**Exercise 2: E-commerce Platform Search Function**

**Scenario:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**Steps:**

1. **Understand Asymptotic Notation:**
   * Explain Big O notation and how it helps in analyzing algorithms.
   * Describe the best, average, and worst-case scenarios for search operations.
2. **Setup:**
   * Create a class **Product** with attributes for searching, such as **productId, productName**, and **category**.
3. **Implementation:**
   * Implement linear search and binary search algorithms.
   * Store products in an array for linear search and a sorted array for binary search.
4. **Analysis:**
   * Compare the time complexity of linear and binary search algorithms.
   * Discuss which algorithm is more suitable for your platform and why.

Solution:

Product.java:

**package** product;

**public** **class** Product **implements** Comparable<Product>{

**private** String prodId;

**private** String prodName;

**private** String category;

**public** Product(String prodId,String prodName,String category) {

// **TODO** Auto-generated constructor stub

**this**.prodId=prodId;

**this**.prodName=prodName;

**this**.category=category;

}

**public** String getprodId() {

**return** prodId;

}

**public** String getprodName() {

**return** prodName;

}

**public** String getCategory() {

**return** category;

}

@Override

**public** String toString() {

**return** prodId+" "+prodName+" "+category;

}

@Override

**public** **boolean** equals(Object obj) {

**if**(**this**==obj) **return** **true**;

**if**(obj==**null** || getClass()!=obj.getClass()) **return** **false**;

Product product=(Product)obj;

**return** prodId.equals(product.prodId);

}

@Override

**public** **int** hashCode() {

**return** prodId.hashCode();

}

@Override

**public** **int** compareTo(Product other) {

**return** **this**.prodId.compareTo(other.prodId);

}

}

Lsearch.java:

**package** product;

**import** java.util.\*;

**public** **class** lsearch {

**public** **static** Product linearSearch(List<Product> prods,String sv) {

// **TODO** Auto-generated constructor stub

**for**(Product prod:prods) {

**if**(prod.getprodId().equals(sv)) {

**return** prod;

}

}

**return** **null**;

}

}

Bsearch.java:

**package** product;

**import** java.util.\*;

**public** **class** bsearch {

**public** **static** Product binarySearch(List<Product> sortedprod,String prodId) {

// **TODO** Auto-generated constructor stub

**int** low=0;

**int** high=sortedprod.size()-1;

**while**(low<=high) {

**int** mid=low+(high-low)/2;

Product midp=sortedprod.get(mid);

String midpid=midp.getprodId();

**int** comp=prodId.compareTo(midpid);

**if**(comp==0) {

**return** midp;

}**else** **if**(comp<0) {

high=mid-1;

}**else** {

low=mid+1;

}

}

**return** **null**;

}

}

Ecomm.java:

**package** product;

**import** java.util.\*;

**public** **class** Ecomm {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Product p1=**new** Product("p1","xxx","abc");

Product p2=**new** Product("p11","yyy","abc");

Product p3=**new** Product("p3","zzz","abc");

Product p4=**new** Product("p10","aaa","pqr");

List<Product> lsprod=**new** ArrayList<>();

lsprod.add(p1);

lsprod.add(p2);

lsprod.add(p3);

lsprod.add(p4);

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

Product isPres=lsearch.*linearSearch*(lsprod, "p3");

**if**(isPres!=**null**) {

System.***out***.println("found!");

}**else** {

System.***out***.println("not found.");

}

isPres=lsearch.*linearSearch*(lsprod, "p9");

**if**(isPres!=**null**) {

System.***out***.println("found!");

}**else** {

System.***out***.println("not found.");

}

List<Product> bsprod=**new** ArrayList<>();

bsprod.add(p1);

bsprod.add(p2);

bsprod.add(p3);

bsprod.add(p4);

Collections.*sort*(bsprod);

System.***out***.println("Binary search");

isPres=bsearch.*binarySearch*(bsprod, "p10");

**if**(isPres!=**null**) {

System.***out***.println("found!");

}**else** {

System.***out***.println("not found.");

}

isPres=bsearch.*binarySearch*(bsprod, "p9");

**if**(isPres!=**null**) {

System.***out***.println("found!");

}**else** {

System.***out***.println("not found.");

}

}

}

Output Screenshot:

