Министерство образования Республики Беларусь
Учреждение образования
“Брестский государственный технический университет”
Кафедра ИИТ

Отчёт
По лабораторной работе №5
По дисциплине СПП

Выполнил
Студент группы ПО-3
3-го курса
Кулинкович И. Т.

Проверил
Крощенко А. А.

Брест 2020

Лабораторная работа №5

ВАРИАНТ 13

Задание 1. Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов:
interface Транспортное Средство ← abstract class Общественный Транспорт ← class Троллейбус.

Задание 2. В следующих заданиях требуется создать суперкласс (абстрактный класс, интерфейс) и определить общие методы для данного класса. Создать подклассы, в которых добавить специфические свойства и методы. Часть методов переопределить. Создать массив объектов суперкласса и заполнить объектами подклассов. Объекты подклассов идентифицировать конструктором по имени или идентификационному номеру. Использовать объекты подклассов для моделирования реальных ситуаций и объектов. Создать абстрактный класс Работник фирмы и подклассы Менеджер, Аналитик, Программист, Тестировщик, Дизайнер, Бухгалтер. Реализовать логику начисления зарплаты.

Задание 3. В задании 3 ЛР №4, где возможно, заменить объявления суперклассов объявлениями абстрактных классов или интерфейсов.

Код программы

live.ilyusha.spp5.task1.PublicTransport

```
package live.ilyusha.spp5.task1;

abstract class PublicTransport implements Transport {

    private String routeNumber;
    private int seats;
    private String driver;

    public PublicTransport(String routeNumber, int seats, String driver) {
        this.routeNumber = routeNumber;
        this.seats = seats;
        this.driver = driver;
    }

    /* helper methods */

    public String toString() {
        return String.format(
            "<PublicTransport routeNumber=\"%s\" seats=%d driver=\"%s\">",
            routeNumber, seats, driver
        );
    }

    /* abstract methods */

    public abstract void refuel();

    /* codegen */

    public String getRouteNumber() {
        return routeNumber;
    }

    public void setRouteNumber(String routeNumber) {
```

```
        this.routeNumber = routeNumber;
    }

    public int getSeats() {
        return seats;
    }

    public void setSeats(int seats) {
        this.seats = seats;
    }

    @Override
    public String getDriver() {
        return this.driver;
    }

    public void setDriver(String driver) {
        this.driver = driver;
    }

}
```

live.ilyusha.spp5.task1.Transport

```
package live.ilyusha.spp5.task1;

interface Transport {

    /* data */

    String getDriver();

    /* actions */

    void goToLocation(double x, double y);

}
```

live.ilyusha.spp5.task1.Trolleybus

```
package live.ilyusha.spp5.task1;

class Trolleybus extends PublicTransport {

    private double ticketPrice;
    private String brand;

    public Trolleybus(String routeNumber, int seats, String driver, double
ticketPrice, String brand) {
        super(routeNumber, seats, driver);
        this.ticketPrice = ticketPrice;
        this.brand = brand;
    }

    /* helper methods */

    @Override
    public void refuel() {
        System.out.printf("%s is refueling...\n", this);
    }
```

```
    @Override
    public void goToLocation(double x, double y) {
        System.out.printf("%s is driving to location (%f, %f)...\n", this, x,
y);
    }

    @Override
    public String toString() {
        return String.format(
            "<Trolleybus routeNumber=\"%s\" seats=%d driver=\"%s\"
ticketPrice=%f brand=\"%s\">",
            super.getRouteNumber(), super.getSeats(), super.getDriver(),
ticketPrice, brand
        );
    }

    public String toStringCompact() {
        return super.toString();
    }

    /* codegen */

    public double getTicketPrice() {
        return ticketPrice;
    }

    public void setTicketPrice(double ticketPrice) {
        this.ticketPrice = ticketPrice;
    }

    public String getBrand() {
        return brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

}
```

live.ilyusha.spp5.task1.Main

```
package live.ilyusha.spp5.task1;

public class Main {

    public static void main(String[] args) {
        Trolleybus t1 = new Trolleybus("19a", 24, "Mukhov", 5.30, "PAZ");
        Trolleybus t2 = new Trolleybus("24", 24, "Kabachuk", 4.90, "Tesla");

        t1.goToLocation(20.63145, 17.9714);
        System.out.printf("%s abstract class\n", t2.toStringCompact());
        t2.refuel();
    }

}
```

live.ilyusha.spp5.task2.Accountant

```
package live.ilyusha.spp5.task2;

class Accountant extends Worker {

    public Accountant(String name, int age, double salary, double
moneyAccount) {
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Accountant is accounting...");
    }

    @Override
    public void goOnVacation() {
        System.out.println("Accountant is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Analyst

```
package live.ilyusha.spp5.task2;

class Analyst extends Worker {

    public Analyst(String name, int age, double salary, double moneyAccount)
{
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Analyst is analyzing...");
    }

    @Override
    public void goOnVacation() {
        System.out.println("Analyst is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Designer

```
package live.ilyusha.spp5.task2;

class Designer extends Worker {

    public Designer(String name, int age, double salary, double moneyAccount)
{
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Designer is designing...");
```

```
    }

    @Override
    public void goOnVacation() {
        System.out.println("Designer is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Developer

```
package live.ilyusha.spp5.task2;

class Developer extends Worker {

    public Developer(String name, int age, double salary, double
moneyAccount) {
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Developer is developing...");
    }

    @Override
    public void goOnVacation() {
        System.out.println("Developer is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Manager

```
package live.ilyusha.spp5.task2;

class Manager extends Worker {

    public Manager(String name, int age, double salary, double moneyAccount)
{
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Manager is managing...");
    }

    @Override
    public void goOnVacation() {
        System.out.println("Manager is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Tester

```
package live.ilyusha.spp5.task2;

class Tester extends Worker {
```

```
    public Tester(String name, int age, double salary, double moneyAccount) {
        super(name, age, salary, moneyAccount);
    }

    @Override
    public void doWork() {
        System.out.println("Tester is testing...");
    }

    @Override
    public void goOnVacation() {
        System.out.println("Tester is vacating...");
    }

}
```

live.ilyusha.spp5.task2.Worker

```
package live.ilyusha.spp5.task2;

abstract class Worker {

    private String name;
    private int age;
    private double salary;
    private double moneyAccount;

    public Worker(String name, int age, double salary, double moneyAccount) {
        this.name = name;
        this.age = age;
        this.salary = salary;
        this.moneyAccount = moneyAccount;
    }

    /* helper methods */

    public void receiveSalary() {
        moneyAccount += salary;
    }

    public void buyProduct(String product, double price) {
        if (moneyAccount < price) {
            throw new IllegalStateException("Not enough money");
        }
        moneyAccount -= price;
        System.out.printf("%s bought %s for %d\n", name, product, price);
    }

    /* abstract methods */

    public abstract void doWork();
    public abstract void goOnVacation();

    /* codegen */

    public String getName() {
        return name;
    }
```

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

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    public double getMoneyAccount() {
        return moneyAccount;
    }

    public void setMoneyAccount(double moneyAccount) {
        this.moneyAccount = moneyAccount;
    }

}
```

live.ilyusha.spp5.task2.Main

```
package live.ilyusha.spp5.task2;

import java.util.ArrayList;

public class Main {

    public static void main(String[] args) {
        ArrayList<Worker> workers = new ArrayList<>();
        workers.add(new Analyst("John Doe", 30, 3000, 1000));
        workers.add(new Designer("Foo Bar", 30, 3500, 5000));
        workers.add(new Developer("Bar Foo", 30, 2000, 2500));
        workers.add(new Analyst("Bill Gates", 30, 1000, 3000));

        for (Worker i: workers) {
            i.goOnVacation();
            i.doWork();
        }
    }

}
```

live.ilyusha.spp5.task3.Administrator

```
package live.ilyusha.spp5.task3;

class Administrator extends Client {

    public Administrator(String name, CreditCard card) {
```

```
        super(name, card);
    }

    public void blockCard() {
        getCard().setBlocked(true);
    }

    public void unblockCard() {
        getCard().setBlocked(false);
    }

}
```

live.ilyusha.spp5.task3.BankAccount

```
package live.ilyusha.spp5.task3;

abstract class BankAccount {

    abstract void purchase(String product, double price);
    abstract double getCurrencyAmount();

}
```

live.ilyusha.spp5.task3.Client

```
package live.ilyusha.spp5.task3;

public class Client {

    private String name;
    private CreditCard card;

    public Client(String name, CreditCard card) {
        this.name = name;
        this.card = card;
    }

    /* codegen */

    public String getName() {
        return name;
    }

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

    public CreditCard getCard() {
        return card;
    }

    public void setCard(CreditCard card) {
        this.card = card;
    }

}
```

live.ilyusha.spp5.task3.CreditCard

```java
package live.ilyusha.spp5.task3;

class CreditCard extends BankAccount {

    private double currencyAmount;
    private String owner;
    private boolean blocked;

    public CreditCard(double currencyAmount, String owner, boolean blocked) {
        this.currencyAmount = currencyAmount;
        this.owner = owner;
        this.blocked = blocked;
    }

    @Override
    void purchase(String product, double price) {
        if (currencyAmount < price) {
            throw new IllegalStateException("Not enough money");
        }
        if (blocked) {
            throw new IllegalStateException("Card is blocked");
        }
        currencyAmount -= price;
        System.out.printf("%s bought %s for %f\n", owner, product, price);
    }

    /* codegen */

    public String getOwner() {
        return owner;
    }

    public void setOwner(String owner) {
        this.owner = owner;
    }

    @Override
    double getCurrencyAmount() {
        return currencyAmount;
    }

    public void setCurrencyAmount(double currencyAmount) {
        this.currencyAmount = currencyAmount;
    }

    public boolean isBlocked() {
        return blocked;
    }

    public void setBlocked(boolean blocked) {
        this.blocked = blocked;
    }

}
```

live.ilyusha.spp5.task3.Main

```
package live.ilyusha.spp5.task3;

public class Main {

    public static void main(String[] args) {
        String name = "John Doe";
        Administrator a = new Administrator(name, new CreditCard(100, name,
false));
        a.getCard().purchase("Coffee", 20);
        a.blockCard();
        try {
            a.getCard().purchase("Tea", 5);
        } catch (IllegalStateException e) {
            System.out.println(e.toString());
        }
    }

}
```

Спецификация ввода

```
>java Main
```

Пример

```
>java Main
```

Спецификация вывода

Для задачи 1:

```
<результат передвижения в точку>
<данные абстрактного класса>
<результат перезаправки>
```

Для задачи 2:

```
<действие работника 1>
<действие работника 2>
…
<действие работника N>
```

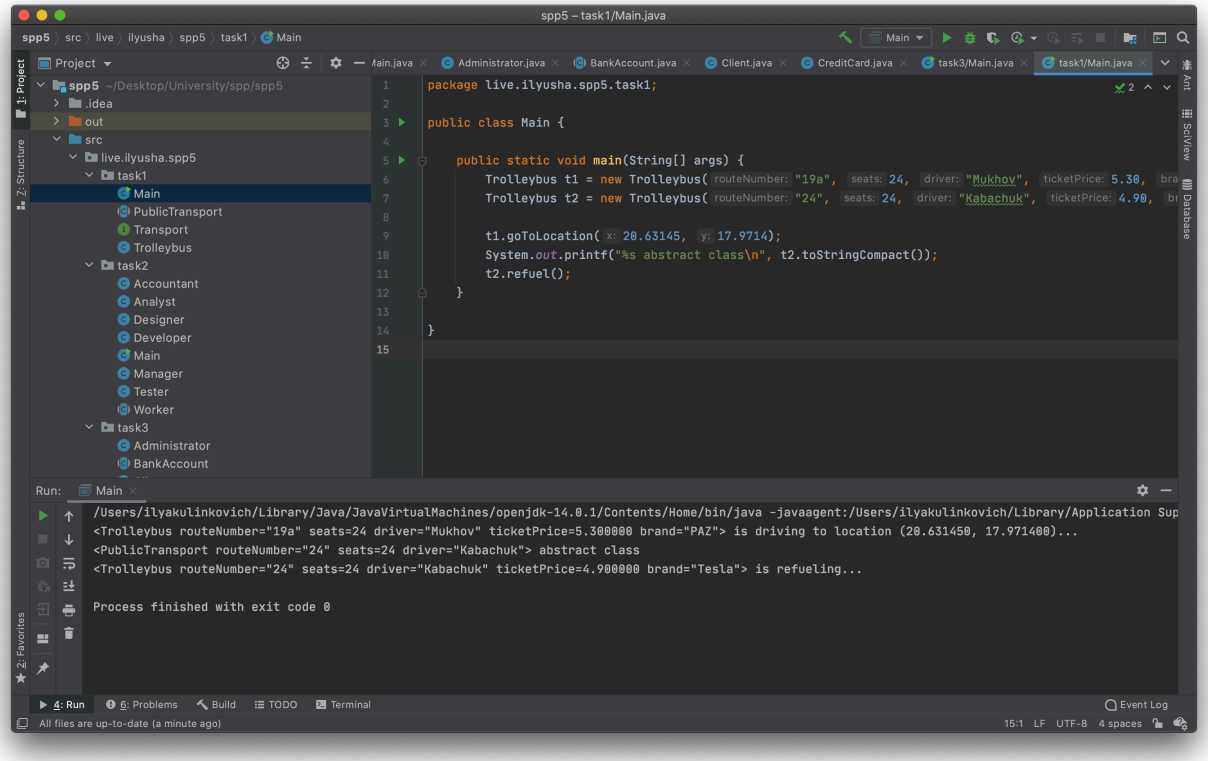
Для задачи 3:

```
<результат операции покупки>
<ошибка вызванная неуспешной покупкой>
```

Пример

```
<Trolleybus routeNumber="19a" seats=24 driver="Mukhov" ticketPrice=5.300000
brand="PAZ"> is driving to location (20.631450, 17.971400)...
<PublicTransport routeNumber="24" seats=24 driver="Kabachuk"> abstract class
<Trolleybus routeNumber="24" seats=24 driver="Kabachuk" ticketPrice=4.900000
brand="Tesla"> is refueling...
```

Рисунки с результатами работы программы



Вывод

В данной лабораторной работе я приобрел практические навыки в области объектно-ориентированного проектирования.