

# Data Structures and Algorithms

## Exercise 7:

### Financial Forecasting

This project implements a financial forecasting tool using Java. It uses a recursive algorithm to predict future values based on historical growth and includes an optimized iterative version for efficiency. Here's a detailed explanation:

#### Step 1: Understand Recursive Algorithms

- Recursion is a method where a function calls itself to solve smaller instances of the same problem.
  - It simplifies problems like computing compounded growth.
- 

#### Step 2: Setup

- Define a class *FinancialForecasting*.
  - Inside it, create methods to calculate the future value recursively and iteratively.
  - Used input values like:
    - *initialValue* - base amount.
    - *growthRate* - annual percentage increase.
    - *years* - time period for forecasting.
- 

#### Step 3: Implementation

- *forecastRecursive()* - recursively computes future value.
  - *forecastIterative()* - optimized iterative approach to avoid recursion stack overhead.
  - *main()* - sets up initial values and runs both methods.
- 

#### Step 4: Time Complexity Analysis

Method	Time Complexity	Space Complexity
Recursive	O(n)	O(n)

Method	Time Complexity	Space Complexity
Iterative (Optimized)	$O(n)$	$O(1)$

- Use **Recursive** approach for simplicity and learning.
- Use **Iterative** approach for performance and large datasets.

## Output

The screenshot shows a Visual Studio Code editor with a Java file named `FinancialForecasting.java`. The code implements a financial forecasting algorithm using recursion. The terminal window shows the output of the program, which is a future value of 12762.82 for both the recursive and iterative methods.

```

1 // Exercise 7: Financial Forecasting using Recursion
2
3 // Step 1: Understanding Recursion
4 // Recursion is a method where the solution to a problem depends on solutions to smaller instances of the
5 // It is useful for problems that have a natural recursive structure (like financial projections, Fibonacci
6
7 public class FinancialForecasting {
8
9     // Step 2 & 3: Recursive method to calculate future value
10    // FV = PV * (1 + rate)^years
11    public static double forecastRecursive(double currentValue, double growthRate, int years) {
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

PS C:\Users\User> & 'C:\Program Files\Eclipse Adoptium\jdk-21.0.7-hotspot\bin\java.exe' '-agentlib:jdwp=trans
port=dt_socket,server=n,suspend=y,address=localhost:50756' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\U
sers\User\AppData\Local\Temp\vscodesws_b7c85\jdt_ws\jdt.ls-java-project\bin' 'FinancialForecasting'
--- Recursive Forecast ---
Future Value (Recursive): 12762.82

--- Iterative Forecast (Optimized) ---
Future Value (Iterative): 12762.82
PS C:\Users\User>

```