# Exercise 2 and 7 - Java Code and Analysis

## Exercise 2: E-commerce Platform Search Function

Scenario: Optimize search functionality using linear and binary search.

### Asymptotic Notation

Big O Notation is used to analyze algorithm efficiency as data size increases.  
  
| Algorithm | Best Case | Average Case | Worst Case |  
|-----------------|-----------|--------------|-------------|  
| Linear Search | O(1) | O(n) | O(n) |  
| Binary Search | O(1) | O(log n) | O(log n) |

### Product.java

public class Product {  
 int productId;  
 String productName;  
 String category;  
  
 public Product(int id, String name, String cat) {  
 this.productId = id;  
 this.productName = name;  
 this.category = cat;  
 }  
}

### SearchDemo.java

import java.util.Arrays;  
import java.util.Comparator;  
  
public class SearchDemo {  
 public static void main(String[] args) {  
 Product[] products = {  
 new Product(101, "Laptop", "Electronics"),  
 new Product(102, "Shirt", "Clothing"),  
 new Product(103, "Camera", "Electronics")  
 };  
  
 for (Product p : products) {  
 if (p.productName.equals("Camera")) {  
 System.out.println("Linear Search: " + p.category);  
 }  
 }  
  
 Arrays.sort(products, Comparator.comparing(p -> p.productName));  
  
 int low = 0, high = products.length - 1;  
 String target = "Laptop";  
 while (low <= high) {  
 int mid = (low + high) / 2;  
 int cmp = products[mid].productName.compareTo(target);  
 if (cmp == 0) {  
 System.out.println("Binary Search: " + products[mid].category);  
 break;  
 } else if (cmp < 0) {  
 low = mid + 1;  
 } else {  
 high = mid - 1;  
 }  
 }  
 }  
}

### Analysis

- Linear Search: Simple but slower for large data.  
- Binary Search: Much faster but requires sorted data.  
- Binary search is preferred for large, sorted data sets.

## Exercise 7: Financial Forecasting

Scenario: Predict future values using recursion and memoization.

### Concept of Recursion

Recursion solves problems by dividing them into smaller instances.  
It simplifies problems like mathematical series, search trees, etc.

### Forecast.java

public class Forecast {  
 public static double futureValueRecursive(double value, double rate, int years) {  
 if (years == 0) {  
 return value;  
 }  
 return futureValueRecursive(value, rate, years - 1) \* (1 + rate);  
 }  
  
 public static double futureValueMemo(double value, double rate, int years, double[] memo) {  
 if (years == 0) {  
 return value;  
 }  
 if (memo[years] != 0) {  
 return memo[years];  
 }  
 memo[years] = futureValueMemo(value, rate, years - 1, memo) \* (1 + rate);  
 return memo[years];  
 }  
  
 public static void main(String[] args) {  
 double initial = 1000;  
 double rate = 0.1;  
 int years = 3;  
  
 double result = futureValueRecursive(initial, rate, years);  
 System.out.println("Recursive result: " + result);  
  
 double[] memo = new double[years + 1];  
 double optimizedResult = futureValueMemo(initial, rate, years, memo);  
 System.out.println("Memoized result: " + optimizedResult);  
 }  
}

### Output

Recursive result: 1331.0000000000002  
Memoized result: 1331.0000000000002

### Analysis

| Method | Time Complexity |  
|-------------|------------------|  
| Recursion | O(n) |  
| Memoization | O(n) |  
  
Memoization reduces repeated computation by caching results.