МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

# Отчет по лабораторной работе №3

Специальность ПО9(з)

Выполнил
Д. Н. Кухарев,
студент группы ПО9

Проверил
А. А. Крощенко,
ст. преп. кафедры ИИТ,
«___k_________2024 г.

Брест 2024

Цель работы: научиться создавать и использовать классы в программах на языке программирования Java.

**Вариант 9**

**Задание 1. Реализовать простой класс.**

**Требования к выполнению**

- **Реализовать пользовательский класс по варианту.**
- **Создать другой класс с методом main, в котором будут находится примеры использования пользовательского класса.**

**Для каждого класса**

- **Создать поля классов**
- **Создать методы классов**
- **Добавьте необходимые get и set методы (по необходимости)**
- **Укажите соответствующие модификаторы видимости**
- **Добавьте конструкторы**
- **Переопределить методы toString() и equals();**

**Множество вещественных чисел переменной мощности – Предусмотреть возможность пересечения двух множеств, вывода на печать элементов множества, а так же метод, определяющий, принадлежит ли указанное значение множеству. Класс должен содержать методы, позволяющие добавлять и удалять элемент в/из множества. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализацию множества осуществить на базе структуры ArrayList. Реализовать метод equals, выполняющий сравнение объектов данного типа.**

Выполнение:

**Код программы**

**Main.java:**

```
import java.util.ArrayList;


public class Main {
    public static void main(String[] args) {
        ArrayList<Double> r = new ArrayList<>();
        r.add(21.3); r.add(22.5);
        VariablePowerRealNumbers set1 = new VariablePowerRealNumbers();
        VariablePowerRealNumbers set2 = new VariablePowerRealNumbers();
        VariablePowerRealNumbers set3 = new VariablePowerRealNumbers();
        VariablePowerRealNumbers set4 = new VariablePowerRealNumbers();
        set1.add(50.3); set1.add(0.3); set1.add(20.3); set1.add(10.3); set1.add(40.3);
        set2.add(50.3); set2.add(0.5); set2.add(20.3); set2.add(10.3); set2.add(40.4);
        set3.add(10); set3.add(20); set3.add(30); set3.add(40); set3.add(50);
        set4.add(10); set4.add(20); set4.add(30); set4.add(40); set4.add(50);


        if(!set1.equals(set2)){
            System.out.println("Sets are not equal");
            System.out.println(set1.Intersection(set2).toString());
        }else{
            System.out.println("Sets are equal");
            System.out.println(set1.toString());
        }

        if(!set3.equals(set4)){
            System.out.println("Sets are not equal");
            System.out.println(set3.Intersection(set4).toString());
        }else{
            System.out.println("Sets are equal");
            System.out.println(set3.toString());
        }
    }
}
```

**VariablePowerRealNumbers.java:**

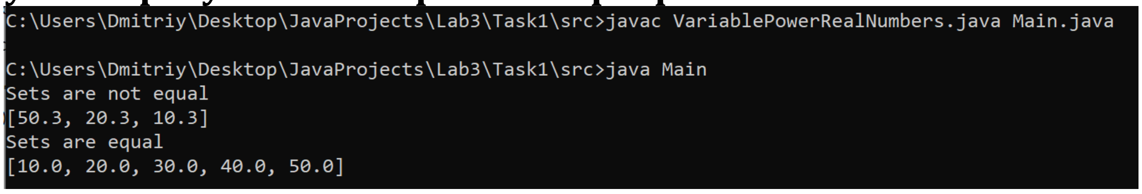
```
import java.util.ArrayList;


public class VariablePowerRealNumbers {
    private ArrayList<Double> set;
    public VariablePowerRealNumbers(){
        set = new ArrayList<Double>();
    }
    public double get(int index){
        return set.get(index);
    }
    public void add(double value){
        set.add(value);
    }
    public void remove(int index){
        set.remove(index);
    }
    public int size(){
        return set.size();
    }
    public VariablePowerRealNumbers Intersection(VariablePowerRealNumbers set2){
        VariablePowerRealNumbers set1 = new VariablePowerRealNumbers();
        set1 = ConvertToVPRN(set);
        int i = 0;
        int sizeof_s1 = set1.size();
        int sizeof_s2 = set2.size();
    if(sizeof_s1 > sizeof_s2){
        while (i < sizeof_s1) {
            if (set2.Find(set1.get(i)) > -1) {
                i++;
            } else {
                set1.remove(i);
                sizeof_s1--;
            }
        }
        return set1;
    }else{
        while (i < sizeof_s2) {
            if (set1.Find(set2.get(i)) > -1) {
                i++;
            } else {
                set2.remove(i);
                sizeof_s2--;
            }
        }
        return set2;
    }
}
public VariablePowerRealNumbers ConvertToVPRN(ArrayList<Double> array){
    VariablePowerRealNumbers converted = new VariablePowerRealNumbers();
    for(int i = 0; i < array.size(); ++i){
        converted.add(array.get(i));
    }
    return converted;
}
public int Find(double value){
    int index = set.indexOf(value);
    if(index < 0){
        return -1;
    }else{
        return index;
    }
}
    @Override
public boolean equals(Object obj){
    if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }
        VariablePowerRealNumbers other = (VariablePowerRealNumbers) obj;
        if (set.size() != other.set.size()) {
            return false;
        }
        for (int i = 0; i < set.size(); i++) {
            if (!set.get(i).equals(other.set.get(i))) {
                return false;
            }
        }
        return true;
    }
    @Override
    public String toString() {//Вывод на печать
        String out ="[";
        for(int i = 0; i < set.size(); ++i){
            if(i < set.size()-1){
                out += set.get(i) + ", ";
            }else{
                out += set.get(i) + "]";
            }
        }
        return out;
    }
}
```

**Спецификация ввода**

>java Main

**Рисунки с результатами работы программы**



```
C:\Users\Dmitriy\Desktop\JavaProjects\Lab3\Task1\src>javac VariablePowerRealNumbers.java Main.java

C:\Users\Dmitriy\Desktop\JavaProjects\Lab3\Task1\src>java Main
Sets are not equal
[50.3, 20.3, 10.3]
Sets are equal
[10.0, 20.0, 30.0, 40.0, 50.0]
```

**Задание 2. Автоматизированная система склада**

**Написать программу для моделирования автоматизированного склада. На складе хранится различная продукция (Product). Каждая**

**продукция характеризуется следующей информацией:**

- **id;**
- **Наименование;**
- **UPC (штрих-код);**
- **Производитель;**
- **Цена;**
- **Срок хранения;**
- **Количество.**

**Программа должна иметь следующий функционал:**

- **Генерация списка продукции на складе;**
- **Предоставлять список товаров для заданного наименования;**
- **Предоставлять список товаров для заданного наименования, цена которых не превосходит заданную;**
- **Предоставлять список товаров, срок хранения которых истек.**

Выполнение:

**Код программы**

**Main.java:**

```
import java.time.LocalDate;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        if(args.length == 0){
            System.out.println("Warehouse name can't be empty!");
            System.exit(1);
        }
        Warehouse storage1 = new Warehouse(args[0]);
        ManageStorage(storage1);
    }
    private static int ManageStorage(Warehouse storage){
        Scanner in = new Scanner(System.in);
        int choose = 0;
        while(true){
            ShowName(storage.getName());
            Menu();
            try{
                choose = in.nextInt();
            }catch (Exception ex){
                System.out.println(ex.getMessage());
                in.nextLine();
                ManageStorage(storage);
            }
            in.nextLine();
            switch (choose){
                case Const.EXIT:
```

```
                System.exit(Const.EXIT);
                break;
            case Const.CHANGE_NAME:
                ShowName(storage.getName());
                ChangeName(storage);
                ManageStorage(storage);
                break;
            case Const.ADD_PRODUCT:
                ShowName(storage.getName());
                AddProduct(storage);
                ManageStorage(storage);
                break;
            case Const.SHOW_ALL:
                ShowName(storage.getName());
                ShowAll(storage);
                ManageStorage(storage);
                break;
            case Const.SHOW_BY_NAME:
                ShowName(storage.getName());
                ShowByName(storage);
                ManageStorage(storage);
                break;
            case Const.SHOW_BY_NAME_AND_PRICE:
                ShowName(storage.getName());
                ShowByNamePrice(storage);
                ManageStorage(storage);
                break;
            case Const.SHOW_EXPIRED:
                ShowName(storage.getName());
                ShowExpired(storage);
                ManageStorage(storage);
                break;
            case Const.BACK:
                ShowName(storage.getName());
                ManageStorage(storage);
                break;
            default:
                System.out.println("No such action");
                continue;
        }
        break;
    }
    return Const.ALL_PROCESSED;
}
private static void Menu(){
        System.out.println("1 - Change name;\n2 - Add product\n3 - Show all products\n" +
                "4 - Show products list with selected name\n" +
                "5 - Show products list with selected name with price lower then selected\n" +
                "6 - Show products with expired shelf life date\n7 - Back\n0 - Exit\nChoose action: ");
    }
    private static void ChangeName(Warehouse storage){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter new name: " );
        storage.setName(in.nextLine());
    }
    private static void AddProduct(Warehouse storage){
        Scanner in = new Scanner(System.in);
        Product prod;
        System.out.println("Choose action:\n\t1 - Auto product\n\t2 - Enter name, manufacturer and price\n\t3 -
Enter all data");
        int choose = 0;
        try{
            choose = in.nextInt();
        }catch(Exception ex){
            System.out.println(ex.getMessage());
            ManageStorage(storage);
        }
        in.nextLine();
        String name, manufacturer;
        int price = 0;
        LocalDate date_of_manufacture, shelf_time;
        int year_of_manufacture = 0, month_of_manufacture = 0, day_of_manufacture = 0;
        int shelf_year = 1, shelf_month = 1, shelf_day = 1;
        switch (choose){
            case Const.AUTO:
                prod = new Product();
                storage.addProduct(prod);
                break;
            case Const.MINIMAL:
                ShowName(storage.getName());
                System.out.print("Enter product name: "); name = in.nextLine();
                System.out.print("Enter product manufacturer: "); manufacturer = in.nextLine();
                System.out.print("Enter product price: "); price = in.nextInt();
                prod = new Product(name, manufacturer, price);
                storage.addProduct(prod);
                break;
            case Const.ALL:
                ShowName(storage.getName());
                System.out.print("Enter product name: "); name = in.nextLine();
                System.out.print("Enter product manufacturer: "); manufacturer = in.nextLine();
                System.out.print("Enter product price: ");
                try{
                   price = in.nextInt();
                }catch(Exception ex){
                   ex.getMessage();
                   in.nextLine();
                   AddProduct(storage);
                }
                System.out.print("\t- Enter year of manufacture: ");
                if (in.hasNextInt()) {
                   year_of_manufacture = in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                System.out.print("\t- Enter month of manufacture: ");
                if (in.hasNextInt()) {
                   month_of_manufacture = in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                System.out.print("\t- Enter day of manufacture: ");
                if (in.hasNextInt()) {
                   day_of_manufacture = in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                date_of_manufacture = LocalDate.of(year_of_manufacture, month_of_manufacture,
day_of_manufacture);
                System.out.print("\t- Enter years of shelf time: ");
                if (in.hasNextInt()) {
                   shelf_year += in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                System.out.print("\t- Enter months of shelf time: ");
                if (in.hasNextInt()) {
                   shelf_month += in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                System.out.print("\t- Enter days of shelf time: ");
                if (in.hasNextInt()) {
                   shelf_day += in.nextInt();
                } else {
                   System.out.println("Invalid input. Please enter an integer value.");
                   AddProduct(storage);
                }
                shelf_time = LocalDate.of(shelf_year, shelf_month, shelf_day);

                prod = new Product(name, manufacturer, price, date_of_manufacture, shelf_time);
                storage.addProduct(prod);
                break;
            default:
                System.out.println("No such action");
                ManageStorage(storage);
                break;
        }
    }
    private static void ShowAll(Warehouse storage){
        Scanner in = new Scanner(System.in);
        storage.ShowList();
        String name = in.nextLine();
    }
    private static void ShowByName(Warehouse storage){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter name: ");
        String name = in.nextLine();
        storage.ShowListOf(name);
        name = in.nextLine();
    }
    private static void ShowByNamePrice(Warehouse storage){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter name: ");
        String name = in.nextLine();
        System.out.print("Enter price: ");
        int price = 0;
        try{
            price = in.nextInt();
        }catch (Exception ex){
            System.out.println(ex.getMessage());
            in.nextLine();
            ShowByNamePrice(storage);
        }
        storage.ShowListOf(name, price);
        name = in.nextLine();
        name = in.nextLine();
    }
```

```java
    private static void ShowExpired(Warehouse storage){
        Scanner in = new Scanner(System.in);
        storage.ExpiredShelfLifeList();
        String name = in.nextLine();
    }
    public static void ShowName(String storage_name){
        //System.out.print("\033[H\033[J"); //it has to clear the console, but it didn't...
        System.out.println(storage_name);
    }
}
```

**Product.java:**

```java
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class Product {
    private static int nextId = Const.INITIALIZE_INT;
    private int id;
    private String name;
    private String UPC;
    private String manufacturer;
    private int price;
    private LocalDate date_of_manufacture;
    private LocalDate shelf_time;

    public Product(){
        id = nextId;
        nextId++;
        name = Const.def_name;
        UPC = MakeUPC();
        manufacturer = Const.def_manufacturer;
        price = Const.def_price;
        date_of_manufacture = LocalDate.now();
        shelf_time = Const.def_shelf_time;
    }
    public Product(String name, String manufacturer, int price){
        id = nextId;
        nextId++;
        this.name = name;
        UPC = MakeUPC();
        this.manufacturer = manufacturer;
        this.price = price;
        date_of_manufacture = LocalDate.now();
        shelf_time = Const.def_shelf_time;
    }
    public Product(String name, String manufacturer, int price, LocalDate date_of_manufacture, LocalDate
shelf_time){
        id = nextId;
        nextId++;
        this.name = name;
        UPC = MakeUPC();
        this.manufacturer = manufacturer;
        this.price = price;
        this.date_of_manufacture = date_of_manufacture;
        this.shelf_time = shelf_time;
    }

    public int getId(){
        return id;
    }
    public int getAmount(){
        return nextId;
    }
    public String getName(){
        return name;
    }
    public String getUPC(){
        return UPC;
    }
    public String getManufacturer(){
        return manufacturer;
    }
    public int getPrice(){
        return price;
    }
    public String getManufactureDate(){
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy.MM.dd");
        return date_of_manufacture.format(formatter);
    }
    public String getShelf_time(){
        String shelf = (shelf_time.getYear()-1) + "." + (shelf_time.getMonthValue()-1) + "." +
(shelf_time.getDayOfMonth()-1);
        return shelf;
    }
    public void setName(String name){
        this.name = name;
    }
    public void setManufacturer(String manufacturer){
        this.manufacturer = manufacturer;
    }
    public void setPrice(int price){
        this.price = price;
    }
    public void setManufactureDate(LocalDate date_of_manufacture){
        this.date_of_manufacture = date_of_manufacture;
    }
    public void setShelf_time(LocalDate shelf_time){
        this.shelf_time = shelf_time;
    }
    public boolean isExpired(){
        LocalDate now = LocalDate.now();
        LocalDate manufacture = date_of_manufacture;

        date_of_manufacture.plusYears(shelf_time.getYear()-1);
        date_of_manufacture.plusMonths(shelf_time.getMonthValue()-1);
        date_of_manufacture.plusDays(shelf_time.getDayOfMonth()-1);

        if(now.isAfter(manufacture)){
            return false;
        }else{
            return true;
        }
    }
    private String MakeUPC(){
        LocalDateTime now = LocalDateTime.now();
        return now.format(DateTimeFormatter.ofPattern("yyyyMMddHHmmssSSS"+id));
    }
}
```

**Warehouse.java:**

```
import java.util.ArrayList;

public class Warehouse {
    ArrayList<Product> products;
    private String warehouse_name;
    public Warehouse(String name){
        warehouse_name = name;
        products = new ArrayList<Product>();
    }
    public String getName(){
        return warehouse_name;
    }
    public void setName(String name){
        this.warehouse_name = name;
    }
    public void addProduct(Product product){
        products.add(product);
}
public void ShowList(){
   System.out.println("All products: ");
   for(int i = 0; i < products.size(); ++i){
      System.out.println(i+"." + products.get(i).getName() +
            "\n\tUPC: " + products.get(i).getUPC() +
            "\n\tManufacturer: " + products.get(i).getManufacturer()+
            "\n\tPrice: " + products.get(i).getPrice() +
            "\n\tDate of manufacture: " + products.get(i).getManufactureDate() +
            "\n\tShelf life: " + products.get(i).getShelf_time());
   }
   System.out.println("Total: " + products.size());
}
public void ShowListOf(String name){
   int total = 0;
   System.out.println("Products with name '" + name + "': ");
   for(int i = 0; i < products.size(); ++i){
      if(products.get(i).getName().equals(name)){
         ++total;
         System.out.println(i+"." + products.get(i).getName() +
               "\n\tUPC: " + products.get(i).getUPC() +
               "\n\tManufacturer: " + products.get(i).getManufacturer()+
               "\n\tPrice: " + products.get(i).getPrice() +
               "\n\tDate of manufacture: " + products.get(i).getManufactureDate() +
               "\n\tShelf life: " + products.get(i).getShelf_time());
      }
   }
   System.out.println("Total: " + total);
}
public void ShowListOf(String name, int max_price){
   int total = 0;
   System.out.println("Products with name '" + name + "' and price under " + max_price + ": ");
   for(int i = 0; i < products.size(); ++i){
      if(products.get(i).getName().equals(name) && products.get(i).getPrice() <= max_price){
         ++total;
         System.out.println(i+"." + products.get(i).getName() +
               "\n\tUPC: " + products.get(i).getUPC() +
               "\n\tManufacturer: " + products.get(i).getManufacturer()+
               "\n\tPrice: " + products.get(i).getPrice() +
               "\n\tDate of manufacture: " + products.get(i).getManufactureDate() +
               "\n\tShelf life: " + products.get(i).getShelf_time());
      }
   }
   System.out.println("Total: " + total);
}
```

```
    public void ExpiredShelfLifeList(){
        int total = 0;
        System.out.println("Expired products: ");
        for(int i = 0; i < products.size(); ++i){
            if(!products.get(i).isExpired()){
                ++total;
                System.out.println(i+"." + products.get(i).getName() +
                        "\n\tUPC: " + products.get(i).getUPC() +
                        "\n\tManufacturer: " + products.get(i).getManufacturer()+
                        "\n\tPrice: " + products.get(i).getPrice() +
                        "\n\tDate of manufacture: " + products.get(i).getManufactureDate() +
                        "\n\tShelf life: " + products.get(i).getShelf_time());
            }
        }
        System.out.println("Total: " + total);
    }
}
```

**Const.java:**

```
import java.time.LocalDate;

public class Const {
    final public static int INITIALIZE_INT = 0;
    final public static int ALL_PROCESSED = 0;
    //Switch-case making product
    final public static int AUTO = 1;
    final public static int MINIMAL = 2;
    final public static int ALL = 3;
    //Swith-case menu options
    final public static int EXIT = 0;
    final public static int CHANGE_NAME = 1;
    final public static int ADD_PRODUCT = 2;
    final public static int SHOW_ALL = 3;
    final public static int SHOW_BY_NAME = 4;
    final public static int SHOW_BY_NAME_AND_PRICE = 5;
    final public static int SHOW_EXPIRED = 6;
    final public static int BACK = 7;
    //Default values for Product class
    final public static String def_name = "NoName";
    final public static String def_manufacturer = "demos";
    final public static int def_price = 100;
    final public static LocalDate def_shelf_time = LocalDate.of(2, 1, 1);
}
```

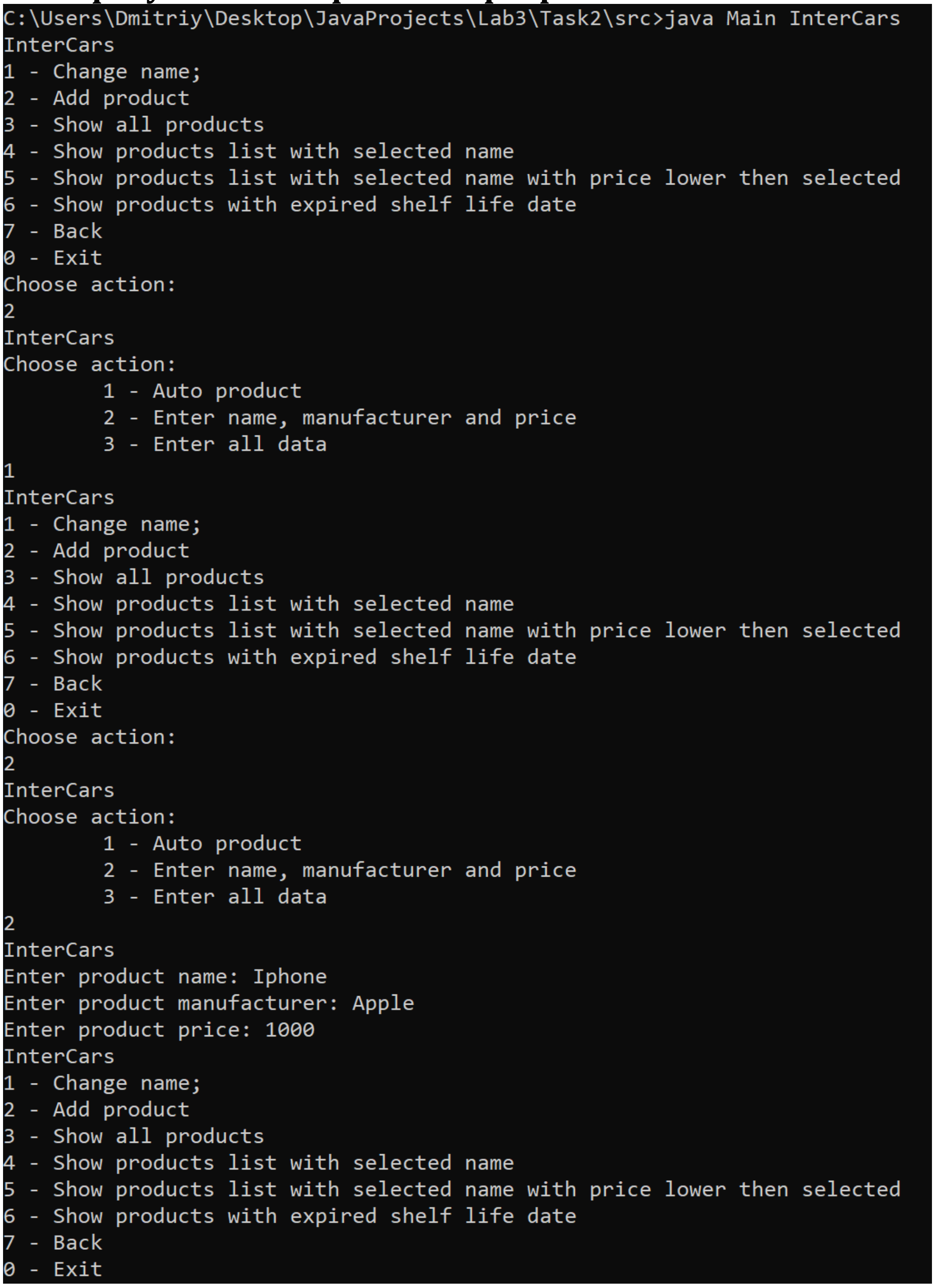
## Спецификация ввода

>java Main “(Название склада)”

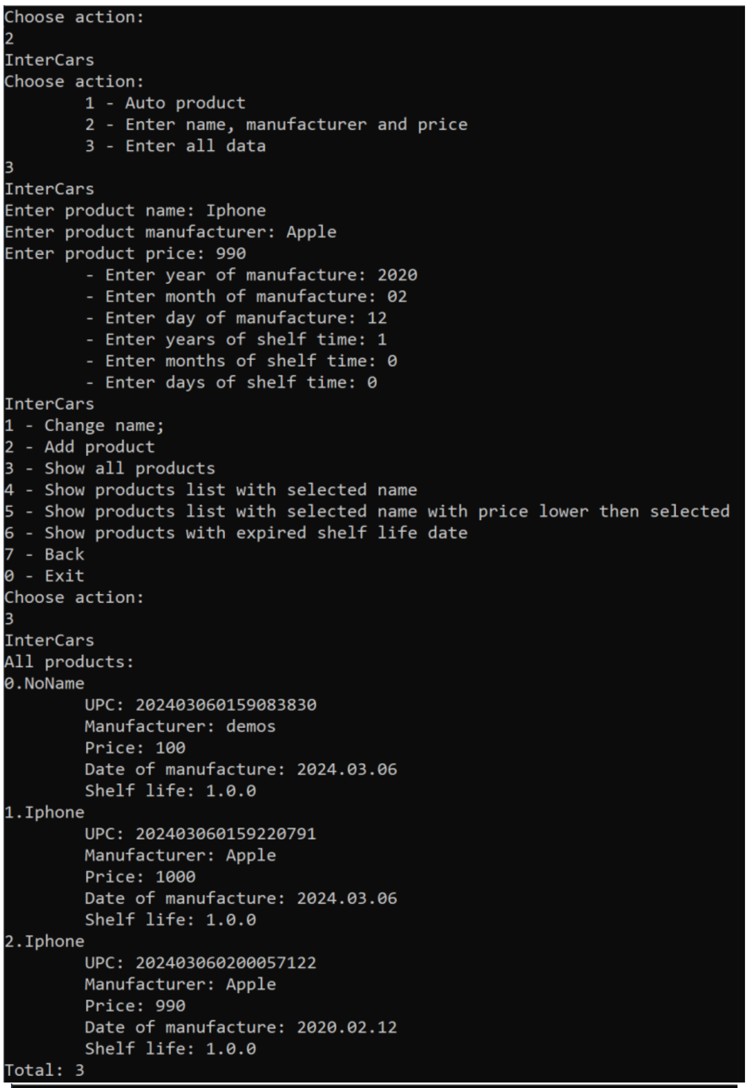
**Пример**

>java Main “InterCars”

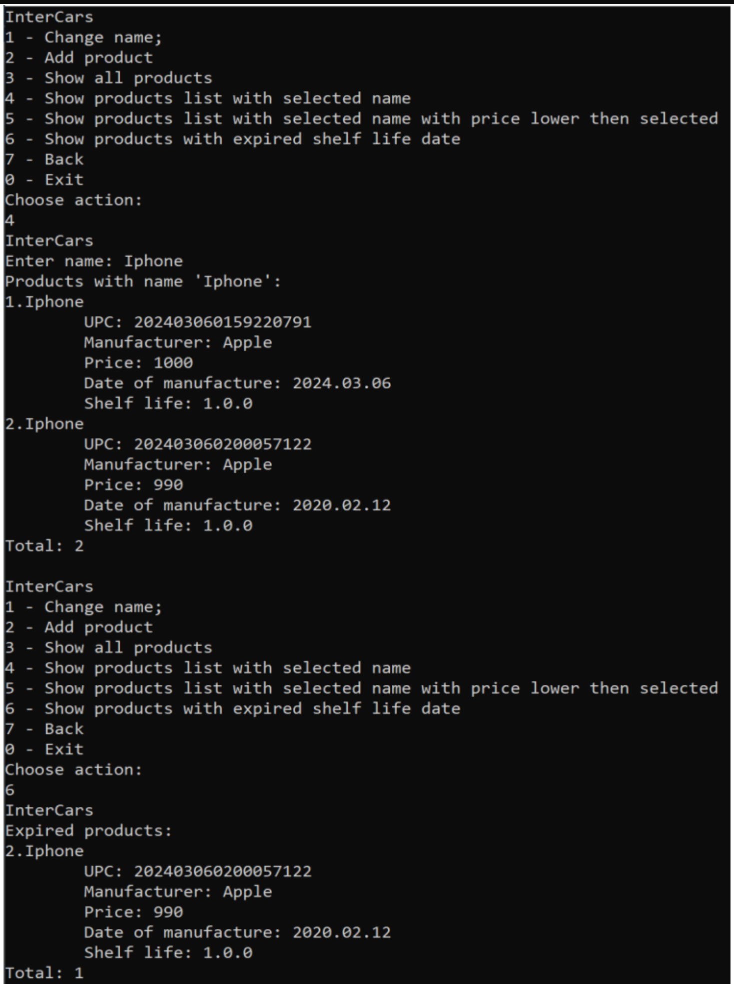
**Рисунки с результатами работы программы**



```
C:\Users\Dmitriy\Desktop\JavaProjects\Lab3\Task2\src>java Main InterCars
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
2
InterCars
Choose action:
        1 - Auto product
        2 - Enter name, manufacturer and price
        3 - Enter all data
1
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
2
InterCars
Choose action:
        1 - Auto product
        2 - Enter name, manufacturer and price
        3 - Enter all data
2
InterCars
Enter product name: Iphone
Enter product manufacturer: Apple
Enter product price: 1000
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
```

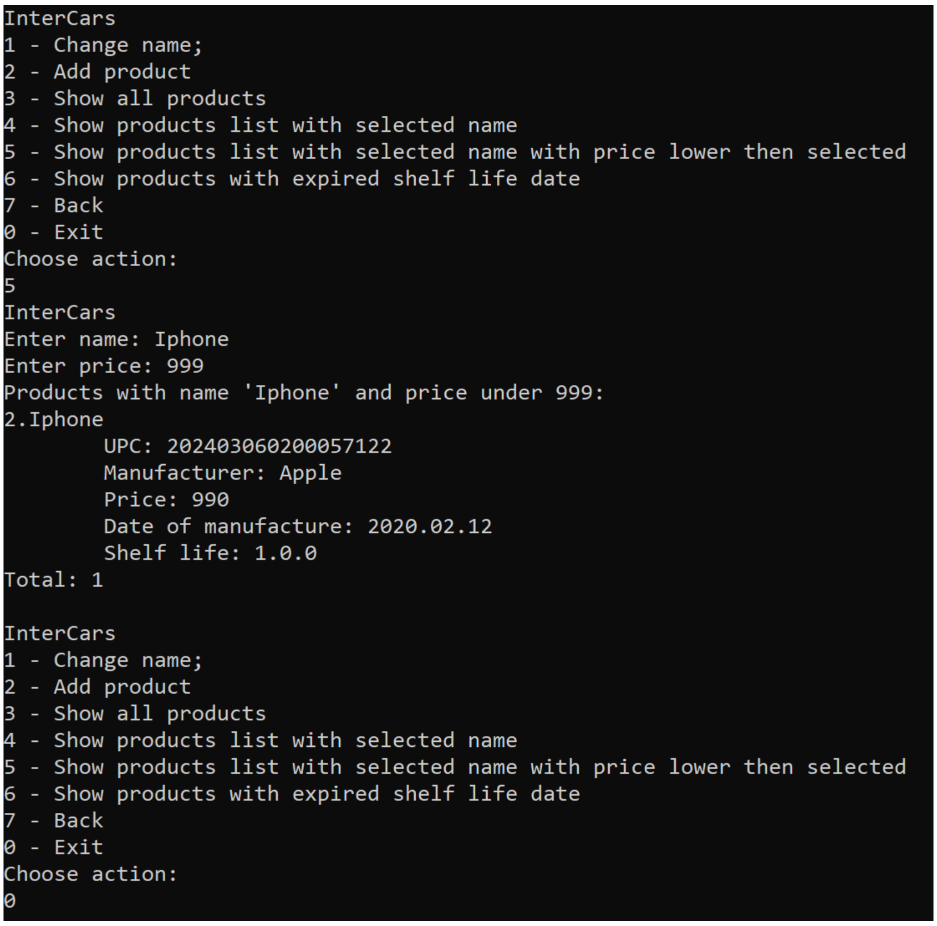


```
Choose action:
2
InterCars
Choose action:
        1 - Auto product
        2 - Enter name, manufacturer and price
        3 - Enter all data
3
InterCars
Enter product name: Iphone
Enter product manufacturer: Apple
Enter product price: 990
        - Enter year of manufacture: 2020
        - Enter month of manufacture: 02
        - Enter day of manufacture: 12
        - Enter years of shelf time: 1
        - Enter months of shelf time: 0
        - Enter days of shelf time: 0
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
3
InterCars
All products:
0.NoName
        UPC: 202403060159083830
        Manufacturer: demos
        Price: 100
        Date of manufacture: 2024.03.06
        Shelf life: 1.0.0
1.Iphone
        UPC: 202403060159220791
        Manufacturer: Apple
        Price: 1000
        Date of manufacture: 2024.03.06
        Shelf life: 1.0.0
2.Iphone
        UPC: 202403060200057122
        Manufacturer: Apple
        Price: 990
        Date of manufacture: 2020.02.12
        Shelf life: 1.0.0
Total: 3
```



```
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
4
InterCars
Enter name: Iphone
Products with name 'Iphone':
1.Iphone
        UPC: 202403060159220791
        Manufacturer: Apple
        Price: 1000
        Date of manufacture: 2024.03.06
        Shelf life: 1.0.0
2.Iphone
        UPC: 202403060200057122
        Manufacturer: Apple
        Price: 990
        Date of manufacture: 2020.02.12
        Shelf life: 1.0.0
Total: 2

InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
6
InterCars
Expired products:
2.Iphone
        UPC: 202403060200057122
        Manufacturer: Apple
        Price: 990
        Date of manufacture: 2020.02.12
        Shelf life: 1.0.0
Total: 1
```



```
InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
5
InterCars
Enter name: Iphone
Enter price: 999
Products with name 'Iphone' and price under 999:
2.Iphone
        UPC: 202403060200057122
        Manufacturer: Apple
        Price: 990
        Date of manufacture: 2020.02.12
        Shelf life: 1.0.0
Total: 1

InterCars
1 - Change name;
2 - Add product
3 - Show all products
4 - Show products list with selected name
5 - Show products list with selected name with price lower then selected
6 - Show products with expired shelf life date
7 - Back
0 - Exit
Choose action:
0
```

**Вывод:** научились создавать и использовать классы в программах на языке программирования Java.