Week - 1: Algorithms & Data Structures

**E-commerce Platform Search Function**

# Understanding the Asymptotic Notations

* **Big O Notation**: Describes the upper bound of an algorithm’s running time or space complexity, helping us understand how the algorithm scales with input size.
* **Best-case**: Minimum time required (example: In linear search, the item found at first position).

**Average-case**: Expected time over all possible scenarios. (example: In linear search, the item is found in the middle of the array or near to that position)

**Worst-case**: Maximum time required (example: In linear Search, the item not found at all, or in the last position).

# Setup

# Create a new Java project using any IDE or a basic text editor with terminal. Inside the project directory, create two java classes: Product.java and SearchDemo.java. In Product.java, define attributes such as:

# productId (int): A unique identifier

# productName (String): The name of the product

# category (String): The category the product belongs to

# Include a constructor, getter/setter methods if required, and a toString() method for displaying product details.

1. **Implementation**

public class ProductSearch {

public static Product linearSearch(Product[] products, int id) {

for (Product p : products) {

if (p.id == id)

return p;

}

return null;

}

public static Product binarySearch(Product[] products, int id) {

int left = 0, right = products.length - 1;

while (left <= right) {

int mid = (left + right) / 2;

if (products[mid].id == id)

return products[mid];

else if (products[mid].id < id)

left = mid + 1;

else

right = mid - 1;

}

return null;

}

1. **Analysis**

* **Linear Search** checks each element in the list one by one until it finds the target or reaches the end. It works on both sorted and unsorted data. Its time complexity is **O(n)**, meaning it can be slow for large datasets. However, it's simple to implement and useful when data isn't sorted.
* **Binary Search**, on the other hand, is much faster with a time complexity of **O(log n)**, but it only works on sorted data. It works by dividing the array into halves and narrowing the search range with each step. This makes it efficient for large datasets, but it requires extra steps to ensure the data is sorted before searching.
* In summary, linear search is better for small or unsorted datasets due to its simplicity, while binary search is more suitable for large and sorted datasets because of its speed.