Министерство образования и науки РФ

ФГАОУ ВО «Уральский федеральный университет

имени первого Президента России Б. Н. Ельцина»

ИРИТ-РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ N** **3**

по дисциплине «Объектно-ориентированное программирование»

**Тема:** Введение в наследование

Студент группы РИЗ-280028у: И.С.Орлов

Преподаватель: Архипов Н.А,

Екатеринбург 2019

Цель работы: получить представление о механизме наследования в языке программирования Java.

Описание задачи: Научиться работать с классами, научиться создавать объекты (экземпляры класса), разобраться с областями видимости и наследованием классов, рвзобрвться с цепочками наследования.

Ход выполнения работы

Исходные коды по заданиям:

Lab3.1

public class lab3\_1 {

    public static void main(String[] args) {

       sClass test1 = new sClass("First test");

       secondClass test2 = new secondClass("Second test");

       secondClass test3 = new secondClass("First test", "Second Test");

       System.out.println(test1.toString());

       System.out.println(test2.toString());

       System.out.println(test3.toString());

    }

}

class sClass {

    private String s1;

    public String getS1() {

        return s1;

    }

    sClass(String s1) {

        this.s1 = s1;

    }

    @Override

    public String toString() {

        return "sClass. Field s1 = " + s1;

    }

}

class secondClass extends sClass {

    private String s2 = "";

    secondClass (String s1, String s2) {

        super(s1);

        this.s2 = s2;

    }

    secondClass (String s1) {

        super(s1);

    }

 public String toString() {

        return "secondClass. field s1 = " + getS1() + " " +s2 ;

    }

}

Lab3.2

public class lab3\_2 {

    public static void main(String[] args) {

       sClass test1 = new sClass("First test");

       secondClass test2 = new secondClass("Second test" , 24);

       System.out.println(test1.toString());

       System.out.println(test2.toString());

       test2.setFields();

       System.out.println(test2.toString());

       test2.setFields("test2");

       System.out.println(test2.toString());

       test2.setFields(3);

       System.out.println(test2.toString());

       test2.setFields("test4", 4);

       System.out.println(test2.toString());

      // System.out.println(test3.toString());

    }

}

class sClass {

    private String \_s1;

    private int \_len;

    public void set\_s1(String \_s1) {

        this.\_s1 = \_s1;

    }

    public int get\_len() {

        \_len = \_s1.length();

        return \_len;

    }

    public String getS1() {

        return \_s1;

    }

    sClass(String \_s1) {

        this.\_s1 = \_s1;

    }

    @Override

    public String toString() {

        return "sClass. Field s1 = " + \_s1 + "  l = " + get\_len();

    }

}

class secondClass extends sClass {

    public int i;

    secondClass (String \_s1, int i) {

        super(\_s1);

        this.i = i;

    }

    void setFields () {

        super.set\_s1("woFields");

        i = 0;

    }

    void setFields (String s) {

        super.set\_s1(s);

    }

    void setFields (int i) {

        this.i = i;

    }

    void setFields (String s, int i) {

        super.set\_s1(s);

        this.i = i;

    }

 public String toString() {

        return "secondClass. field s1 = " + getS1() + " i\_open = " + i ;

    }

}

Lab3.3

public class lab3\_3 {

    public static void main(String[] args) {

       sClass test1 = new sClass(1);

       secondClass test2 = new secondClass(2,'2');

       thirdClass test3 = new thirdClass(3,'3', "Test");

       System.out.println(test1.toString());

       System.out.println(test2.toString());

       System.out.println(test3.toString());

    }

}

class sClass {

    public int f1;

    public void setF(int f1) {

        this.f1 = f1;

    }

    sClass(int f1) {

        this.f1 = f1;

    }

    @Override

    public String toString() {

        return "sClass. Field f1 = " + f1;

    }

}

class secondClass extends sClass {

    public char f2 = ' ';

    secondClass (int f1, char f2) {

        super(f1);

        this.f2 = f2;

    }

    public void setF(int f1, char f2) {

        super.setF(f1);

        this.f2 = f2;

    }

 public String toString() {

        return "secondClass. field f1 = " + super.f1 + " field f2 = " +f2 ;

    }

}

class thirdClass extends secondClass {

    public String f3 = "";

    thirdClass(int f1, char f2, String f3) {

        super(f1, f2);

        this.f3 = f3;

    }

    public void setF(int f1, char f2, String f3) {

        super.setF(f1, f2);

        this.f3 = f3;

    }

 public String toString() {

        return "thirdClass. field f1 = " + super.f1 + " field f2 = " + super.f2 + " field f3 = " + f3 ;

    }

}

Lab3.4

public class lab3\_4 {

    public static void main(String[] args) {

       sClass test1 = new sClass('1');

       secondClass test2 = new secondClass('2',"Second");

       thirdClass test3 = new thirdClass('3',"third", 3);

       thirdClass test4 = new thirdClass(test3);

       System.out.println(test1.toString());

       System.out.println(test2.toString());

       System.out.println(test3.toString());

       System.out.println(test4.toString());

    }

}

class sClass {

    public char f1;

    public void setF(char f1) {

        this.f1 = f1;

    }

    sClass(char f1) {

        this.f1 = f1;

    }

    sClass(sClass in) {

        this(in.f1);

    }

    @Override

    public String toString() {

        return "sClass. Field f1 = " + f1;

    }

}

class secondClass extends sClass {

    public String f2 = "";

    secondClass (char f1, String f2) {

        super(f1);

        this.f2 = f2;

    }

    public secondClass(secondClass in) {

        this(in.f1, in.f2);

    }

    public void setF(char f1, String f2) {

        super.setF(f1);

        this.f2 = f2;

    }

 public String toString() {

        return "secondClass. field f1 = " + super.f1 + " field f2 = " +f2 ;

    }

}

class thirdClass extends secondClass {

    public int f3 = 0;

    thirdClass(char f1, String f2, int f3) {

        super(f1, f2);

        this.f3 = f3;

    }

    public thirdClass(thirdClass in) {

        this(in.f1, in.f2, in.f3);

    }

    public void setF(char f1, String f2, int f3) {

        super.setF(f1, f2);

        this.f3 = f3;

    }

 public String toString() {

        return "thirdClass. field f1 = " + super.f1 + " field f2 = " + super.f2 + " field f3 = " + f3 ;

    }

}

Lab3.5

public class lab3\_5 {

    public static void main(String[] args) {

        sClass test1 = new sClass("first test");

        firstClass test2 = new firstClass("Second test", 2);

        secondClass test3 = new secondClass("test3", 'a');

        sClass test4 = new secondClass("test4", 'b');

        System.out.println(test1.toString());

        System.out.println(test2.toString());

        System.out.println(test3.toString());

        System.out.println(test4.toString());

    }

}

class sClass {

    protected String f1 = "";

    sClass(String f1) {

        this.f1 = f1;

    }

    public String toString() {

        return "sClass " + "f1 " + f1;

    }

}

class firstClass extends sClass {

    protected int f2\_int;

    public firstClass(String f1, int f2\_int) {

        super(f1);

        this.f2\_int = f2\_int;

    }

    public String toString() {

        return "firstClass " + "f1 " + f1 + " f2 " + f2\_int;

    }

}

class secondClass extends sClass {

    protected char f3\_char;

    public secondClass(String f1, char f3\_char) {

        super(f1);

        this.f3\_char = f3\_char;

    }

    public String toString() {

        return "secondClass " + "f1 " + f1 + " f3 " + f3\_char;

    }

}

Вывод

В ходе выполнения лабораторной работы N 3, был приобретен навык работы с наследованием в языке программирования Java, было отмечено что при присвоении переменной суперкласса ссылки класса потомка, обращение по этой переменной идет к объекту класса потомка.