МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа №2

Основы объектно-ориентированного программирования

Выполнил: Студент группы

БВТ2402

Зорькина Татьяна

Москва

2025

Цель: Изучение и практическая реализация принципов объектно-ориентированного программирования на примере создания иерархии классов на языке Java.

Задания:

Создайте иерархию классов в соответствии с вариантом. Ваша иерархия должна содержать:
• абстрактный класс;
• два уровня наследуемых классов (классы должны содержать в себе минимум 3 поля и 2 метода, описывающих поведение объекта);
• демонстрацию реализации всех принципов ООП;
• наличие конструкторов (в том числе по умолчанию);
• наличие геттеров и сеттеров;
• ввод/вывод информации о создаваемых объектах;
• предусмотрите в одном из классов создание счетчика созданных объектов с использованием статической переменной, продемонстрируйте работу.

Ход работы:

```
public class Main{
    public static void main(String[] args) {
        Dragon dragon1 = new Dragon("Igor", 12, 4.0, 20, "Red", true);
        Dragon dragon2 = new Dragon("Nur", 5, 7.0, 70, "Blue", false);
        Goblin goblin1 = new Goblin(23, 2);
        Mermaid mermaid1 = new Mermaid("Julia", 34, "Green", "Brown");
        Dragon dragon3 = new Dragon();


        dragon2.setName("Kay");
        dragon1.displayInfo();
        goblin1.displayInfo();
        mermaid1.displayInfo();
        dragon3.displayInfo();
        dragon1.Fly();
        dragon3.isAlive();
        goblin1.Run();
        mermaid1.Swim();
        System.out.println("Сила гоблина: " + goblin1.getStrength());
        System.out.println(" Общее количество монстров: "  +
Monster.monsterCount);

    }
}
```

```
abstract class Monster {

    protected String name;
    protected int age;
    static int monsterCount;
    public Monster(){
        this.name = "Неизвестный монстр";
        this.age = 0;
        monsterCount++;
    }

    public Monster(String name, int age){
        this.name = name;
        this.age = age;
        monsterCount++;
    }
```

```
    public void displayInfo() {
        System.out.println("### Информация о монстре: ###");
        System.out.println("Имя: " + name);
        System.out.println("Возраст: " + age);
    }
    public void isAlive(){
        System.out.println("Этот монстр живой");
    }
    public String getName(){
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }
    public int getAge(){
        return this.age;
    }
}
```

```
public class flyingMonster extends Monster{
    int flightSpeed;
    double wingSpan;


    public flyingMonster(){
        super();
        this.flightSpeed = 50;
        this.wingSpan = 5.0;
    }
    public flyingMonster(String name, int age, double wingSpan, int
flightSpeed) {
        super(name, age);
        this.wingSpan = wingSpan;
        this.flightSpeed = flightSpeed;
    }
    public void Fly(){
        System.out.println("Монстр летает");
    }
    @Override
```

```
    public void displayInfo(){
        System.out.println("### Информация о монстре: ###");
        System.out.println("Имя: " + name);
        System.out.println("Возраст: " + age);
        System.out.println("Размах крыла: " + wingSpan);
        System.out.println("Скорость полета: " + flightSpeed);
    }
}
```

```
public class Dragon extends flyingMonster{
    String color;
    boolean canBreatheFire;


    public Dragon(){
        super();
        this.color = "Red";
        this.canBreatheFire = true;
    }

    public Dragon(String name, int age, double wingSpan, int
flightSpeed, String color, boolean canBreatheFire){
        super(name, age, wingSpan, flightSpeed);
        this.color = color;
        this.canBreatheFire = canBreatheFire;
    }
    @Override
    public void Fly(){
        System.out.println("Дракон летает");
    }

    @Override
    public void displayInfo(){
        System.out.println("### Информация о драконе: ###");
        System.out.println("Имя: " + name);
        System.out.println("Возраст: " + age);
        System.out.println("Цвет: " + color);
        System.out.println("Извергает огонь: " + (canBreatheFire ?
"Да" : "Нет"));
    }
    @Override
    public void isAlive(){
        System.out.println("Этот дракон живой");
    }
}
```

```
public class Mermaid extends Monster{
    String tailColor;
    String hairColor;


    public Mermaid(String name, int age, String tailColor, String
hairColor){
        super(name, age);
        this.tailColor = tailColor;
        this.hairColor = hairColor;
    }
```

```
    public void Swim(){
        System.out.println("Русалка плавает");
    }
    @Override
    public void isAlive(){
        System.out.println("Эта русалка живая");
    }
    @Override
    public void displayInfo(){
        System.out.println("### Информация о русалке: ###");
        System.out.println("Имя: " + name);
        System.out.println("Возраст: " + age);
        System.out.println("Цвет хвоста: " + tailColor);
        System.out.println("Цвет волос: " + hairColor);
    }
}
```

```
public class Goblin extends Monster{
    private final int strength;
    int speed;


    public Goblin(int strength, int speed){
        this.strength = strength;
        this.speed = speed;
    }
    public int getStrength(){
        return this.strength;
    }

    @Override
    public void displayInfo(){
        System.out.println("### Информация о гоблине: ###");
        System.out.println("Сила: " + strength);
        System.out.println("Скорость: " + speed);
    }
    public void Run(){
        System.out.println("Гоблин бежит");
    }
    @Override
    public void isAlive(){
        System.out.println("Этот гоблин живой");
    }
}
```

```
### Информация о драконе: ###
Имя: Igor
Возраст: 12
Цвет: Red
Извергает огонь: Да
### Информация о гоблине: ###
Сила: 23
Скорость: 2
### Информация о русалке: ###
Имя: Julia
Возраст: 34
Цвет хвоста: Green
Цвет волос: Brown
### Информация о драконе: ###
Имя: Неизвестный монстр
Возраст: 0
Цвет: Red
Извергает огонь: Да
Дракон летает
Этот дракон живой
Гоблин бежит
Русалка плавает
Сила гоблина: 23
Общее количество монстров: 5
```

Вывод: В результате выполнения лабораторной работы были изучены и реализованы принципы ООП на примере создания иерархии классов в языке Java.

https://github.com/Kidomone/ITIP