**Data Structures and Algorithms**

**Exercise 2 – Ecommerce Platform Search Functions**

**Code:**

**1. Asymptotic Notations:**

Big O notation helps in showing the worst case time complexity of any algorithm

It can be used to compare different algorithms and choose the best algorithm for our practices.

**Linear Search Binary Search**

Best case O(1) O(1)

Average case O(n) O(log n)

Worst case O(n) O(log n)

**2. Product class with productID, productName, Category**

package EcommerceSearch;

public class Product{

public int ProductID;

public String productName;

public String Category;

Product(int ProductID, String productName, String Category)

{

this.ProductID=ProductID;

this.productName=productName;

this.Category=Category;

}

}

**3. Implementing Binary Search and Linear Search algorithms with Class**

**ProductSearch**

**Code:**

package EcommerceSearch;

import java.util.\*;

public class ProductSearch

{

public Product[] LinearSearch(Product[] products,String ProductName)

{ Product[] result=new Product[products.length];

int i=0;

for(Product p: products)

{

if(p.productName.toLowerCase().contains(ProductName.toLowerCase())||p.Category.toLowerCase().contains(ProductName.toLowerCase()))

{

result[i]=p ;

i++;

}

}

return result;

}

public Product BinarySearch(Product[] products,String ProductName)

{

Arrays.*sort*(products, Comparator.*comparing*(p -> p.productName.toLowerCase()));

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

while(left<=right)

{ int mid=(left+right)/2;

Int cmp=ProductName.compareToIgnoreCase(products[mid].productName);

if(cmp == 0)

{

return products[mid];

}

else if(cmp<0){

right = mid-1;

}

else {

left= mid+1;

}

}

return null;

}

}

**Tested with a Main class**

package EcommerceSearch;

import java.util.Scanner;

public class Main{

public static void main(String [] args)

{

Product[] products = {new Product(1,"HP Pavilion 14","Laptop"),

new Product(2,"IQOO Neo 9 Pro","Mobile"),

new Product(3, "HP Pavilion 13","Laptop"),

new Product(4,"Dell vostro","Laptop"),

new Product(5,"Realme 6","Mobile")

};

Scanner s=new Scanner(System.*in*);

String name= s.nextLine();

ProductSearch search=new ProductSearch();

Product[] n= search.LinearSearch(products,name);

System.*out*.println("Linear Search:");

for(Product p : n)

{

if(p!=null)

{ System.*out*.println(p!=null? p.ProductID + "\n"+ p.productName +"\n" + p.Category + "\n"+" ":"Not Available");

}

}

Product k=search.BinarySearch(products,name);

System.*out*.println("Binary Search:");

System.*out*.println(k!=null?k.ProductID +"\n"+ k.productName +"\n"+ k.Category:"Not Available");

s.close();

}

}

**4. Analysis:**

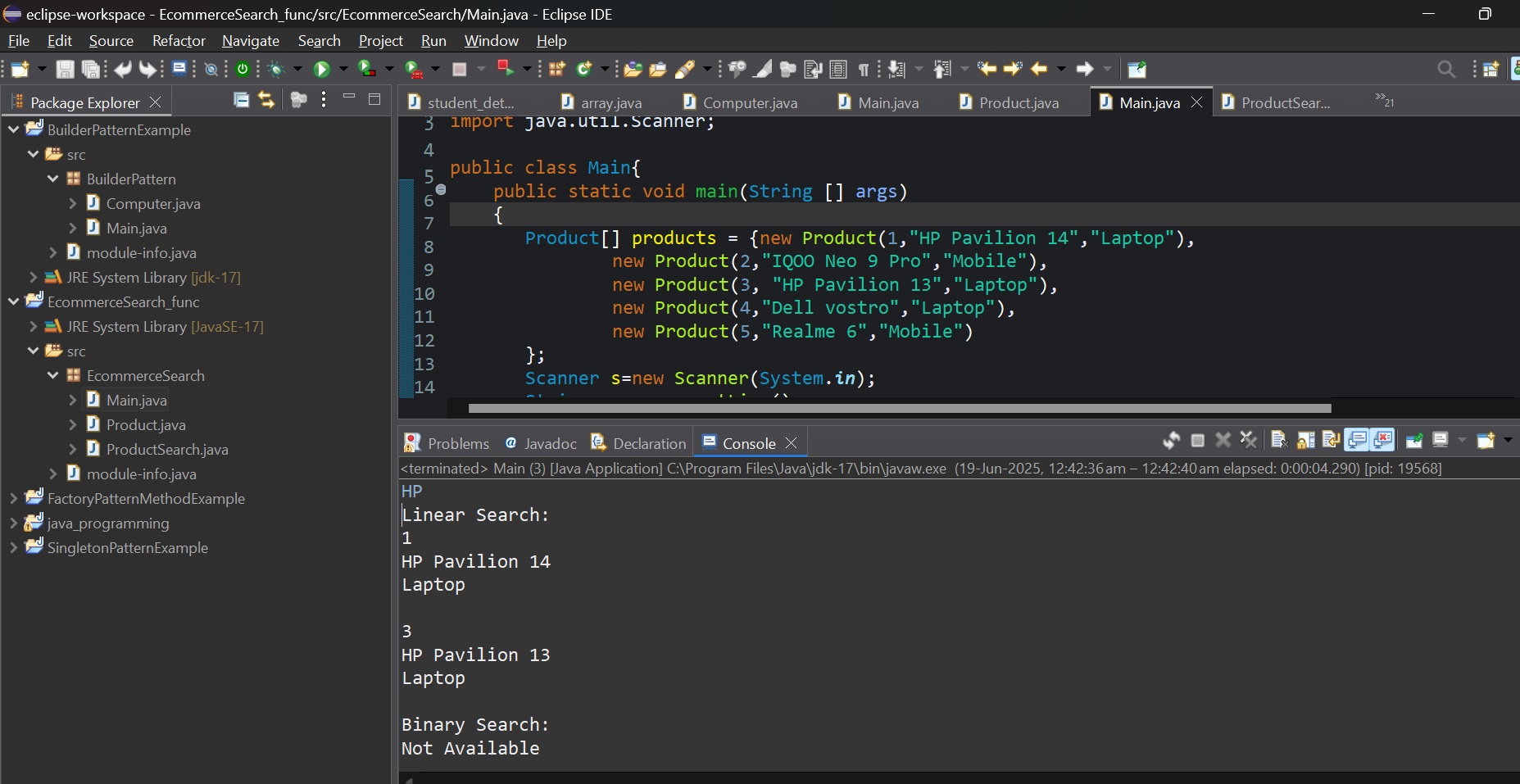
Linear Search time complexity is O(n) as the worst and average case which is greater than the time complexity of Binary Search which is O(log n) for both Worst and average cases.

In terms of time complexity, Binary search is best for searching any element than Linear Search

But, in E-commerce Platform Linear Search is the best algorithm to use cause, it can show all the matches related to the input which are present in the platform that to by searching in all atributes, where as binary search can only search for single element in either of the attributes.

The best algorithm for E-commerce Search Function is Linear Search.

**Output:** User Input is in Blue colour on the console.

****

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**