**Exercise 7: Financial Forecasting**

* **Step-1: Recursive Algorithms**
* Recursion is a programming technique in which a function calls itself to solve a problem. This can be a very effective strategy since it enables us to break down big problems into smaller sub-problems, making solution finding easier.
* It is best suitable for problems like Fibonacci Sequence, Permutations, Forecasting Etc..
* **Step-2: Setup**

Let us consider any Formula which is used to find future Financial Forecasting.

### **Compound Interest Formula:**

C.I = P \*(1+r)^n

Where,

C.I= Compound Interest (or future Value in our case)

P=Principal amount

R=rate of interest

N=Number of years

* **Step-3:Implementation**

**FinancialForecast.java**

public class FinancialForecast {

    public static double forecast(double principal, double rate, int years) {

        if (years == 0) {

            return principal;

        }

        return forecast(principal, rate, years - 1) \* (1 + rate);

    }

    public static void main(String[] args) {

        double p = 10000;

        double r = 0.05;

        int y = 3;

        double fv = forecast(p, r, y);

        System.out.printf("Future Value after %d years = %f\n", y, fv);

    }

}

**Step-4:analysis**

* The time Complexity of the Recursive algo is O(n) as we are calling recursively (n-1) years.
* To avoid the excessive computations we can use Dynamic Programming ,i.e Memorization using DP array for recursive algorithms but in our case there is no use of DP its already efficient O(n).

**Output:**

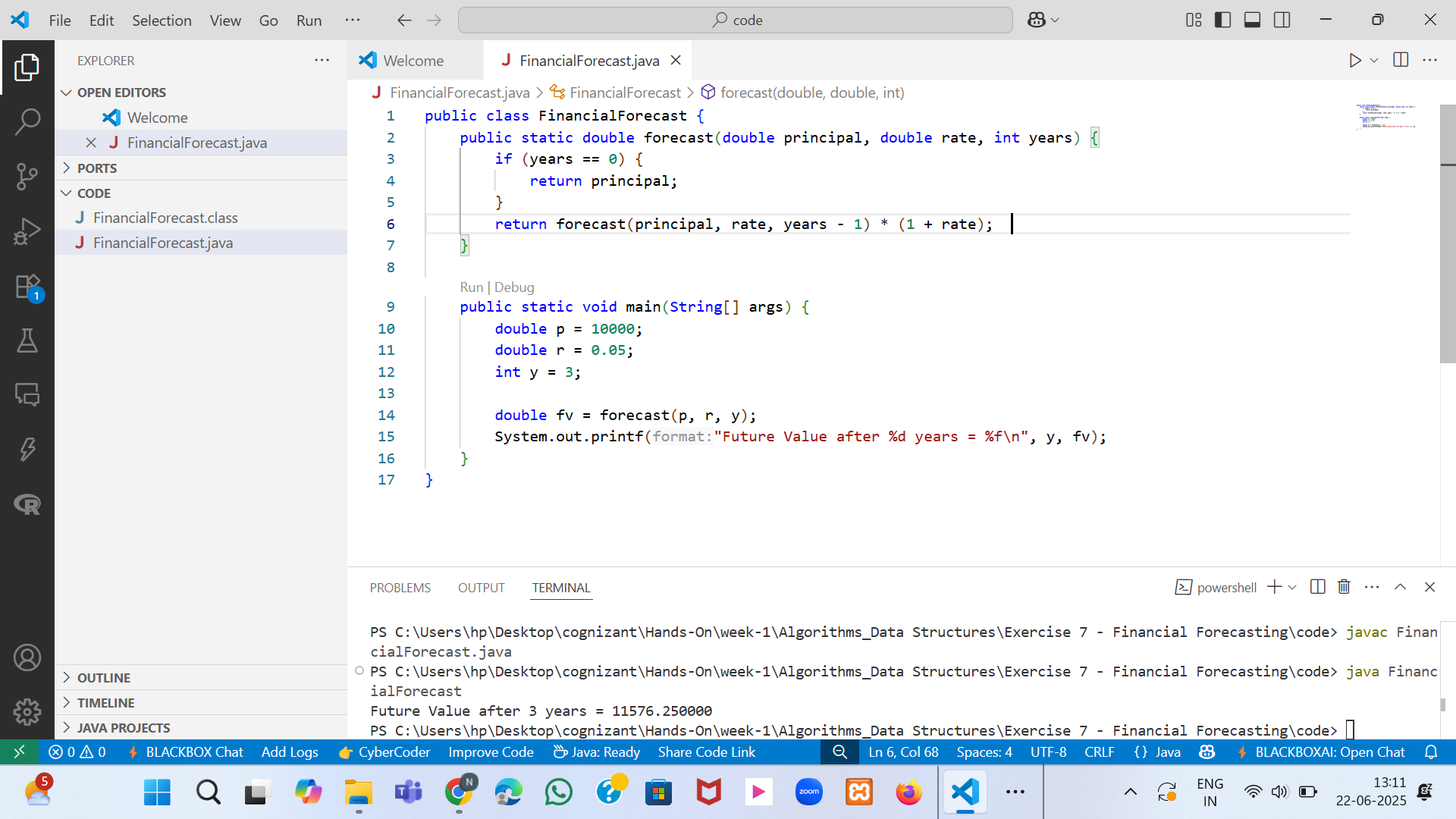


Fig-1:FinancialForecast class with output