**Exercise 1: Inventory Management System**

**Scenario:**

You are developing an inventory management system for a warehouse. Efficient data storage and retrieval are crucial.

**Steps:**

1. **Understand the Problem:**
   * Explain why data structures and algorithms are essential in handling large inventories.
   * Discuss the types of data structures suitable for this problem.
2. **Setup:**
   * Create a new project for the inventory management system.
3. **Implementation:**
   * Define a class Product with attributes like **productId**, **productName**, **quantity**, and **price**.
   * Choose an appropriate data structure to store the products (e.g., ArrayList, HashMap).
   * Implement methods to add, update, and delete products from the inventory.
4. **Analysis:**
   * Analyze the time complexity of each operation (add, update, delete) in your chosen data structure.

Discuss how you can optimize these operations.

Solutions:

1.

In a storehouse that contains lots of products, we require quick methods to search, update, and sort the inventory. Data structures that are efficient save time when handling data and prevent slow performance when the inventory increases.

2,3.

Source codes:

Product .java

public class Product {

    private String productId;

    private String productName;

    private int quantity;

    private double price;

    public Product(String productId, String productName, int quantity, double price) {

        this.productId = productId;

        this.productName = productName;

        this.quantity = quantity;

        this.price = price;

    }

    public String getProductId() {

        return productId;

    }

    public String getProductName() {

        return productName;

    }

    public int getQuantity() {

        return quantity;

    }

    public double getPrice() {

        return price;

    }

    public void setQuantity(int quantity) {

        this.quantity = quantity;

    }

    @Override

    public String toString() {

        return "ID: " + productId + ", Name: " + productName + ", Qty: " + quantity + ", Price: " + price;

    }

}

Inventory.java

import java.util.HashMap;

public class Inventory {

    private HashMap<String, Product> products = new HashMap<>();

    public void addProduct(Product product) {

        products.put(product.getProductId(), product);

    }

    public void updateProduct(String productId, Product updatedProduct) {

        products.put(productId, updatedProduct);

    }

    public void deleteProduct(String productId) {

        products.remove(productId);

    }

    public Product getProduct(String productId) {

        return products.get(productId);

    }

    @Override

    public String toString() {

        if (products.isEmpty()) {

            return "Inventory is empty.";

        }

        StringBuilder sb = new StringBuilder("Inventory List:\n");

        for (Product p : products.values()) {

            sb.append(p).append("\n");

        }

        return sb.toString();

    }

}

Management.java

import java.util.Scanner;

public class Management {

    private static Inventory inventory = new Inventory();

    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

        boolean exit = false;

        while (!exit) {

            showMenu();

            int choice = scanner.nextInt();

            scanner.nextLine();

            switch (choice) {

                case 1:

                    addProduct();

                    break;

                case 2:

                    updateProduct();

                    break;

                case 3:

                    deleteProduct();

                    break;

                case 4:

                    getProduct();

                    break;

                case 5:

                    printInventory();

                    break;

                case 6:

                    exit = true;

                    break;

                default:

                    System.out.println("Invalid choice. Please enter a valid choice.");

            }

        }

    }

    private static void showMenu() {

        System.out.println("\nInventory Management System");

        System.out.println("1. Add Product");

        System.out.println("2. Update Product");

        System.out.println("3. Delete Product");

        System.out.println("4. Get Product");

        System.out.println("5. Print Inventory");

        System.out.println("6. Exit");

        System.out.print("Enter your choice: ");

    }

    private static void addProduct() {

        System.out.print("Enter Product ID: ");

        String productId = scanner.nextLine();

        System.out.print("Enter Product Name: ");

        String productName = scanner.nextLine();

        System.out.print("Enter Quantity: ");

        int quantity = scanner.nextInt();

        System.out.print("Enter Price: ");

        double price = scanner.nextDouble();

        scanner.nextLine();

        Product product = new Product(productId, productName, quantity, price);

        inventory.addProduct(product);

        System.out.println("Product added successfully.");

    }

    private static void updateProduct() {

        System.out.print("Enter Product ID to update: ");

        String productId = scanner.nextLine();

        System.out.print("Enter New Product Name: ");

        String productName = scanner.nextLine();

        System.out.print("Enter New Quantity: ");

        int quantity = scanner.nextInt();

        System.out.print("Enter New Price: ");

        double price = scanner.nextDouble();

        scanner.nextLine();

        Product updatedProduct = new Product(productId, productName, quantity, price);

        inventory.updateProduct(productId, updatedProduct);

        System.out.println("Product updated successfully.");

    }

    private static void deleteProduct() {

        System.out.print("Enter Product ID to delete: ");

        String productId = scanner.nextLine();

        inventory.deleteProduct(productId);

        System.out.println("Product deleted successfully.");

    }

    private static void getProduct() {

        System.out.print("Enter Product ID to retrieve: ");

        String productId = scanner.nextLine();

        Product product = inventory.getProduct(productId);

        if (product != null) {

            System.out.println("Retrieved Product: " + product);

        } else {

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

        }

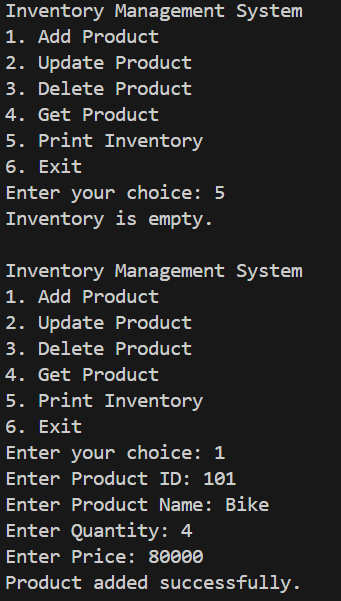
    }

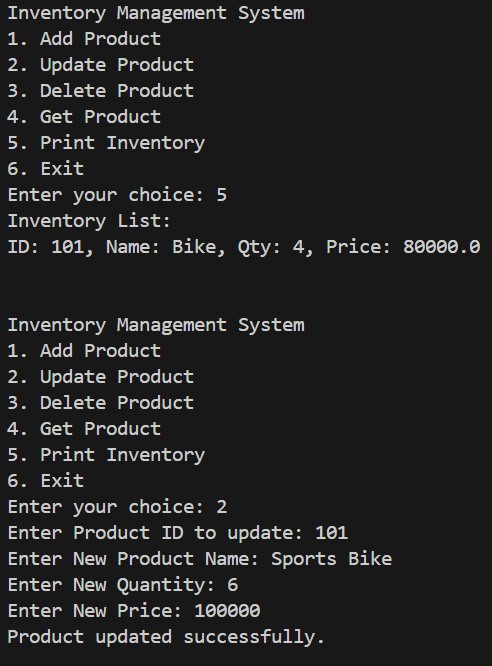
    private static void printInventory() {

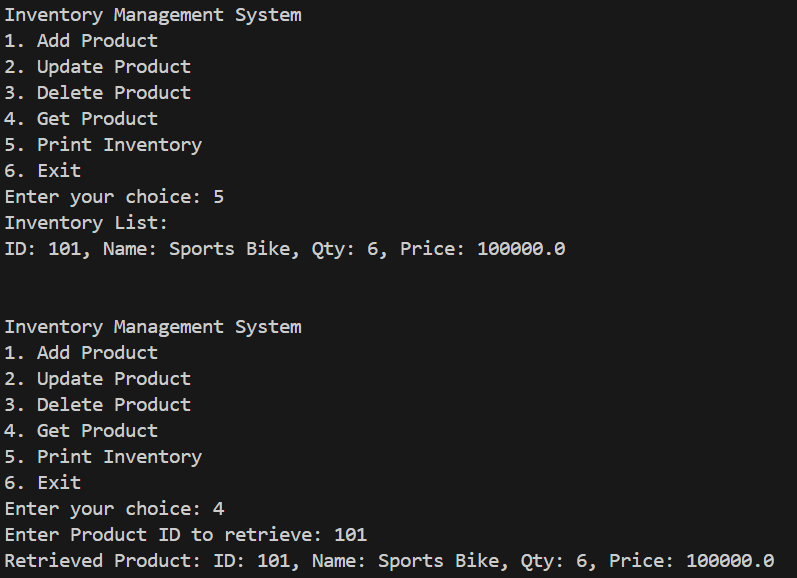
        System.out.println(inventory);

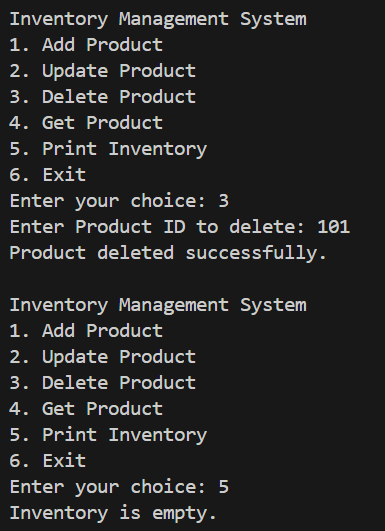
    }

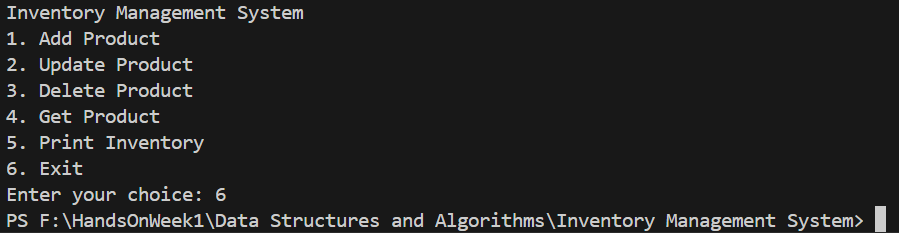
}

Outputs:









4.

In the inventory system, we utilize a HashMap as the data structure to hold products, where the productId is used as the key. The average time complexity of the fundamental operations add, update, and delete is O(1), i.e., each operation takes constant time in proportion to the size of the inventory. This is due to the fact that HashMap employs a hashing technique to directly access the address location of the key value pair in memory. To further refine these operations, we make sure that the productId is distributed and unique to prevent hash collisions that might reduce performance down to O(n) in the worst situation. Moreover, operations might be further optimized in large scale systems by utilizing concurrent maps, lazy loading, or indexing solutions if connecting to a database or persistent storage. In general, HashMap ensures great efficiency in handling large inventories for real time applications.