPORT NO: **01**

Course Title: Introduction to Programming Language II (Java) Lab

Course Code: CSE282.2

Batch: 65

Lab Experiment Name: Installation of IDE and write the basic program in JAVA.

Student Details

	Name	ID
1.	Md. Mahmud Hossain	2023200000799

Submission Date : **23-07-25**

Course Teacher's Name : **Dr. Mohammed Ashikur Rahman**

Lab Report Status

Marks: …… **Signature:.....**

Comments:..... **Date:.....**

Lab Task 1: Installation of IDE and write the basic program in JAVA.

OBJECTIVES:

- Getting acquainted with JDK and Netbeans IDE
- Writing the first Java program using Netbeans IDE
- To be familiar with the basic concept of class and object

PROBLEM:

1. Create a class called BankAccount with instance variables accountNumber and balance. Add methods to deposit and withdraw money from the account. Create objects of BankAccount and perform deposit and withdrawal operations.

2. Create a class rectangle with properties such as length and width. Add methods to calculate the perimeter and area of the rectangle. Create objects and display their corresponding perimeter and area.

3. Create a class called movie which as properties such as title, genre, leadactor, director, release year, rating and review. Create two movie objects and display their properties. If the rating is <5, the review should be “Not Good”. Otherwise, the review would be “Good”.

Solution:

1.

Problem Analysis:

We need to create a BankAccount class with properties such as accountNumber & balance.Then, we need to create two objects of the BankAccount class and perform deposit and withdraw using methods.

Background Theory:

- Object variables (accountNumber, balance) are declared without static, so each object has its own copy.
- Object methods (display(), deposit(), withdraw()) are called using object references and perform actions specific to each object.
- The main method is a class method and serves as the entry point of the program.
- Class methods can be called without creating an object, whereas object methods require an instance.
- The class does not contain any class variables .
- Each object (acc1, acc2) stores and processes its own data independently.
- The method logic currently prints new balances but does not update the actual balance variable.

Algorithm Design:

1. Create a BankAccount class.
2. Declare object variables, such as accountNumber and balance, to store account information.
3. Define object methods, such as display(), to show account details.
4. Define object methods, such as deposit(double) and withdraw(double), to perform transactions.
5. In the main method, create bank account objects (acc1, acc2) using the default constructor.
6. Assign values to object variables like accountNumber and balance manually.
7. Use object methods to display account information and perform deposit or withdrawal.

Code:

```
public class BankAccount {
    int accountNumber;
    double balance;

    void display()
    {
        System.out.println("Account Number: "+accountNumber);
        System.out.println("Previous Balance: "+balance);
    }
    void deposit(double add)
    {
        add = balance + add;
        System.out.println("New balance: "+add);
    }
    void withdraw(double substract)
    {
        substract = balance - substract;
```

```
        System.out.println("New balance: "+substract);
    }

    public static void main(String[] args) {

        BankAccount acc1 = new BankAccount();
        acc1.accountNumber = 101;
        acc1.balance = 1000;
        acc1.display();
        acc1.deposit(500);

        System.out.println("");

        BankAccount acc2 = new BankAccount();
        acc2.accountNumber = 201;
        acc2.balance = 3000;
        acc2.display();
        acc2.withdraw(300);
```

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

Output:

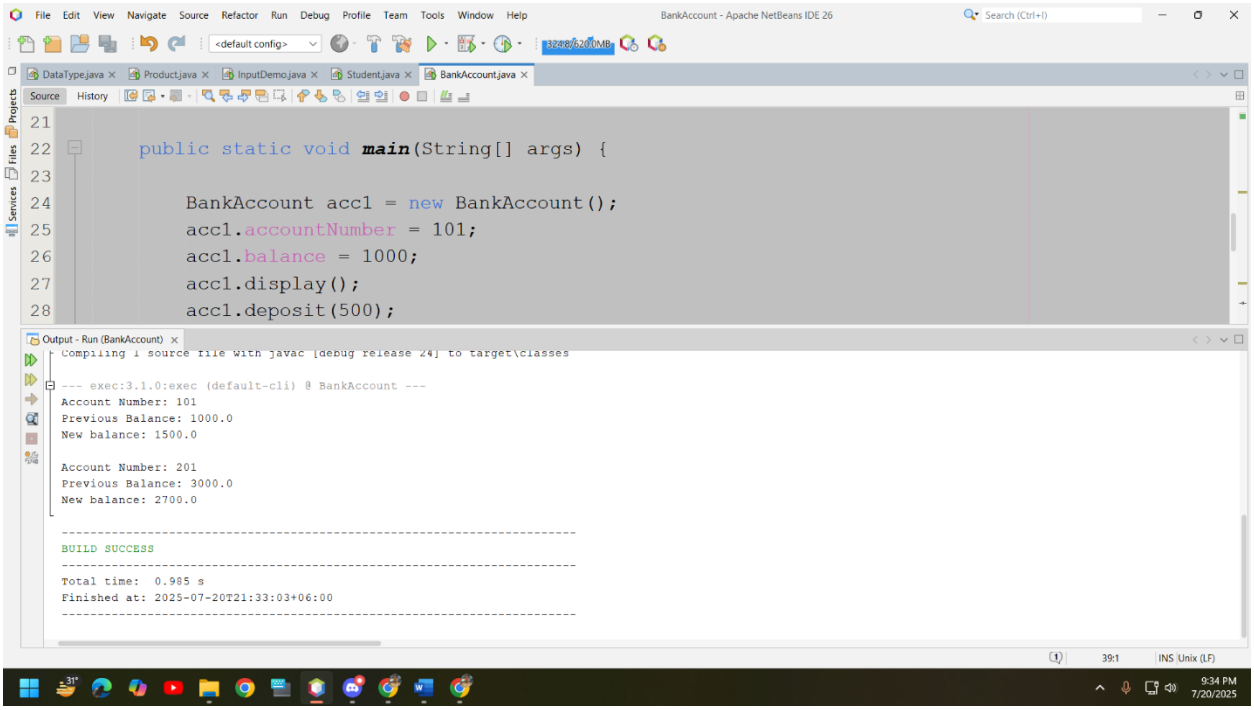

Figure 1: Output

2.

Problem Analysis:

We need to create a rectangle class with properties such as length & width. .Then, we need to create two objects of the rectangle class and calculate perimeter and area using methods. Lastly display their corresponding perimeter and area.

Background Theory:

- Object variables length and width are used to store individual rectangle dimensions.
- Object methods like perimeter(), area(), and display() are called using object references and perform actions for each specific rectangle.
- The main method is a class method (static) and serves as the program's entry point.
- The display() method combines object data and method outputs to show full rectangle info.
- Each rectangle object (r1, r2) holds its own unique values for length and width.

Algorithm Design:

1. Create a Rectangle class.
2. Declare object variables: length and width to store rectangle properties.
3. Define object methods:
4. perimeter() to calculate and return the perimeter.
5. area() to calculate and return the area.
6. display() to show length, width, perimeter, and area.
7. In the main method, create rectangle objects (r1, r2) using the default constructor.
8. Assign values to each rectangle's length and width.
9. Use the display() method to output the details of each rectangle.

Code:

```
public class Rectangle {
    double length;
    double width;

    double perimeter()
    {
        return 2*(length+width);
    }
    double area()
    {
        return length*width;
    }
```

```
    void display(){
        System.out.println("Length: "+length);
        System.out.println("Width: "+width);
        System.out.println("Perimeter: "+perimeter());
        System.out.println("Area: "+area());
    }

    public static void main(String[] args) {

        Rectangle r1 = new Rectangle();
        r1.length=25;
        r1.width=45;
        System.out.println("Rectangle no.1:");
        r1.display();

        System.out.println("");

        Rectangle r2 = new Rectangle();
        r2.length=15;
        r2.width=64;
        System.out.println("Rectangle no.2:");
        r2.display();
    }
}
```

Output:

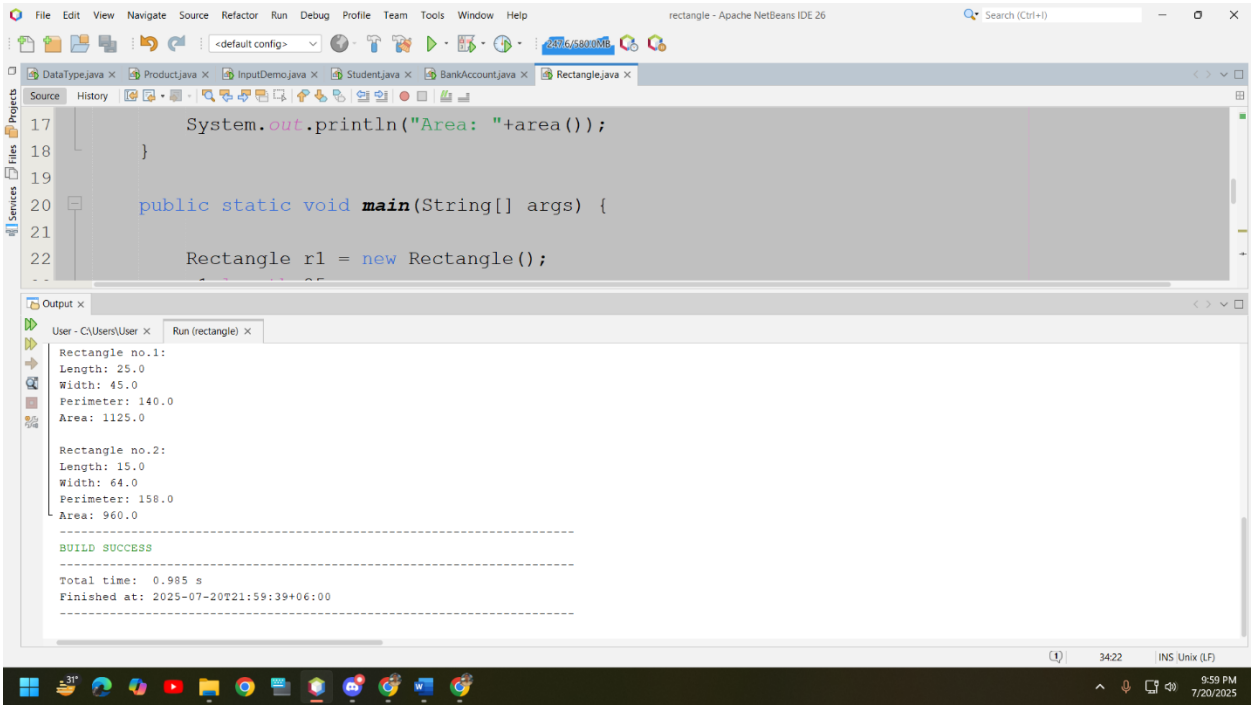

Figure 2: Output

3.

Problem Analysis:

We need to create a movie class with properties such as title, genre, leadActor, director, releaseYear, rating & review. Then, we need to create two objects of the movie class and give review using methods. Lastly display their properties.

Background Theory:

- Object variables (title, genre, leadActor, director, releaseYear, rating, review) are declared without static, so each Movie object stores its own values.
- Object methods (display(), review()) are used to display movie information and evaluate the movie based on its rating.
- The method review() uses a conditional (ternary) operator to assign a review comment based on the rating value.
- The main method is a class method (static) and serves as the entry point of the program.

- Objects (m1, m2) are created manually and their properties are set directly through object references.
- Each object functions independently, storing and processing its own movie details and rating.

Algorithm Design:

1. Create a Movie class.
2. Declare object variables such as title, genre, leadActor, director, releaseYear, rating, and review to store movie details.
3. Define an object method display() to print the movie's properties.
4. Define an object method review() that evaluates the rating and sets the review as "Good!" or "Not Good!" based on the value.
5. In the main method, create Movie objects using the default constructor.
6. Set values of movie properties directly using the object (e.g., m1.title = "Cats").
7. Use the display() method to show each movie's details.
8. Use the review() method to display a comment based on the movie's rating.

Code:

```
public class Movie {
    String title;
    String genre;
    String leadActor;
    String director;
    int releaseYear;
    int rating;
    String review;
```

```
void display(){
    System.out.println("");
    System.out.println("Title: "+title);
    System.out.println("Genre: "+genre);
    System.out.println("Lead-Actor: "+leadActor);
    System.out.println("Director: "+director);
    System.out.println("Release Year: "+releaseYear);
    System.out.println("Rating: "+rating);
}
void review(){
    review = (rating<5) ? "Not Good!" : "Good!";
    System.out.println("Review: "+review);
}

public static void main(String[] args) {
    Movie m1 = new Movie();
    m1.title = "Cats";
    m1.genre = "Fantasy";
```

```
m1.leadActor = "Francesca Hayward";
m1.director = "Tom Hooper";
m1.releaseYear = 2019;
m1.rating = 2;

Movie m2 = new Movie();
m2.title = "Cats";
m2.genre = "Fantasy";
m2.leadActor = "Francesca Hayward";
m2.director = "Tom Hooper";
m2.releaseYear = 2019;
m2.rating = 6;

System.out.println("Movie no.1:");
m1.display();
m1.review();
System.out.println("");
System.out.println("Movie no.2:");
```

```
        m2.display();

        m2.review();

    }

}
```

Output:

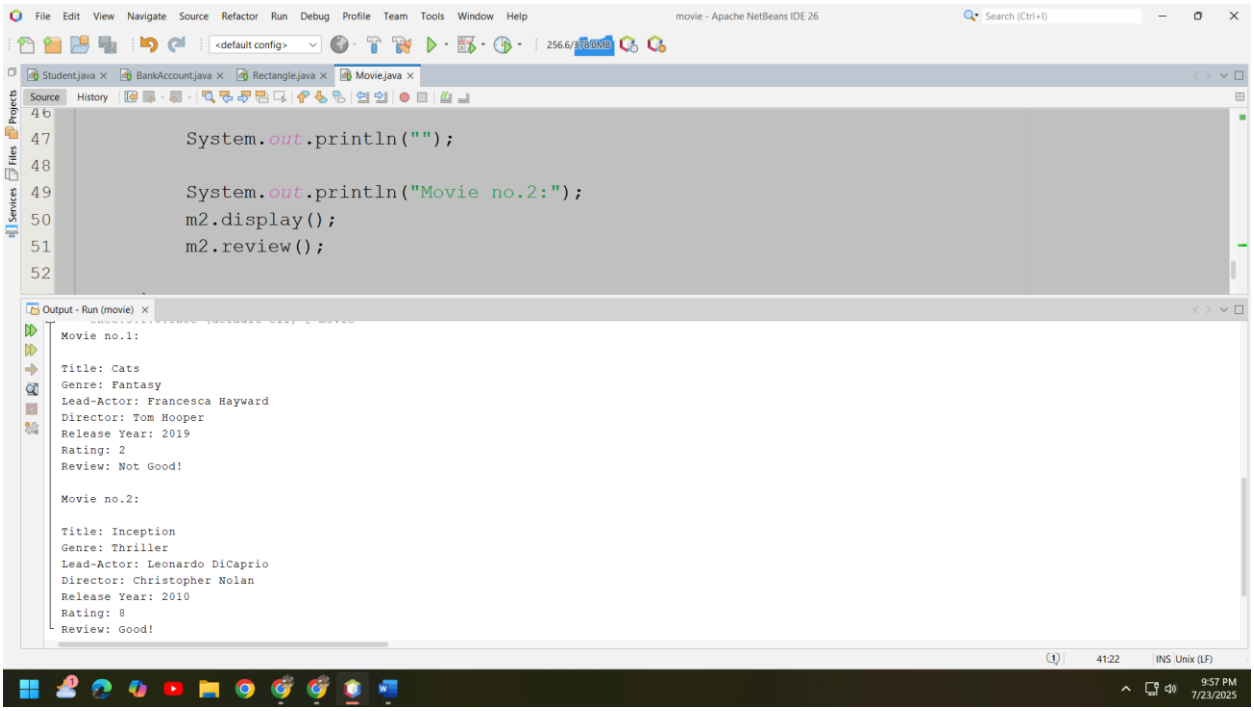

Figure 3: Output