**Name: Amogh Girish Nagarkar**

**Superset ID: 6403503**

**DN-4.0 - Java FSE - Deep Skilling**

**Exercise 7: Financial Forecasting**

**Scenario:**

**You are developing a financial forecasting tool that predicts future values based on past data.**

1. **Understanding Recursive Algorithms**

Recursion is a programming concept where a method calls itself to solve smaller parts of a bigger problem. It helps break complex problems into simpler steps.

For example, instead of multiplying a number several times with a loop, we can let a function call itself with a smaller number each time. Recursion is especially helpful when the pattern of solving a problem repeats itself in smaller chunks.

1. **Setup**

We want to calculate the future value of an investment using this formula:

FutureValue = PresentValue × (1 + growthRate) ^ years

1. **Java Implementation**

public class Forecast { public static double power(double base, int years) {

if (years == 0) { return 1;

} else {

return base \* power(base, years - 1);

}

}

public static double futureValue(double presentValue, double growthRate, int years) { double base = 1 + growthRate; return presentValue \* power(base, years);

}

public static void main(String[] args) { double presentValue = 10000; double growthRate = 0.08;

int years = 5;

double result = futureValue(presentValue, growthRate, years);

System.out.println("Predicted Future Value: ₹" + result);

}

}

**Sample Output**

Predicted Future Value: ₹14693.28

**4. Analysis**

**Time Complexity:**

The power() function runs once for each year, so its time complexity is O(n), where n is the number of years.

The futureValue() method depends on it, so overall the solution is O(n).

**Optimization:**

Recursive functions can be slow and use more memory when the number of recursive calls is large. To optimize:

• We can use an iterative approach (using a loop) instead of recursion.