

МАГИСТЕРСКАЯ ПРОГРАММА 09.04.01/07 Интеллектуальные системы анализа,  
обработки и интерпретации больших данных

## Вариант 15

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

Преподаватель	<hr/>	П.В. Степанов
	(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

## Цель работы:

Получение навыков работы с внутренними классами и интерфейсами языка программирования Java.

## Выполнение:

### Задание 1:

1. Создать класс Календарь с внутренним классом, с помощью объектов которого можно хранить информацию о выходных и праздничных днях.
2. Создать класс Shop (магазин) с внутренним классом, с помощью объектов которого можно хранить информацию об отделах, товарах и услуг.

Листинг выполнения подзадачи 1 (файл Calendar.java)

```
package lr41;

import java.util.ArrayList;

public class Calendar {
    private ArrayList<Integer> all_days;

    public Calendar() {
        this.all_days = new ArrayList<>();
        for (int i = 1; i < 366; i++) {
            this.all_days.add(i);
        }
    }

    class Days{
        ArrayList<Integer> holidays;
        ArrayList<Integer> days_off;

        public Days(ArrayList<Integer> holidays, ArrayList<Integer> days_off) {
            this.holidays = holidays;
            this.days_off = days_off;
        }

        public Days() {
            this.days_off = new ArrayList<>();
            this.holidays = new ArrayList<>();
        }

        public void add_holiday(Integer date) {
            this.holidays.add(date);
        }

        public void add_day_off(Integer date) {
            this.days_off.add(date);
        }

        public void clear() {
            this.days_off = new ArrayList<>();
            this.holidays = new ArrayList<>();
        }
    }
}
```

```

        @Override
        public String toString() {
            return "days{" +
                "holidays=" + holidays +
                ", days_off=" + days_off +
                '}';
        }
    }

    public ArrayList<Integer> getAll_days() {
        return all_days;
    }

    public void setAll_days(ArrayList<Integer> all_days) {
        this.all_days = all_days;
    }

    @Override
    public String toString() {
        return "Calendar{" +
            "all_days=" + all_days +
            '}';
    }
}

```

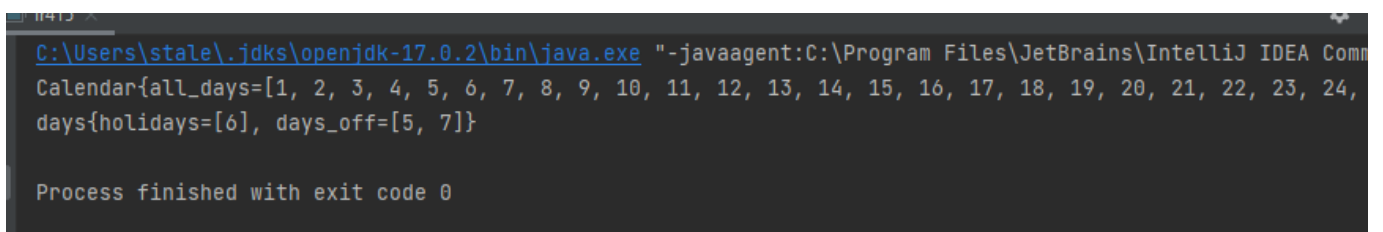
Листинг выполнения подзадачи 1 (файл lr415.java)

```

package lr41;

public class lr415 {
    public static void main(String[] args) {
        Calendar cal = new Calendar();
        Calendar.Days d = cal.new Days();
        d.add_day_off(5);
        d.add_day_off(7);
        d.add_holiday(6);
        System.out.println(cal);
        System.out.println(d);
    }
}

```



```

C:\Users\stale\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Comm
Calendar{all_days=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
days{holidays=[6], days_off=[5, 7]}

Process finished with exit code 0

```

Рисунок 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл Shop.java)

```

package lr41;

import java.util.ArrayList;

public class Shop {
    private String name;

    public Shop(String name) {
        this.name = name;
    }
}

```

```

@Override
public String toString() {
    return "Shop{" +
        "name='" + name + '\'' +
        '}';
}

public String getName() {
    return name;
}

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

class Storage{
    ArrayList<String> departments;
    ArrayList<String> goods;
    ArrayList<String> servivices;

    public Storage() {
        this.departments = new ArrayList<>();
        this.goods = new ArrayList<>();
        this.servivices = new ArrayList<>();
    }

    public void add_department(String d){
        this.departments.add(d);
    }

    public void add_goods(String g){
        this.goods.add(g);
    }

    public void add_service(String s){
        this.servivices.add(s);
    }

    @Override
    public String toString() {
        return "Storage{" +
            "departments=" + departments +
            ", goods=" + goods +
            ", servivices=" + servivices +
            '}';
    }
}
}

```

#### Листинг выполнения подзадачи 2 (файл lr416.java)

```

package lr41;

public class lr416 {
    public static void main(String[] args) {
        Shop shop = new Shop("Dicksy");
        Shop.Storage s = shop.new Storage();
        System.out.println(shop);
        s.add_department("toys");
        s.add_department("long things");
        s.add_goods("balls");
        s.add_goods("Cucumbers");
        s.add_service("Delivery");
        System.out.println(s);
    }
}

```

```
}  
}  
  
C:\Users\stale\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ  
Shop{name='Dicksy'}  
Storage{departments=[toys, long things], goods=[balls, Cucumbers], servivices=[Delivery]}  
  
Process finished with exit code 0
```

Рисунок 2 - Результат выполнения кода решения подзадачи 2

## Задание 2:

Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов:

1. interface Mobile <- abstract class Siemens Mobile <- class Model.
2. interface Корабль <- abstract class Военный Корабль <- class Авианосец.

Листинг выполнения подзадачи 1 (файл Ship.java)

```
package lr42;  
  
public interface Ship {  
    void swim();  
    String getName();  
}
```

Листинг выполнения подзадачи 1 (файл Warship.java)

```
package lr42;  
  
public abstract class Warship implements Ship{  
  
    private String code;  
  
    public Warship(String code) {  
        this.code = code;  
    }  
  
    @Override  
    public void swim() {  
        System.out.println(this.code + " goes brrrrr");  
    }  
  
    @Override  
    public String getName() {  
        return code;  
    }  
}
```

Листинг выполнения подзадачи 1 (файл Carrier.java)

```

package lr42;

public class Carrier extends Warship{
    private int aircraft_amount;
    public Carrier(String code, int aircraft_amount) {
        super(code);
        this.aircraft_amount = aircraft_amount;
    }

    public Carrier(String code) {
        super(code);
        aircraft_amount = 0;
    }

    public int getAircraft_amount() {
        return aircraft_amount;
    }

    public void setAircraft_amount(int aircraft_amount) {
        this.aircraft_amount = aircraft_amount;
    }

    public void sendAircraft(){
        if (aircraft_amount > 0){
            System.out.println("aircraft goes yyyyyyyyy");
            this.aircraft_amount--;
        }else{
            System.out.println("no aircraft to go yyyyyyyyy");
        }
    }

    @Override
    public String getName() {
        return super.getName();
    }

    @Override
    public void swim() {
        super.swim();
    }
}

```

Листинг выполнения подзадачи 1 (файл lr426.java)

```

package lr42;

public class lr425 {
    public static void main(String[] args) {
        Model m = new Model("Mi 11");
        System.out.println(m);
        m.phone();
    }
}

```

```

C:\Users\stale\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent
BJ69 goes brrrr
aircraft goes yyyyyyyyy

Process finished with exit code 0

```

Рисунок 3 - Результат выполнения кода решения подзадачи 1

#### Листинг выполнения подзадачи 2 (файл Mobile.java)

```
package lr42;

public interface Mobile {
    String getFirm();
    void phone();
}
```

#### Листинг выполнения подзадачи 2 (файл siemens\_mobile.java)

```
package lr42;

public abstract class Siemens_mobile implements Mobile{
    private final String firm;

    public Siemens_mobile() {
        this.firm = "Siemens";
    }

    public String getFirm() {
        return firm;
    }

    @Override
    public String toString() {
        return "Siemens_mobile{" +
            "firm='" + firm + '\'' +
            '}';
    }
}
```

#### Листинг выполнения подзадачи 2 (файл Model.java)

```
package lr42;

public class Model extends Siemens_mobile{
    private final String model;

    public Model(String model) {
        super();
        this.model = model;
    }

    @Override
    public String getFirm() {
        return super.getFirm();
    }

    @Override
    public void phone() {
        System.out.println(this.getFirm() + " " + this.model + " says bz-bz");
    }

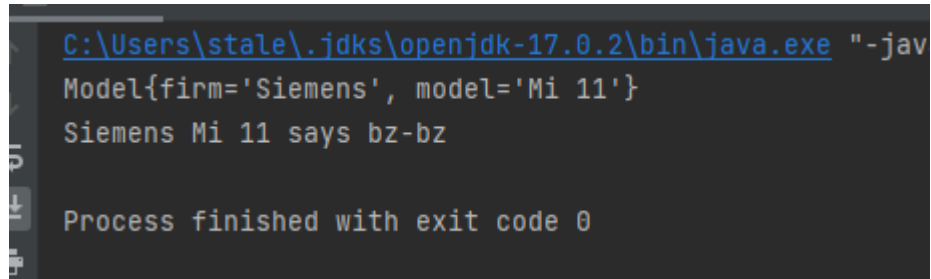
    public String getModel() {
        return model;
    }

    @Override
    public String toString() {
        return "Model{" +
            "firm='" + this.getFirm() + '\'' + ", " +
            "model='" + model + '\'' +
            '}';
    }
}
```

```
}  
}
```

Листинг выполнения подзадачи 2 (файл lr426.java)

```
package lr42;  
  
public class lr425 {  
    public static void main(String[] args) {  
        Model m = new Model("Mi 11");  
        System.out.println(m);  
        m.phone();  
    }  
}
```



```
C:\Users\stale\.jdk\openjdk-17.0.2\bin\java.exe -jav  
Model{firm='Siemens', model='Mi 11'}  
Siemens Mi 11 says bz-bz  
  
Process finished with exit code 0
```

Рисунок 4 - Результат выполнения кода решения подзадачи 2

### **Ссылка на программное решение:**

Программное решение представлено в репозитории распределённой системы управления версиями Git:

[https://github.com/stalekc/java\\_magister/tree/main/lr4/src](https://github.com/stalekc/java_magister/tree/main/lr4/src)

### **Вывод:**

При выполнении лабораторной работы были получены навыки работы с внутренними классами и интерфейсами языка программирования Java.