|  |  |
| --- | --- |
|  |  |
| МИНОБРНАУКИ РОССИИ |  |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |  |

**Институт информационных технологий**

КАФЕДРА ИНСТРУМЕТНАЛЬНОГО И ПРИКЛАДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (ИППО)

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине «Объектно-ориентированное программирование»

«НАСЛЕДОВАНИЕ В JAVA»

ВАРИАНТ 2

Выполнил студент группы ИНБО-01-18 *Трофимов С. М.*

Принял Батанов А.О.

Лабораторная работа выполнена «\_\_\_»\_\_\_\_\_\_\_2019г.

«Зачтено» «\_\_\_»\_\_\_\_\_\_\_2019г.

Москва 2019

**Цель работы**

Цель данной лабораторной работы - изучить понятие наследования, и научиться реализовывать наследование в Java.

**Теоретическое введение**

Одним из ключевых аспектов объектно-ориентированного программирования является наследование. С помощью наследования можно расширить функционал уже имеющихся классов за счет добавления нового функционала или изменения старого.

Чтобы объявить один класс наследником от другого, надо использовать после имени класса-наследника ключевое слово extends, после которого идет имя базового класса.

С помощью ключевого слова super мы можем обратиться к любому члену базового класса - методу или полю, если они не определены с модификатором private.

Кроме обычных классов в Java есть абстрактные классы. Абстрактный класс похож на обычный класс. В абстрактном классе также можно определить поля и методы, в то же время нельзя создать объект или экземпляр абстрактного класса. Абстрактные классы призваны предоставлять базовый функционал для классов- 23 наследников. А производные классы уже реализуют этот функционал.

Производный класс обязан переопределить и реализовать все абстрактные методы, которые имеются в базовом абстрактном классе. Также следует учитывать, что если класс имеет хотя бы один абстрактный метод, то данный класс должен быть определен как абстрактный.

**Выполнение практической работы**

*Задание:*

Создать абстрактный класс, описывающий собак(Dog). С помощью наследования реализовать различные породы собак. Протестировать работу классов.

*Решение:*

Файл Dog.java

**package** com.gradess.company.dogs;  
  
**public abstract class** Dog {  
  
 **protected** String **name**;  
 **protected** String **breed**;  
  
 **protected int age**;  
  
 **private float weight**;  
  
 **protected float height**;  
  
 **protected boolean isMale**;  
  
 **public** Dog(String name, **int** age, **float** weight, **float** height, **boolean** isMale) {  
 **this**.**name** = name;  
 **this**.**age** = age;  
 **this**.**weight** = weight;  
 **this**.**height** = height;  
 **this**.**isMale** = isMale;  
 }  
  
 **public abstract void** bark();  
 **public abstract void** jump();  
 **public abstract void** run();  
  
 @Override  
 **public** String toString() {  
 **return "Собака "** + **name** +  
 **"\nПорода: "** + **breed** +  
 **"\nВозраст: "** + **age** +  
 **"\nВес: "** + **weight** +  
 **"\nРост: "** + **height** +  
 **"\nПол: "** + (**isMale** ? **"М"** : **"Ж"**);  
 }  
}

Файл Aussie.java

**package** com.gradess.company.dogs;  
  
**public class** Aussie **extends** Dog {  
  
 **public** Aussie(String name, **int** age, **float** weight, **float** height, **boolean** isMale) {  
 **super**(name, age, weight, height, isMale);  
 **breed** = **"Аусси"**;  
 }  
  
 @Override  
 **public void** bark() {  
 System.***out***.println(**name** + **" громко лает"**);  
 }  
  
 @Override  
 **public void** jump() {  
 System.***out***.println(**name** + **" прыгает"**);  
 }  
  
 @Override  
 **public void** run() {  
 System.***out***.println(**name** + **" медленно бегает"**);  
 }  
}

Файл BassetHound.java

**package** com.gradess.company.dogs;  
  
**public class** BassetHound **extends** Dog {  
  
 **public** BassetHound(String name, **int** age, **float** weight, **float** height, **boolean** isMale) {  
 **super**(name, age, weight, height, isMale);  
 **breed** = **"Бассет-Хаунд"**;  
 }  
  
 @Override  
 **public void** bark() {  
 System.***out***.println(**name** + **" тихо лает"**);  
 }  
  
 @Override  
 **public void** jump() {  
 System.***out***.println(**name** + **" высокоо прыгает"**);  
 }  
  
 @Override  
 **public void** run() {  
 System.***out***.println(**name** + **" бегает"**);  
 }  
}

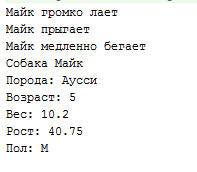
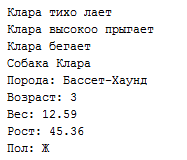
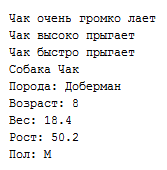
Файл Doberman.java

**package** com.gradess.company.dogs;  
  
**public class** Doberman **extends** Dog {  
  
 **public** Doberman(String name, **int** age, **float** weight, **float** height, **boolean** isMale) {  
 **super**(name, age, weight, height, isMale);  
 **breed** = **"Доберман"**;  
 }  
  
 @Override  
 **public void** bark() {  
 System.***out***.println(**name** + **" очень громко лает"**);  
 }  
  
 @Override  
 **public void** jump() {  
 System.***out***.println(**name** + **" высоко прыгает"**);  
 }  
  
 @Override  
 **public void** run() {  
 System.***out***.println(**name** + **" быстро прыгает"**);  
 }  
}

Файл Main.java

**import** com.gradess.company.dogs.Aussie;  
**import** com.gradess.company.dogs.BassetHound;  
**import** com.gradess.company.dogs.Doberman;  
**import** com.gradess.company.dogs.Dog;  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
  
 Dog aussie = **new** Aussie(**"Майк"**, 5, 10.2f, 40.75f, **true**);  
  
 Dog bassetHound = **new** BassetHound(**"Клара"**, 3, 12.59f, 45.36f, **false**);  
  
 Dog doberman = **new** Doberman(**"Чак"**, 8, 18.4f, 50.2f, **true**);  
  
 *testDog*(aussie);  
 *testDog*(bassetHound);  
 *testDog*(doberman);  
  
 }  
 **private static void** testDog(Dog dog) {  
 dog.bark();  
 dog.jump();  
 dog.run();  
  
 System.***out***.println(dog + **"\n"**);  
 }  
}

**Результат выполнения программы**

**Выводы по работе**

Я изучил понятие наследования, и научился реализовывать наследование в Java.

**Используемая литература:**

1. К. Сьерра, Б. Бэйтс. Изучаем Java, 2-е издание.
2. С. Макконнелл. Совершенный код, 3-е издание.