**Лабораторная работа №2.**

Тема: Объективно-ориентированное программирование и диаграмма классов

Цель работы: освоить навыки использования методов ООП: инкапсуляция, полиформизм, наследование

1. **Индивидуальные задания**

1. Напишите программу с классом Mother от которого наследуется класс Daughter.

Сделайте так, чтобы результат print(object) был разный.

Воспользуйтесь принципами полиморфизма, наследования и инкапсуляции.

Main;

public class Main {  
 public static void main(String[] args) {  
 Mother mother = new Mother("Анна");  
 Daughter daughter = new Daughter("Катя", "Танцы");  
 System.*out*.println(mother);  
 System.*out*.println(daughter);  
  
 }  
}

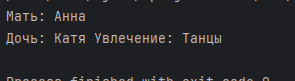
Class mother

public class Mother {  
 private String name;  
  
 public Mother(String name) {  
 this.name = name;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 @Override  
 public String toString() {  
 return "Мать: " + name;  
 }  
}

class daughter

public class Daughter extends Mother{  
 private String hobby;  
 public Daughter (String name, String hobby){  
 super(name);  
 this.hobby=hobby;  
 }  
 @Override  
 public String toString(){  
 return "Дочь: " + getName() + " Увлечение: " + hobby;  
 }  
}

Вывод программы



2.Реализйте класс Animal. Внутри объявите поле для имени и возраста.

От класса Animal унаследуйте класс Zebra и Dolphin.

Оба класса могут вернуть описание, содержащее имя, возраст и какую-то доп.информацию, например, что это за вид животного.

Воспользуйтесь принципами полиморфизма, наследования и инкапсуляции.

Мейн

public class Main {  
 public static void main(String[] args) {  
 Animal Zebra = new Zebra("Майк", 3);  
 Animal Dolphin = new Dolphin("Винни", 5);  
 System.*out*.println(Zebra.getDescription());  
 System.*out*.println(Dolphin.getDescription());  
  
 }  
}

Класс животные

public class Animal {  
 private String name;  
 private int age;  
  
  
 public Animal(String name, int age){  
 this.name=name;  
 this.age=age;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public int getAge() {  
 return age;  
 }  
 public String getDescription(){  
 return "Имя: " + name + ", Возраст: " + age;  
 }  
}

Класс зебра

public class Zebra extends Animal{  
 public Zebra(String name, int age){  
 super(name, age);  
 }  
 @Override  
 public String getDescription(){  
 return super.getDescription() + ", Вид: Зебра";  
 }  
}

Класс дельфин

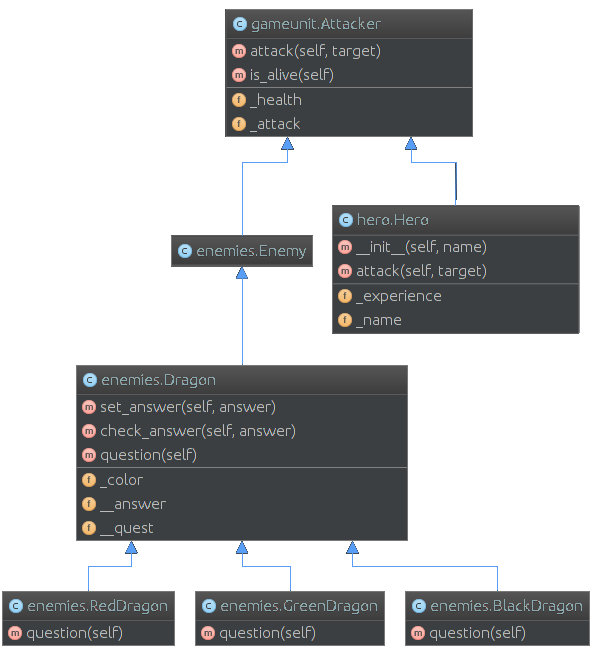
public class Dolphin extends Animal{  
 public Dolphin(String name, int age){  
 super(name, age);  
 }  
 @Override  
 public String getDescription(){  
 return super.getDescription() + ", Вид: Дельфин";  
 }  
}

Вывод программы



3. Игра "Арифметические драконы" предназначена для обучения детей арифметике. На героя нападает дракон, который задаёт вопрос на сложение (если дракон зелёный), вычитание (красный) или умножение (чёрный).

Реализуйте следующие классы

Вывод: освоены навыки использования методов ООП: инкапсуляция, полиформизм, наследование