Exercise 7: Financial Forecasting

# Scenario

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

# 1. Understand Recursive Algorithms

Recursion is a programming technique where a function calls itself to solve a problem. This approach breaks down a complex problem into simpler sub-problems. Recursion is particularly useful in problems that have a natural hierarchical structure or when the same operation is applied repeatedly over a changing input.  
  
For example, calculating compound interest over multiple years can be viewed recursively — the value for year n depends on the value from year n-1.

# 2. Setup

We want to calculate the future value (FV) of an investment using a recursive approach.  
Assume we are given:  
- P: Initial principal amount  
- r: Annual growth rate (in decimal, e.g., 5% = 0.05)  
- n: Number of years  
  
The recursive formula for future value:  
FV(n) = FV(n - 1) \* (1 + r)  
Base case: FV(0) = P

# 3. Implementation

Here is a sample implementation in Python using recursion:

def future\_value\_recursive(P, r, n):  
 if n == 0:  
 return P  
 else:  
 return future\_value\_recursive(P, r, n - 1) \* (1 + r)  
  
# Example usage:  
P = 1000 # Initial amount  
r = 0.05 # 5% annual growth  
n = 5 # Forecast for 5 years  
  
print("Future Value after", n, "years:", future\_value\_recursive(P, r, n))

# 4. Analysis

## Time Complexity

The time complexity of the recursive solution is O(n) because for each year n, the function performs a single recursive call until the base case is reached.

## Optimization

Although the recursive approach is elegant, it can lead to excessive function calls and stack overflow for large n.  
  
To optimize:  
- Memoization can store previously computed values and avoid redundant computations.  
- Alternatively, use an iterative approach, which is more efficient for this type of problem.

Optimized version using iteration:

def future\_value\_iterative(P, r, n):  
 FV = P  
 for \_ in range(n):  
 FV \*= (1 + r)  
 return FV

# Conclusion

Recursive algorithms offer an intuitive way to solve repetitive problems like financial forecasting. However, it's important to be aware of their limitations in terms of performance and memory usage. Understanding when and how to optimize recursive code is key to building efficient forecasting tools.