**WEEK-1 (6397712 BANDARU SOWMYA)**

**Exercise 7: Financial Forecasting**

**CODE:**

**FINANCIALFORECAST.JAVA:**

import java.util.HashMap;

import java.util.Map;

public class FinancialForecast {

// Memoization cache to store computed future values

private Map<Integer, Double> memo;

public FinancialForecast() {

this.memo = new HashMap<>();

}

/\*\*

\* Recursively calculates the future value of an investment.

\* @param initialValue The initial investment amount

\* @param growthRates Array of growth rates for each period

\* @param period The number of periods to forecast

\* @return The future value after the specified periods

\*/

public double calculateFutureValue(double initialValue, double[] growthRates, int period) {

// Base case: if period is 0, return the initial value

if (period == 0) {

return initialValue;

}

// Check memoization cache

int key = period;

if (memo.containsKey(key)) {

return memo.get(key);

}

// Recursive case: FV = PV \* (1 + r)

double previousValue = calculateFutureValue(initialValue, growthRates, period - 1);

double growthRate = growthRates[(period - 1) % growthRates.length]; // Cycle through growth rates

double futureValue = previousValue \* (1 + growthRate);

// Store result in memoization cache

memo.put(key, futureValue);

return futureValue;

}

/\*\*

\* Clears the memoization cache for reuse.

\*/

public void clearCache() {

memo.clear();

}

public static void main(String[] args) {

FinancialForecast forecast = new FinancialForecast();

// Sample data

double initialInvestment = 1000.0; // $1000 initial investment

double[] growthRates = {0.05, 0.03, 0.04}; // Growth rates: 5%, 3%, 4%

int periods = 5; // Forecast for 5 periods

// Calculate future value

double futureValue = forecast.calculateFutureValue(initialInvestment, growthRates, periods);

// Output result

System.out.printf("Initial Investment: $%.2f%n", initialInvestment);

System.out.println("Growth Rates: " + java.util.Arrays.toString(growthRates));

System.out.printf("Future Value after %d periods: $%.2f%n", periods, futureValue);

// Clear cache for next calculation

forecast.clearCache();

}

}

**OUTPUT:**





