**Financial Forecasting**

**Step 1: Understanding Recursive Algorithms**

**Recursion:**

* When a method calls itself to solve smaller components of the same problem, this is known as recursion.
* Before arriving at a straightforward base case, it divides complex issues into smaller ones.

It simplifies code for problems like:

* Factorial
* Fibonacci series
* Tree traversals
* Calculating future values

**Step 2: Setup**

**Formula:**

Future Value = Present Value × (1 + rate)^years

* amount = Starting amount
* rate = Growth rate
* years = Number of years to grow

**Step 3: Implementation**

import java.util.Scanner;

public class FutureValuePredictor {

    public static double PredictFutureValue(double amount, double rate, int years) {

        double result = amount \* Math.pow(1 + rate, years);

        return result;

    }

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        double Amount = sc.nextDouble();

        double Rate = sc.nextDouble();

        int Time = sc.nextInt();

        double FutureValue = PredictFutureValue(Amount, Rate, Time);

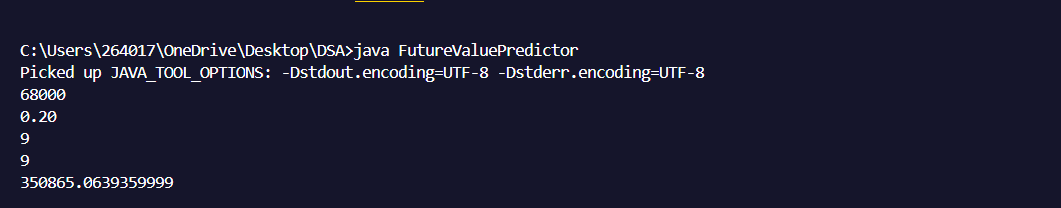
        System.out.println(Time);

        System.out.println(FutureValue);

    }

}

**Output:**

****

**Step 4: Analysis**

**Time Complexity:**

* Once a year, the method makes a call to itself.
* Thus, it runs n times for n years.
* Time Complexity: O(n) (linear)

**Optimization:**

Using Iteration:

Rewrite recursion into a loop to avoid excessive consumption.   
public static double PredictFutureValueIterative(double amount, double rate, int years) {  
for (int i = 0; i < years; i++) {  
amount \*= (1 + rate);  
}  
return amount;  
}