МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №4

Выполнила:

студентка 3 курса
группы ПО-9
Шубич Дарья
Константинова

Проверил:

Крощенко А.А.

Брест 2024

Цель работы:
Приобрести практические навыки в области объектно-ориентированного проектирования.

Вариант 11

Задание 1

Создать класс Payment (покупка) с внутренним классом, с помощью объектов которого можно сформировать покупку из нескольких товаров.
Входные данные:

```
Payment myPayment = new Payment();
myPayment.addProduct("Apple", 3.24);
myPayment.addProduct("Banana", 6.21);
myPayment.addProduct("Peach", 5.99);

myPayment.showBill();
double totalAmount = myPayment.getTotalAmount();
System.out.println("Total amount: "+ totalAmount);
```

Результат программы

```
/Library/Java/JavaVirtualM
Products in the payment:
Apple   3.24
Banana  6.21
Peach   5.99
Total amount: 15.44
```

Код программы:

```
package org.example;

import java.util.ArrayList;
import java.util.List;

public class Payment {
    private List<Product> products;

    public Payment() {
        products = new ArrayList<>();
    }

    public void addProduct(String name, double price) {
        Product product = new Product(name, price);
        products.add(product);
    }

    public double getTotalAmount(){
        double totalAmount = 0;
```

```
        for (Product product : products) {
            totalAmount += product.getPrice();
        }
        return  totalAmount;
    }

    public void showBill() {
        System.out.println("Products in the payment:");
        for (Product product : products) {
            System.out.println(product.getName() + "\t" +
product.getPrice());
        }
    }

    public class Product {
        private String name;
        private double price;

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

        public String getName() {
            return name;
        }

        public double getPrice() {
            return price;
        }

    }
}
```

Задание 2

Создать класс Звездная система, используя классы Планета, Звезда.

Входные данные:

```
Star star = new Star();
star.setStar("Sun");
Planet[] planets = {new Planet("Earth", true),new Planet("Mars", true), new
Planet("Venus", false)};
StarWay starWay = new StarWay(planets, star);

System.out.println();
starWay.getSystem();
System.out.println();
starWay.planetLive();
System.out.println();
starWay.starMethods();
```

Выходные данные:

```
Sun:
Earth true
Mars true
Venus false

Planet Earth live: true
Status changed
Planet Mars live: true
Status changed
Planet Venus live: false
Status changed


make stars generate energy:
true
Sun generate enargy!
```

Код программы:

```
public class Planet {
    private String planet;
    private boolean isHasLive;
    public String getPlanet() {
        return planet;
    }

    public Planet(String planet, boolean isHasLive) {
        this.planet = planet;
        this.isHasLive = isHasLive;
    }

    public void setPlanet(String planet) {
        this.planet = planet;
    }
    public boolean getIsHaslive() {
        return isHasLive;
    }
    public void setLive(boolean live) {
        this.isHasLive = live;
    }

    public void isPlanetHasLive() {
        System.out.println("Planet " + planet + " live: " + isHasLive);
    }
    public void changeLiveStatus() {
        isHasLive = !isHasLive;
        System.out.println("Status changed");
```

```
    }

}
```

```
public class Star {
    private String star;
    private boolean energy = false;

    public String getStar() {
        return star;
    }

    public void setStar(String star) {
        this.star = star;
    }

    public boolean isEnergy() {
        return energy;
    }

    public void setEnergy(boolean energy) {
        this.energy = energy;
    }

    public void generateEnergy() {
       energy = !energy;
       System.out.println(energy);
    }

    public void isStarGenerateEnargy(){
        if (energy) {
            System.out.println(star + " generate enargy!");
        } else {
            System.out.println(star + " not generate enargy(");
        }
    }

}
```

```
package org.example;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

public class StarWay {

    private Planet[] planets;
    private Star star;

    public StarWay(Planet[] planets, Star star) {
        this.planets = planets;
        this.star = star;
    }

    public void getSystem(){
        System.out.println(star.getStar() + ":");
        for (Planet planet : planets){
```

```
            System.out.println(planet.getPlanet() + " " +
planet.getIsHaslive());
        }
    }

    public void planetLive(){
        for (Planet planet : planets){
            planet.isPlanetHasLive();
            planet.changeLiveStatus();
        }
    }

    public void starMethods(){
        System.out.println();
        System.out.println("make stars generate energy: ");
        star.generateEnergy();
        star.isStarGenerateEnargy();

    }


}
```

Задание 3

Создать класс Звездная система, используя классы Планета, Звезда.

Входные данные:

```
Airport departureAirport = new Airport("Start Airport", "City A");
Airport destinationAirport = new Airport("Destination Airport", "City B");

Aircraft aircraft = new Aircraft(200, 1000);


Flight flight = new Flight("JJJJ123", departureAirport, destinationAirport,
aircraft);


FlightCrew pilot1 = new FlightCrew("Ovodok Vadim", "Pilot");
FlightCrew pilot2 = new FlightCrew("Shubich Darya", "Pilot");
FlightCrew navigator = new FlightCrew("Dmitry Solyshko", "Navigator");
FlightCrew radioOperator = new FlightCrew("Zakhar Kharitanovich", "Radio
Operator");
FlightCrew flightAttendant1 = new FlightCrew("Dmitry Stupak", "Flight
Attendant");
FlightCrew flightAttendant2 = new FlightCrew("Vlad Melnichuk", "Flight
Attendant");


Administrator administrator = new Administrator();


List<FlightCrew> crewMembers = Arrays.asList(pilot1, pilot2, navigator,
radioOperator, flightAttendant1, flightAttendant2);
administrator.formFlightCrew(flight, crewMembers);


Airport newDestination = new Airport("Finish Airport", "City C");
```

```
administrator.changeDestinationAirport(flight, newDestination);


administrator.cancelFlight(flight);


System.out.println("Flight Number: " + flight.getFlightNumber());
System.out.println("Departure Airport: " +
flight.getDepartureAirport().getName());
System.out.println("Destination Airport: " +
flight.setDestinationAirport().getName());

System.out.println("Flight Crew:");
for (FlightCrew crewMember : flight.getFlightCrew()) {
    System.out.println("- " + crewMember.getName() + ", " +
crewMember.getPosition());
}
flight.setCanceled(false);
System.out.println("Flight Status: " + (flight.isCanceled() ? "Canceled" :
"On Schedule"));
```

Выходные данные:

```
Flight Number: JJJJ123
Departure Airport: Start Airport
Destination Airport: Finish Airport
Flight Crew:
- Ovodok Vadim, Pilot
- Shubich Darya, Pilot
- Dmitry Solyshko, Navigator
- Zakhar Kharitanovich, Radio Operator
- Dmitry Stupak, Flight Attendant
- Vlad Melnichuk, Flight Attendant
Flight Status: On Schedule
```

Код программы:

```
package org.example;

import java.util.ArrayList;
import java.util.List;

    class FlightCrew {

        public String getPosition() {
            return position;
```

```
        }

        private String name;
        private String position;

        public FlightCrew(String name, String position) {
            this.name = name;
            this.position = position;
        }

        public String getName() {
            return name;
        }

    }

    class Aircraft {
        private int seatingCapacity;
        private int flightRange;

        public Aircraft(int seatingCapacity, int flightRange) {
            this.seatingCapacity = seatingCapacity;
            this.flightRange = flightRange;
        }
    }

    class Flight {
        private String flightNumber;
        private Airport departureAirport;
        private Airport destinationAirport;
        private boolean isCanceled;
        private Aircraft aircraft;
        private List<FlightCrew> flightCrew;

        public Flight(String flightNumber, Airport departureAirport,
Airport destinationAirport, Aircraft aircraft) {
            this.flightNumber = flightNumber;
            this.departureAirport = departureAirport;
            this.destinationAirport = destinationAirport;
            this.aircraft = aircraft;
            this.isCanceled = false;
            this.flightCrew = new ArrayList<>();
        }

        public void addFlightCrewMember(FlightCrew crewMember) {
            flightCrew.add(crewMember);
        }

        public void setDestinationAirport(Airport newDestination) {
            this.destinationAirport = newDestination;
        }

        public void cancelFlight() {
            this.isCanceled = true;
        }

        public List<FlightCrew> getFlightCrew() {
            return flightCrew;
        }

        public Airport getDepartureAirport() {
```

```
            return departureAirport;
        }

        public String getFlightNumber() {
            return flightNumber;
        }

        public boolean isCanceled() {
            return isCanceled;
        }

        public void setCanceled(boolean canceled) {
            this.isCanceled = canceled;
        }

        public Airport setDestinationAirport() {
            return  destinationAirport;
        }

    }

    class Airport {
        private String name;
        private String location;

        public Airport(String name, String location) {
            this.name = name;
            this.location = location;
        }

        public String getName() {
            return name;

        }
    }

    class Administrator {
        public void formFlightCrew(Flight flight, List<FlightCrew>
crewMembers) {
            for (FlightCrew crewMember : crewMembers) {
                flight.addFlightCrewMember(crewMember);
            }
        }

        public void changeDestinationAirport(Flight flight, Airport
newDestination) {
            flight.setDestinationAirport(newDestination);
        }

        public void cancelFlight(Flight flight) {
            flight.cancelFlight();
        }
    }
```