

**Solution:**

Big O notation-

Big O notation focuses on finding the worst-case scenario of an algorithm and helps compare the efficiency of different algorithms.

Best case scenario-

* Linear search – O(1)
* Binary search – O(1)

Average case scenario-

* Linear search – O(n/2)
* Binary search – O(log n)

Worst case scenario-

* Linear search – O(n)
* Binary search – O(log n)

Binary search only works on sorted data.

**Code-**

**Ecommerce class-**

package week\_1\_datastructures\_algorithms;

import java.util.\*;

public class ecommerce {

    public static boolean linear(Product[] p,int n,String prname){

        for(int i=0;i<n;i++){

            if(p[i].productname.equals(prname)){

                return true;

            }

        }

        return false;

    }

    public static void sort(Product[] p,int n){

        for(int i=0;i<n-1;i++){

            for(int j=0;j<n-i-1;j++){

                if(p[j].productname.compareTo(p[j+1].productname)>0){

                    Product temp=p[j];

                    p[j]=p[j+1];

                    p[j+1]=temp;

                }

            }

        }

    }

    public static boolean binary(Product[] p,int n,String prname){

        int left=0,right=n-1;

        while(left<=right){

            int mid=(left+right)/2;

            if(p[mid].productname.equals(prname)){

                return true;

            }

            else if(p[mid].productname.compareTo(prname)<0){

                left=mid+1;

            }

            else{

                right=mid-1;

            }

        }

        return false;

    }

    public static void printlist(Product[] p,int n){

        for(int i=0;i<n;i++){

            System.out.println(p[i].productid+">>"+p[i].productname+">>"+p[i].category);

        }

    }

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);

        int n=sc.nextInt();

        Product[] p=new Product[n];

        for(int i=0;i<n;i++){

            int prid=sc.nextInt();

            sc.nextLine();

            String prname=sc.nextLine();

            String prcat=sc.nextLine();

            p[i]=new Product(prid,prname,prcat);

        }

        boolean found=linear(p,n,"mobile");

        if(found){

            System.out.println("Product found");

        }

        else{

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

        }

        printlist(p,n);

        sort(p, n);

        System.out.println("sorted list");

        printlist(p, n);

        boolean bfound=binary(p,n,"mobile");

        if (bfound) {

            System.out.println("Prouduct found using binary");

        }

        else{

            System.out.println("Product not found using binary");

        }

        sc.close();

    }

}

**Product class-**

package week\_1\_datastructures\_algorithms;

public class Product{

    int productid;

    String productname;

    String category;

    Product(int productid, String productname, String category){

        this.productid=productid;

        this.productname=productname;

        this.category=category;

    }

}

**Output-**

