|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 3 |
| **Вариант №** 8 |  |

**Название:**

Классы, наследование и полиморфизм

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ6-23М | |  |  | Н.М. Иванюк |
|  | | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  | |  |  |  |  |
| Преподаватель | |  |  |  | П.В. Степанов |
|  | |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2022

## Цель работы

Получение навыков работы с классами языка программирования Java. Понимание механизмов наследования и полиморфизма объектно-ориентированного программирования.

## Ход работы

*Задание 1.*

8. Определить класс Комплекс. Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения, деления, присваивания комплексных чисел. Создать два вектора размерности n из комплексных координат. Передать их в метод, который выполнит их сложение.

9. Определить класс Квадратное уравнение. Класс должен содержать несколько конструкторов. Реализовать методы для поиска корней, экстремумов, а также интервалов убывания/возрастания. Создать массив объектов и определить наибольшие и наименьшие по значению корни.

Листинг 1 – Код основной программы подзадания 1 первого задания

import java.util.Scanner;

import java.util.Vector;

public class App {

public static void main(String[] args) throws Exception {

System.out.println("Введите n:");

Scanner console = new Scanner(System.in);

int n = console.nextInt();

Vector<Complex> complex\_1 = new Vector<>();

for (int i = 0; i < n; i++) {

System.out.println("Введите re для числа " + (i + 1) + "в векторе 1:");

int re = console.nextInt();

System.out.println("Введите im для числа " + (i + 1) + "в векторе 1:");

int im = console.nextInt();

complex\_1.add(new Complex(re, im));

}

Vector<Complex> complex\_2 = new Vector<>();

for (int i = 0; i < n; i++) {

System.out.println("Введите re для числа " + (i + 1) + "в векторе 2:");

int re = console.nextInt();

System.out.println("Введите im для числа " + (i + 1) + "в векторе 2:");

int im = console.nextInt();

complex\_2.add(new Complex(re, im));

}

PrintVector(complex\_1);

PrintVector(complex\_2);

Vector<Complex> result = SumVectors(complex\_1, complex\_2);

PrintVector(result);

console.close();

}

private static void PrintVector(Vector<Complex> vector) {

System.out.print("[");

for (int i = 0; i < vector.size(); i++) {

System.out.print(vector.get(i).toString());

if (i != vector.size() - 1) System.out.print(", ");

}

System.out.print("]\n");

}

private static Vector<Complex> SumVectors(Vector<Complex> a, Vector<Complex> b) {

Vector<Complex> result = new Vector<>();

for (int i = 0; i < a.size(); i++) {

result.add(a.get(i).Add(b.get(i)));

}

return result;

}

}

Листинг 2 – Код класса Complex подзадания 1 первого задания

public class Complex {

int re, im;

public Complex() {

this.re = 0;

this.im = 0;

}

public Complex(int re) {

this.re = re;

this.im = 0;

}

public Complex(int re, int im) {

this.re = re;

this.im = im;

}

public Complex(Complex complex) {

this.re = complex.re;

this.im = complex.im;

}

// Сложение

public Complex Add(Complex addee) {

// Создаем новый объект класса Complex

Complex sum = new Complex(

this.re + addee.re,

this.im + addee.im

);

return sum;

}

// Вычитание

public Complex Subtract(Complex subtractee) {

Complex sub = new Complex(

this.re - subtractee.re,

this.im - subtractee.im

);

return sub;

}

// Умножение

public Complex Multiply(Complex multee) {

int k1 = multee.re \* (this.re + this.im);

int k2 = this.re \* (multee.im - multee.re);

int k3 = this.im \* (multee.re + multee.im);

int real\_part = k1 - k3;

int imaginary\_part = k1 + k2;

Complex mul = new Complex(real\_part, imaginary\_part);

return mul;

}

// Деление

public Complex Divide(Complex dividee) {

int re\_num = (this.re \* dividee.re) + (this.im \* dividee.im);

int re\_den = (dividee.re \* dividee.re + dividee.im \* dividee.im);

int im\_num = (this.im \* dividee.re) - (this.re \* dividee.im);

int im\_den = (dividee.re \* dividee.re + dividee.im \* dividee.im);

Complex div = new Complex(

re\_num / re\_den,

im\_num / im\_den

);

return div;

}

// Присвоение

public void SetComplex(int re, int im) {

this.re = re;

this.im = im;

}

// Вывод числа

public String toString() {

String complex = re + " + i" + im;

return complex;

}

}

Приведем результаты выполнения данного кода.

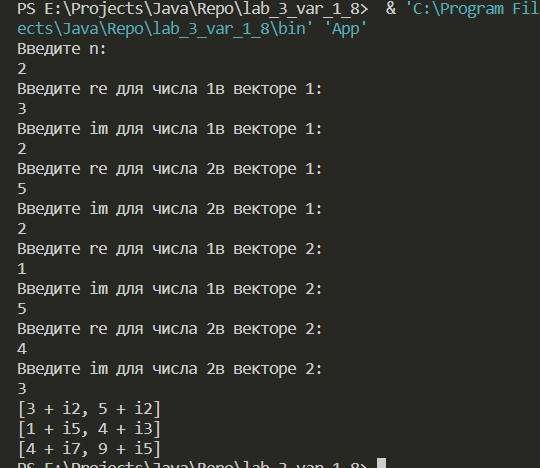


Рисунок 1 – Результат выполнения подзадачи 1 варианта задания 1

Вторая подзадача.

Листинг 3 – Код основной программы подзадачи 2 задания 1

import java.util.ArrayList;

import java.util.Collections;

public class App {

public static void main(String[] args) throws Exception {

SquareEquation one = new SquareEquation(1, 5, 1);

SquareEquation two = new SquareEquation(5, 2, 3);

SquareEquation three = new SquareEquation(6, -17, 12);

ArrayList<Double> roots = new ArrayList<>();

roots.addAll(one.FindRoots());

roots.addAll(two.FindRoots());

roots.addAll(three.FindRoots());

System.out.println(Collections.max(roots));

System.out.println(Collections.min(roots));

}

}

Листинг 4 – Код класса SquareEquation второй подзадачи задания 1

import java.util.ArrayList;

public class SquareEquation {

// ax^2+bx+c=0

int a, b, c;

public SquareEquation() {

a = 1;

b = 1;

c = 0;

}

public SquareEquation(int a, int b, int c) {

this.a = a;

this.b = b;

this.c = c;

}

public ArrayList<Double> FindRoots() {

double d = b \* b - 4.0 \* a \* c;

ArrayList<Double> result = new ArrayList<>();

if (d> 0.0)

{

double r1 = (-b + Math.pow(d, 0.5)) / (2.0 \* a);

double r2 = (-b - Math.pow(d, 0.5)) / (2.0 \* a);

result.add(r1);

result.add(r2);

System.out.println("The roots are " + r1 + " and " + r2);

}

else if (d == 0.0)

{

double r1 = -b / (2.0 \* a);

result.add(r1);

System.out.println("The root is " + r1);

}

else

{

System.out.println("Roots are not real.");

}

return result;

}

public void Intervals() {

double x0 = - b / (2 \* a);

// Print the values

if (a > 0)

{

// Open upward parabola function

System.out.print("Increase = "

+ "(" + x0 + "; +Inf)\n");

System.out.print("Decrease = "

+ "(-Inf; " + x0 + ")\n");

}

else if (a < 0)

{

System.out.print("Increase = "

+ "(-Inf; " + x0 + ")\n");

// Open upward parabola function

System.out.print("Decrease = "

+ "(" + x0 + "; +Inf)\n");

}

else

{

// If a=0 then it is not a quadratic function

System.out.print("Not a quadratic function\n");

}

}

public ArrayList<Double> Extremum() {

double secondPart = c \* 1.0 - (b \* b / (4.0 \* a));

ArrayList<Double> result = new ArrayList<>();

// Print the values

if (a > 0)

{

// Open upward parabola function

System.out.print("Maxvalue = "

+ "Infinity\n");

result.add(Double.POSITIVE\_INFINITY);

System.out.print("Minvalue = " + secondPart);

result.add(secondPart);

}

else if (a < 0)

{

// Open downward parabola function

System.out.print("Maxvalue = " + secondPart + "\n");

result.add(secondPart);

System.out.print("Minvalue = "

+ "-Infinity");

result.add(Double.POSITIVE\_INFINITY);

}

else

{

// If a=0 then it is not a quadratic function

System.out.print("Not a quadratic function\n");

}

return result;

}

}

Приведем результаты выполнения задачи.

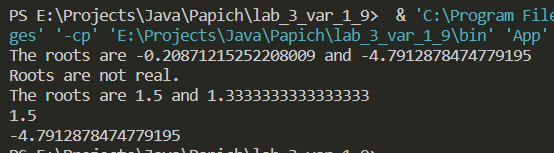


Рисунок 2 – Результат выполнения подзадачи 2 задания 1

*Задание 2.*

Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

8. Car: id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер. Создать массив объектов. Вывести: a) список автомобилей заданной марки; b) список автомобилей заданной модели, которые эксплуатируются больше n лет; c) список автомобилей заданного года выпуска, цена которых больше указанной.

9. Product: id, Наименование, UPC, Производитель, Цена, Срок хранения, Количество. Создать массив объектов. Вывести: a) список товаров для заданного наименования; b) список товаров для заданного наименования, цена которых не превосходит заданную; c) список товаров, срок хранения которых больше заданного.

Листинг 5 – Код выполнения основной программы задания 2

import java.util.Arrays;

public class App {

public static void main(String[] args) throws Exception {

Car cars[] = new Car[4];

// Car 1

cars[0] = new Car();

cars[0].setId(1);

cars[0].setBrand("Volkswagen");

cars[0].setModel("Passat");

cars[0].setYear(2008);

cars[0].setColor("Metallic");

cars[0].setPrice(450000);

cars[0].setNumber("н312ае");

// Car 2

cars[1] = new Car();

cars[1].setId(2);

cars[1].setBrand("Volkswagen");

cars[1].setModel("Polo");

cars[1].setYear(2008);

cars[1].setColor("Black");

cars[1].setPrice(350000);

cars[1].setNumber("н154бг");

// Car 3

cars[2] = new Car();

cars[2].setId(3);

cars[2].setBrand("Mitsubishi");

cars[2].setModel("Lancer");

cars[2].setYear(2010);

cars[2].setColor("Red");

cars[2].setPrice(1350000);

cars[2].setNumber("а102пе");

// Car 4

cars[3] = new Car();

cars[3].setId(4);

cars[3].setBrand("Hyundai");

cars[3].setModel("Accent");

cars[3].setYear(2010);

cars[3].setColor("Red");

cars[3].setPrice(400000);

cars[3].setNumber("н252см");

System.out.println("Автомобили по марке:");

PickBrand(cars, "Volkswagen");

System.out.println("=================================");

System.out.println("Автомобили по модели и эксплуатации:");

PickModel(cars, "Accent", 2022, 10);

System.out.println("=================================");

System.out.println("По году выпуска с превышением заданной цены:");

PickYearPrice(cars, 2010, 1000000);

}

private static void PickBrand(Car[] cars, String brand) {

Arrays.stream(cars)

.filter(c -> c.getBrand() == brand)

.forEach(c -> System.out.println(c.toString()));

}

private static void PickModel(Car[] cars, String model, int currentYear, int n) {

Arrays.stream(cars)

.filter(c -> c.getModel() == model)

.filter(c -> currentYear - c.getYear() > n)

.forEach(c -> System.out.println(c.toString()));

}

private static void PickYearPrice(Car[] cars, int year, int priceLimit) {

Arrays.stream(cars)

.filter(c -> c.getYear() == year)

.filter(c -> c.getPrice() > priceLimit)

.forEach(c -> System.out.println(c.toString()));

}

}

Листинг 6 – Код класса Car

public class Car {

int id;

String brand;

String model;

int year;

String color;

int price;

String number;

public void setId(int id) {

this.id = id;

}

public void setBrand(String brand) {

this.brand = brand;

}

public void setModel(String model) {

this.model = model;

}

public void setColor(String color) {

this.color = color;

}

public void setPrice(int price) {

this.price = price;

}

public void setNumber(String number) {

this.number = number;

}

public void setYear(int year) {

this.year = year;

}

public int getId() {

return id;

}

public String getBrand() {

return brand;

}

public String getModel() {

return model;

}

public String getColor() {

return color;

}

public String getNumber() {

return number;

}

public int getPrice() {

return price;

}

public int getYear() {

return year;

}

@Override

public String toString() {

String result = "{ " +

"ID: " + this.id + "\n" +

"Brand: " + this.brand + "\n" +

"Model: " + this.model + "\n" +

"Year: " + this.year + "\n" +

"Color: " + this.color + "\n" +

"Price: " + this.price + "\n" +

"Number: " + this.number + "\n" +

"}";

return result;

}

}

Результаты выполнения задания приведем на рисунке далее.

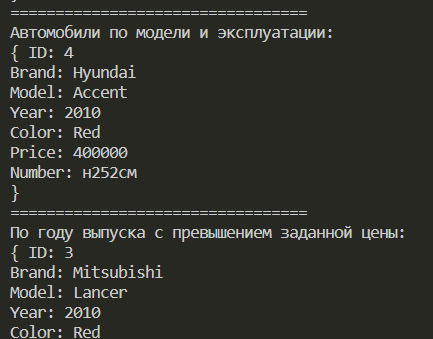


Рисунок 3 – Выполненный код п. 1 задания 2

Второй подпункт.

Листинг 7 – Код выполнения второго пункта задания 2

import java.util.Arrays;

public class App {

public static void main(String[] args) throws Exception {

Product products[] = new Product[4];

products[0] = new Product(

1,

"Milk",

324678,

"Domik v Derevne",

100,

14,

5

);

products[1] = new Product(

2,

"Milk",

324688,

"Prostokvashino",

120,

7,

15

);

products[2] = new Product(

3,

"Milk",

324698,

"Bryansky Molochny Kombinat",

90,

14,

10

);

products[3] = new Product(

4,

"Cereal",

122333,

"Ovsyanka, Sir",

150,

90,

10

);

System.out.println("По наименованию:");

PickTitle(products, "Milk");

System.out.println("-------------");

System.out.println("По наименованию и не превышающие цену:");

PickCost(products, "Milk", 110);

System.out.println("-------------");

System.out.println("По предельному сроку хранения:");

PickValid(products, 14);

}

private static void PickTitle(Product[] products, String title) {

Arrays.stream(products)

.filter(c -> c.getTitle() == title)

.forEach(c -> System.out.println(c.toString()));

}

private static void PickCost(Product[] products, String title, int cost) {

Arrays.stream(products)

.filter(c -> c.getTitle() == title)

.filter(c -> c.getCost() < cost)

.forEach(c -> System.out.println(c.toString()));

}

private static void PickValid(Product[] products, int valid) {

Arrays.stream(products)

.filter(c -> c.getValid\_until() > valid)

.forEach(c -> System.out.println(c.toString()));

}

}

Листинг 8 – Код класса Product

public class Product {

int id;

String title;

int UPC;

String Manufacturer;

int cost;

int valid\_until;

int amount;

public Product(int id, String title, int UPC, String Manufacturer, int cost, int valid\_until, int amount) {

this.id = id;

this.title = title;

this.UPC = UPC;

this.Manufacturer = Manufacturer;

this.cost = cost;

this.valid\_until = valid\_until;

this.amount = amount;

}

public int getId() {

return id;

}

public String getTitle() {

return title;

}

public int getUPC() {

return UPC;

}

public String getManufacturer() {

return Manufacturer;

}

public int getCost() {

return cost;

}

public int getValid\_until() {

return valid\_until;

}

public int getAmount() {

return amount;

}

public void setId(int id) {

this.id = id;

}

public void setTitle(String title) {

this.title = title;

}

public void setUPC(int uPC) {

UPC = uPC;

}

public void setManufacturer(String manufacturer) {

Manufacturer = manufacturer;

}

public void setCost(int cost) {

this.cost = cost;

}

public void setValid\_until(int valid\_until) {

this.valid\_until = valid\_until;

}

public void setAmount(int amount) {

this.amount = amount;

}

@Override

public String toString() {

return "{" +

" id='" + getId() + "'" +

", title='" + getTitle() + "'" +

", UPC='" + getUPC() + "'" +

", Manufacturer='" + getManufacturer() + "'" +

", cost='" + getCost() + "'" +

", valid\_until='" + getValid\_until() + "'" +

", amount='" + getAmount() + "'" +

"}";

}

}

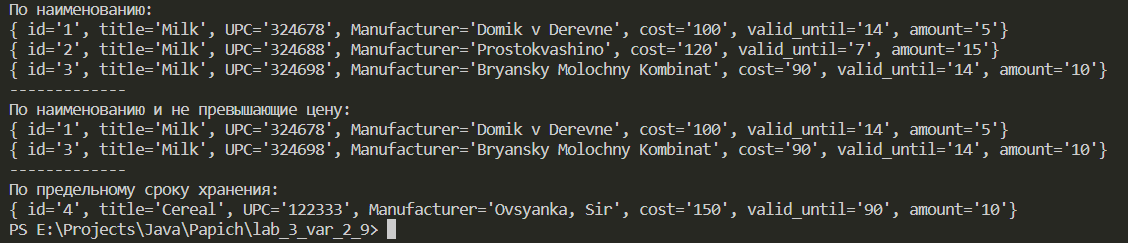


Рисунок 4 – Результаты выполнения задания

*Задание 3.*

Создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

8. Создать объект класса Пианино, используя класс Клавиша. Методы: настроить, играть на пианино, нажимать клавишу.

9. Создать объект класса Фотоальбом, используя класс Фотография. Методы: задать название фотографии, дополнить фотоальбом фотографией, вывести на консоль количество фотографий.

Листинг 9 – Код класса «Piano»

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Objects;

public class Piano {

String model = "Yamaha";

ArrayList<Button> buttons = new ArrayList<>();

public Piano(Button[] buttons) {

this.buttons = new ArrayList<>(Arrays.asList(buttons));

}

public Piano(String title, Button[] buttons) {

this.buttons = new ArrayList<>(Arrays.asList(buttons));

model = title;

}

public void PressButton(int index) {

buttons.get(index).Press();

}

public String getModel() {

return this.model;

}

public void setModel(String model) {

this.model = model;

}

public ArrayList<Button> getButtons() {

return this.buttons;

}

public void setButtons(ArrayList<Button> buttons) {

this.buttons = buttons;

}

@Override

public boolean equals(Object o) {

if (o == this)

return true;

if (!(o instanceof Piano)) {

return false;

}

Piano piano = (Piano) o;

return Objects.equals(model, piano.model) && Objects.equals(buttons, piano.buttons);

}

@Override

public int hashCode() {

return Objects.hash(model, buttons);

}

@Override

public String toString() {

return "{" +

" model='" + getModel() + "'" +

", buttons='" + getButtons() + "'" +

"}";

}

}

Листинг 10 – Код класса «Button»

import java.util.Objects;

public class Button {

String sound = "do";

public Button(String sound) {

this.sound = sound;

}

public void Press() {

System.out.println(sound);

}

public String getSound() {

return this.sound;

}

@Override

public String toString() {

return "{" +

" sound='" + getSound() + "'" +

"}";

}

@Override

public boolean equals(Object o) {

if (o == this)

return true;

if (!(o instanceof Button)) {

return false;

}

Button button = (Button) o;

return Objects.equals(sound, button.sound);

}

@Override

public int hashCode() {

return Objects.hashCode(sound);

}

}

Вторая подзадача.

Листинг 11 – Код класса «Album»

import java.util.ArrayList;

import java.util.Objects;

public class Album {

ArrayList<Photo> photos;

public Album(ArrayList<Photo> photos) {

this.photos = new ArrayList<>(photos);

}

public void SetPhotoTitle(int index, String newTitle) {

photos.get(index).setTitle(newTitle);

}

public void AddPhoto(Photo photo) {

photos.add(photo);

}

public void GetCount() {

System.out.println(photos.size());

}

@Override

public String toString() {

return "{" +

" photos='" + photos + "'" +

"}";

}

@Override

public boolean equals(Object o) {

if (o == this)

return true;

if (!(o instanceof Album)) {

return false;

}

Album album = (Album) o;

return Objects.equals(photos, album.photos);

}

@Override

public int hashCode() {

return Objects.hashCode(photos);

}

}

Листинг 12 – Код класса «Photo»

import java.util.Objects;

public class Photo {

String title;

public Photo(String title) {

this.title = title;

}

public String getTitle() {

return this.title;

}

public void setTitle(String title) {

this.title = title;

}

@Override

public boolean equals(Object o) {

if (o == this)

return true;

if (!(o instanceof Photo)) {

return false;

}

Photo photo = (Photo) o;

return Objects.equals(title, photo.title);

}

@Override

public int hashCode() {

return Objects.hashCode(title);

}

@Override

public String toString() {

return "{" +

" title='" + getTitle() + "'" +

"}";

}

}

*Задание 4.*

Построить модель программной системы.

9. Система Интернет-магазин. Администратор добавляет информацию о Товаре. Клиент делает и оплачивает Заказ на Товары. Администратор регистрирует Продажу и может занести неплательщиков в «черный список».

Листинг 13 – Код класса «Order»

import java.util.ArrayList;

public class Order {

ArrayList<Item> items;

OrderStatus status;

public Order(ArrayList<Item> items) {

this.items = new ArrayList<>(items);

this.status = OrderStatus.CREATED;

}

public void AddToOrder(Item item) {

if (status == OrderStatus.CREATED) {

items.add(item);

}

}

public void FixPayment() {

this.status = OrderStatus.PAYED;

}

public void FixFinish() {

if (status == OrderStatus.PAYED) {

this.status = OrderStatus.CLOSED;

}

}

}

Листинг 14 – Код класса «Client»

import java.util.ArrayList;

public class Client {

String name;

ArrayList<Order> orders = new ArrayList<>();

boolean isBlacklisted;

public Client(String name) {

this.name = name;

this.isBlacklisted = false;

}

public void AddToOrder(int id, Item item) {

orders.get(id).AddToOrder(item);

}

public void PayOrder(int id) {

orders.get(id).FixPayment();

}

public void MoveToBlacklist() {

isBlacklisted = true;

}

}

Листинг 15 – Код класса «Service»

public enum OrderStatus {

CREATED, PAYED, CLOSED

}

Листинг 16 – Код класса «Item»

public class Item {

String title;

int cost;

public String getTitle() {

return this.title;

}

public void setTitle(String title) {

this.title = title;

}

public int getCost() {

return this.cost;

}

public void setCost(int cost) {

this.cost = cost;

}

}

Система Автобаза. Диспетчер распределяет заявки на Рейсы между Водителями и назначает для этого Автомобиль. Водитель может сделать заявку на ремонт. Диспетчер может отстранить Водителя от работы. Водитель делает отметку о выполнении Рейса и состоянии Автомобиля.

Листинг 17 – Класс «Car»

public class Car {

String brand;

public Car(String brand) {

this.brand = brand;

}

}

Листинг 18 – enum «CarState»

public enum CarState {

GOOD, AGED, DAMAGED

}

Листинг 19 – Класс «Driver»

import java.util.Objects;

public class Driver {

String name;

Trip trip;

DriverStatus status = DriverStatus.ON\_HOLD;

public Driver(String name) {

this.name = name;

}

public void SetTrip(Trip trip) {

this.trip = trip;

status = DriverStatus.ON\_TRIP;

}

public void TakeOffTrip() {

this.trip = null;

status = DriverStatus.ON\_HOLD;

}

public void RequestRepair() {

status = DriverStatus.ON\_REPAIR;

}

public void DoRepair() {

if (Objects.isNull(trip))

status = DriverStatus.ON\_HOLD;

else

status = DriverStatus.ON\_TRIP;

}

}

Листинг 20 – enum «DriverStatus»

public enum DriverStatus {

ON\_HOLD, ON\_TRIP, ON\_REPAIR

}

Листинг 21 – Класс «Trip»

public class Trip {

String destination;

Car car;

boolean finished = false;

CarState carState;

public Trip(String destination) {

this.destination = destination;

}

public void SetCar(Car car) {

this.car = car;

}

public void MarkTripFinish(CarState state) {

carState = state;

finished = true;

}

}

В результате выполнения задания были построены структуры двух программных систем. Классы в данных системах взаимодействуют между собой в том числе в рамках взаимоотношений агрегации.

## Вывод

По итогам выполнения лабораторной работы были получены навыки работы с классами ЯП Java. Были также изучены механизмы полиморфизма и наследования.