**Exercise 7: Financial Forecasting**

**Financial Forecasting.java**

**package** mypackage;

**import** java.util.Scanner;

**public** **class** FinancialForecasting {

**public** **static** **double** predictFutureValue(**double** currentValue, **double** growthRate, **int** years) {

// Base case

**if** (years == 0) {

**return** currentValue;

}

// Recursive call

**return** *predictFutureValue*(currentValue \* (1 + growthRate / 100), growthRate, years - 1);

}

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

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

System.***out***.println("Financial Forecasting Tool");

System.***out***.print("Enter current value (in Indian Rupees): ");

**double** currentValue = sc.nextDouble();

System.***out***.print("Enter annual growth rate (percentage): ");

**double** growthRate = sc.nextDouble();

System.***out***.print("Enter number of years: ");

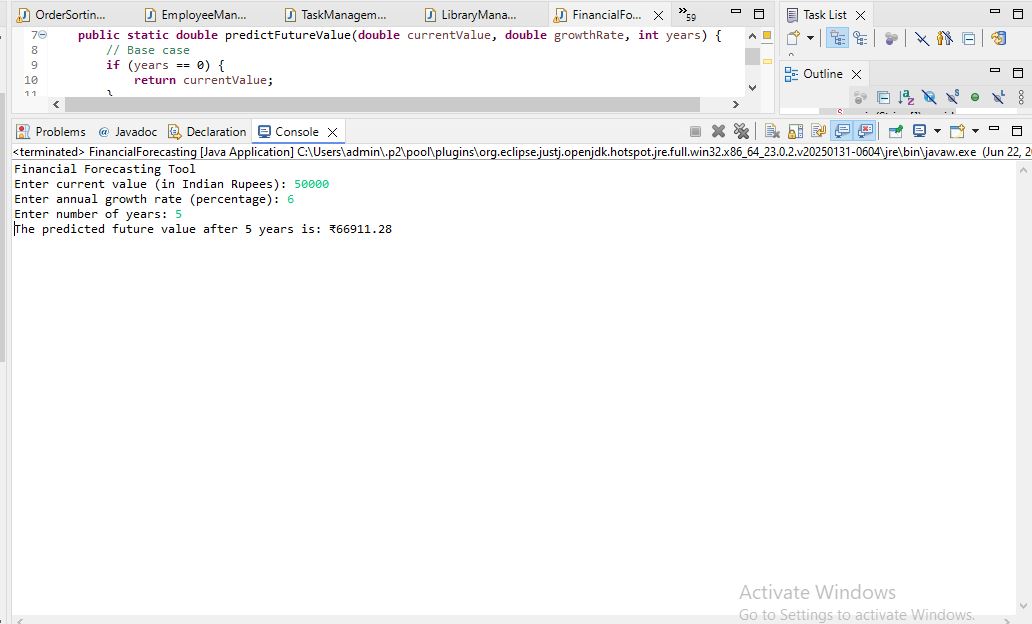
**int** years = sc.nextInt();

**double** futureValue = *predictFutureValue*(currentValue, growthRate, years);

System.***out***.printf("The predicted future value after %d years is: ₹%.2f\n", years, futureValue);

}

}



**Financial Forecasting using Recursion**

We are developing a financial forecasting tool that predicts future financial values based on past data and growth rates. The problem will be solved using recursion.

**Understand Recursive Algorithms:**

Recursion is a programming technique where a method calls itself to solve smaller instances of the same problem. Each recursive call works with a smaller input until it reaches a base case that stops the recursion.

Recursion simplifies problems that have repetitive subproblems, such as financial forecasting, where the same calculation is applied year after year.

In financial forecasting, each year’s value depends on the previous year’s value. This makes recursion a natural approach for calculating future values.

**Setup:**

We create a method called predictFutureValue that accepts:

* currentValue: The present financial amount.
* growthRate: The annual growth rate in percentage.
* years: The number of years to predict into the future.

1. Implementation:

**The recursive method works as follows:**

* If the number of years is zero, return the current value (base case).
* Otherwise, calculate the value for the next year by multiplying the current value by (1 + growthRate / 100), and then call the function recursively for (years - 1).

This continues until all years are processed.

1. Analysis:

**Time Complexity:**

* Each recursive call reduces the problem size by one year.
* The total number of calls equals the number of years.
* Therefore, time complexity is O(n), where n is the number of years.

**Optimization:**

* For very large values of years, recursion may cause stack overflow due to deep recursion.
* This can be optimized using iterative solutions or dynamic programming.
* Alternatively, the compound interest formula can directly compute the result without recursion:

Future Value **= Current Value × (1 + growthRate / 100) ^ years**

* This direct formula has constant time complexity O(1), which is highly efficient.

**Example:**

Financial Forecasting Tool

Enter current value (in Indian Rupees): 100000

Enter annual growth rate (percentage): 8

Enter number of years: 5

**The predicted future value after 5 years is: ₹146933.28**

Year 1: 100000 \* (1 + 8/100) = 108000.0

Year 2: 108000.0 \* (1 + 8/100) = 116640.0

Year 3: 116640.0 \* (1 + 8/100) = 125971.20

Year 4: 125971.20 \* (1 + 8/100) = 136048.90

Year 5: 136048.90 \* (1 + 8/100) = 146933.28