Object Oriented Programming with Java Lab

Digital Assignment 4

22MCA0139

Rajat Singh

Question) Consider a scenario where you are developing an online shopping platform. Each product on the platform has attributes such as product ID, name, category, and price. You need to implement this scenario using either an ArrayList or LinkedList in Java.

Your program should be capable of:

1. Adding a new product to the inventory.

2. Deleting a product using its product ID.

3. Updating the price of a specific product.

4. Generating a list of all products within a certain price range.

5. Search product using product ID/name.

6. Exit.

Create a menu for the above options. Articulate your choice between ArrayList and LinkedList, explain the methods used for each operation, and evaluate the efficiency of your chosen data structure in terms of time complexity. Bear in mind that your design must be adaptable for the future inclusion of new product attributes and categories. (10 marks)

Solution:

Code:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Product {
    private String productId;
    private String name;
    private String category;
    private double price;
```

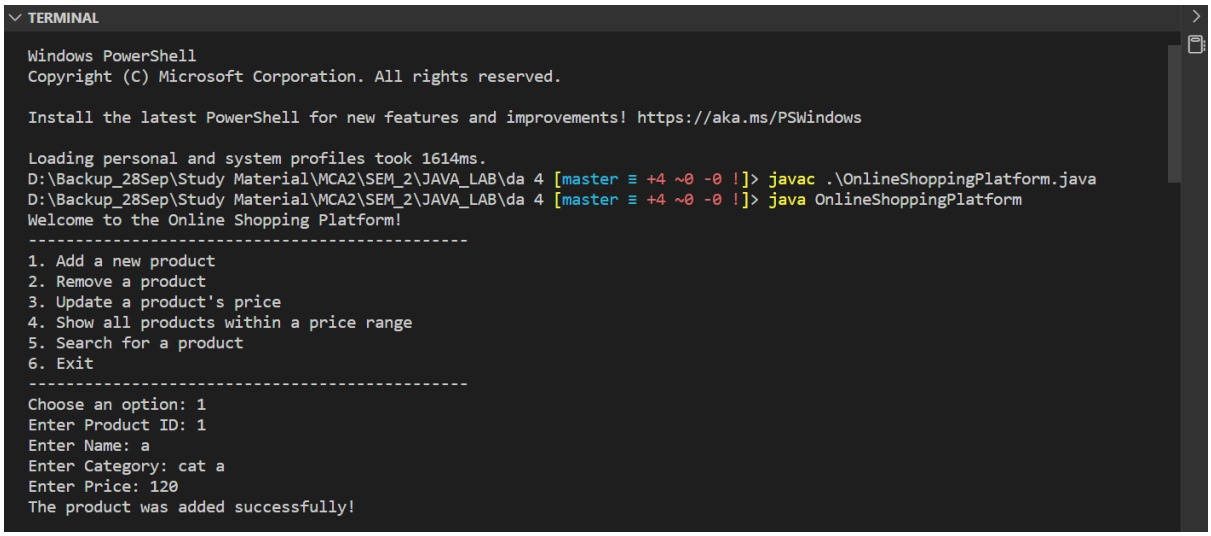
```
    public Product(String productId, String name, String category, double
price) {
        this.productId = productId;
        this.name = name;
        this.category = category;
        this.price = price;
    }

    // Getters and setters

    public String getProductId() {
        return productId;
    }

    public void setProductId(String productId) {
        this.productId = productId;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCategory() {
        return category;
    }

    public void setCategory(String category) {
        this.category = category;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    @Override
    public String toString() {
        return "Product ID: " + productId + ", Name: " + name + ", Category: "
+ category + ", Price: " + price;
    }
```

```
}

public class OnlineShoppingPlatform {
    private List<Product> inventory;

    public OnlineShoppingPlatform() {
        inventory = new ArrayList<>();
    }

    public void addProduct(Product product) {
        inventory.add(product);
        System.out.println("The product was added successfully!");
    }

    public void removeProduct(String productId) {
        boolean removed = false;
        for (Product product : inventory) {
            if (product.getProductId().equals(productId)) {
                inventory.remove(product);
                removed = true;
                break;
            }
        }
        if (removed) {
            System.out.println("The product was removed successfully!");
        } else {
            System.out.println("Product not found!");
        }
    }

    public void updateProductPrice(String productId, double newPrice) {
        boolean updated = false;
        for (Product product : inventory) {
            if (product.getProductId().equals(productId)) {
                product.setPrice(newPrice);
                updated = true;
                break;
            }
        }
        if (updated) {
            System.out.println("The price was updated successfully!");
        } else {
            System.out.println("Product not found!");
        }
    }

    public void showProductsInRange(double minPrice, double maxPrice) {
        System.out.println("Products within price range:");
```

```
        for (Product product : inventory) {
            if (product.getPrice() >= minPrice && product.getPrice() <=
maxPrice) {
                System.out.println("- " + product.toString());
            }
        }
    }

    public void searchProduct(String searchKey) {
        System.out.println("Search results:");
        for (Product product : inventory) {
            if (product.getProductId().equals(searchKey) ||
product.getName().equalsIgnoreCase(searchKey)) {
                System.out.println("- " + product.toString());
            }
        }
    }

    public static void main(String[] args) {
        OnlineShoppingPlatform platform = new OnlineShoppingPlatform();
        Scanner scanner = new Scanner(System.in);
        int choice;

        System.out.println("Welcome to the Online Shopping Platform!");
        System.out.println("-----------------------------------------------");

        do {
            System.out.println("1. Add a new product");
            System.out.println("2. Remove a product");
            System.out.println("3. Update a product's price");
            System.out.println("4. Show all products within a price range");
            System.out.println("5. Search for a product");
            System.out.println("6. Exit");
            System.out.println("-----------------------------------------------
-");
            System.out.print("Choose an option: ");
            choice = scanner.nextInt();
            scanner.nextLine();

            switch (choice) {
                case 1:
                    System.out.print("Enter Product ID: ");
                    String productId = scanner.nextLine();
                    System.out.print("Enter Name: ");
                    String name = scanner.nextLine();
                    System.out.print("Enter Category: ");
                    String category = scanner.nextLine();
                    System.out.print("Enter Price: ");
```

```
                    double price = scanner.nextDouble();
                    scanner.nextLine();
                    platform.addProduct(new Product(productId, name, category,
price));
                    break;
                case 2:
                    System.out.print("Enter Product ID to be removed: ");
                    String removeId = scanner.nextLine();
                    platform.removeProduct(removeId);
                    break;
                case 3:
                    System.out.print("Enter Product ID for price update: ");
                    String updateId = scanner.nextLine();
                    System.out.print("Enter new price: ");
                    double newPrice = scanner.nextDouble();
                    scanner.nextLine();
                    platform.updateProductPrice(updateId, newPrice);
                    break;
                case 4:
                    System.out.print("Enter minimum price: ");
                    double minPrice = scanner.nextDouble();
                    System.out.print("Enter maximum price: ");
                    double maxPrice = scanner.nextDouble();
                    scanner.nextLine();
                    platform.showProductsInRange(minPrice, maxPrice);
                    break;
                case 5:
                    System.out.print("Enter product ID or name to search: ");
                    String searchKey = scanner.nextLine();
                    platform.searchProduct(searchKey);
                    break;
                case 6:
                    System.out.println("Thank you for using the Online
Shopping Platform!");
                    break;
                default:
                    System.out.println("Invalid choice! Please try again.");
                    break;
            }
            System.out.println();
        } while (choice != 6);
    }
}
```

Output:




```
1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
-----------------------------------------------
Choose an option: 1
Enter Product ID: 2
Enter Name: b
Enter Category: cat b
Enter Price: 145
The product was added successfully!

1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
-----------------------------------------------
Choose an option: 4
Enter minimum price: 100
Enter maximum price: 150
Products within price range:
- Product ID: 1, Name: a, Category: cat a, Price: 120.0
- Product ID: 2, Name: b, Category: cat b, Price: 145.0
```

```
1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
------------------------------------------------
Choose an option: 4
Enter minimum price: 100
Enter maximum price: 130
Products within price range:
- Product ID: 1, Name: a, Category: cat a, Price: 120.0

1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
------------------------------------------------
Choose an option: 3
Enter Product ID for price update: 1
Enter new price: 125
The price was updated successfully!
```

```
1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
-----------------------------------------------
Choose an option: 4
Enter minimum price: 122
Enter maximum price: 150
Products within price range:
- Product ID: 1, Name: a, Category: cat a, Price: 125.0
- Product ID: 2, Name: b, Category: cat b, Price: 145.0

1. Add a new product
2. Remove a product
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
-----------------------------------------------
Choose an option: 1
Enter Product ID: 3
Enter Name: cat c
Enter Category: cat cat c
Enter Price: 93
The product was added successfully!
```

```
3. Update a product's price
4. Show all products within a price range
5. Search for a product
6. Exit
-----------------------------------------------
Choose an option: 6
Thank you for using the Online Shopping Platform!

D:\Backup_28Sep\Study Material\MCA2\SEM_2\JAVA_LAB\da 4 [master ≡ +4 ~0 -0 !]>
```

Explanation:

First, we have the Product class. This class represents a single product in the online shopping platform and has attributes such as productId, name, category, and price. These attributes are encapsulated within the class using private

access modifiers, and there are corresponding getter and setter methods for each attribute.

The Product class also overrides the toString() method to provide a custom string representation of a product object. This method is used to display product information when needed.

Next, we have the OnlineShoppingPlatform class. This class represents the online shopping platform itself and manages the inventory of products. The inventory is implemented using an ArrayList of Product objects.

The OnlineShoppingPlatform class provides several methods to perform various operations on the inventory.

The addProduct method allows adding a new product to the inventory. It takes the necessary information for a product (product ID, name, category, and price) as parameters. Inside the method, a new Product object is created with the provided information, and it is added to the inventory list using the add method of ArrayList.

The removeProduct method allows removing a product from the inventory based on its product ID. It takes the product ID as a parameter. The method iterates over the inventory list using an enhanced for loop, and for each product, it checks if the product's ID matches the given ID. If a match is found, the product is removed from the list using the remove method of ArrayList.

The updateProductPrice method allows updating the price of a specific product in the inventory. It takes the product ID and the new price as parameters. Similar to the removeProduct method, it iterates over the inventory list and searches for the product with the given ID. If a match is found, the price of the product is updated using the corresponding setter method.

The showProductsInRange method generates a list of all products within a certain price range. It takes the minimum and maximum prices as parameters. The method iterates over the inventory list and checks if each product's price falls within the specified range. If a product satisfies the condition, it is displayed using the toString method.

The searchProduct method allows searching for a product in the inventory based on the product ID or name. It takes a search key as a parameter. The method iterates over the inventory list and checks if the product's ID or name

matches the search key. If a match is found, the product is displayed using the toString method.

In the main method, the program starts by creating an instance of the OnlineShoppingPlatform class and a Scanner object to read user input. The program then presents a menu to the user, and based on the user's choice, it calls the appropriate methods of the OnlineShoppingPlatform class to perform the corresponding operation.

The program continues to display the menu until the user chooses to exit by entering option 6.

In terms of efficiency, using an ArrayList for the inventory provides several benefits. Adding a new product to the end of the list has an amortized time complexity of O(1), as ArrayList dynamically resizes itself to accommodate new elements. However, removing a product or updating its price requires searching for the product in the list, resulting in a time complexity of O(n) since the list may need to be traversed entirely in the worst case.

Overall, this implementation using an ArrayList provides efficient operations for most scenarios in an online shopping platform, considering the trade-off between insertion efficiency and searching efficiency. However, if frequent removal or insertion of products is expected in large inventories, a LinkedList may be a better choice as it provides constant time complexity for both removal and insertion at any position.