Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

Программирование интернет приложений

Лабораторная работа №2

Вариант: 1822

Выполнил: **Шкаруба Н.Е.**

Проверил: **Гаврилов А.В.**

группа: **P3218**

год: **2015**

1. Исходный код с комментариями

```
class Shellos {
    int surface;
    protected String shadowIce = "ShadowIce"; // 1. Explicit initialization
    public int reach;
    float attack = 8.9f; // 1.
    protected String ice = "Ice"; // 1.
    protected byte phasing;
    protected String shadow = "Shadow"; // 1.

    public Shellos() { // 3. Constructor
        reach = 060;
        phasing = (byte) 0x87;
    }

    {   // 2. Instance's initialization section
        surface = 87;
        phasing = 76;
    }


    public void painSplit(Shellos p) {
        System.out.println("Shellos attacks Shellos with Pain Split");
    }
    public void painSplit(Gastrodon p) {
        System.out.println("Shellos attacks Gastrodon with Pain Split");
    }
    public void batonPass() {
        System.out.println("Shellos casts Baton Pass");
    }
    public void torrent() {
        float weight = 3.9f;

        // correct float equality way is if(Math.abs(attack + weight - 12.8f) < epsilon)
        System.out.println((attack + weight) == 12.8f); // (12.8f == 12.8) == false coz  Java's
float representation
    }
    public static void foresight() {
        System.out.println("Shellos casts Foresight");
    }
    public void splash() {
        System.out.println(reach - phasing); // (48 - -121 == 169)
        System.out.println(surface - reach); // (87 - 48 == 39)
        System.out.println(phasing + surface); // (-121 + 87 == -34)
    }
    public void stickyHold() {
        System.out.println(shadowIce == "Shadow"+ice); // false coz links comparison
        System.out.println(shadowIce.equals("Shadow"+"Ice")); // true coz equals compares content
        System.out.println(shadowIce.equals(shadow+"Ice")); // true
        System.out.println(shadowIce.equals(shadow+ice)); // true
        System.out.println(shadowIce == shadow+ice); // false
        System.out.println(shadowIce == "Shadow"+"Ice"); // true coz of compiler's optimization
    }
}
/* String equality explanation
(1) aString == aConcatentatedString : true
    aString.equals(aConcatentatedString) : true
(2) aString == aRuntimeString : false
    aString.equals(aRuntimeString) : true
(3) aString == anInternedString : true
    aString.equals(anInternedString) : true
(4) aString == anExplicitString : false
    aString.equals(anExplicitString) : true
(5) aString == firstArg : false
    aString.equals(firstArg) : true
(6) aString == firstArgInterned : true
    aString.equals(firstArgInterned) : true */
```

```java
class Gastrodon extends Shellos {
    private String ghost = "Ghost";
    double defense = 3.2;
    private static int fragile = 87; // 0. Static initialization: Initializes right before the
first appeal or before a first object creation
    private String ghostIce = "GhostIce";

    public void painSplit(Gastrodon p) {
        System.out.println("Gastrodon attacks Gastrodon with Pain Split");
    }
    public void focusEnergy() {
        System.out.println(phasing - fragile);  // -121 - 87 == -208
        System.out.println(fragile - surface);  // 87 - 87 == 0
        System.out.println(fragile + reach);    // 87 + 48 == 135
    }
    public void camouflage() {
        double length = 4.9;

        System.out.println((length + defense) == 1.7); // false coz ((4.9 + 3.2) != 1.7)
    }
    public static void foresight() {
        System.out.println("Gastrodon casts Foresight");
    }
    public void painSplit(Shellos p) {
        System.out.println("Gastrodon attacks Shellos with Pain Split");
    }
    public void swordsDance() {
        // ghostIce == ghost+ice : ghostIce == new StringBuilder(ghost).append(ice).toString();
        System.out.println(ghostIce == ghost+ice); // false, coz Equals compares links ghost+ice
        System.out.println(ghostIce == new String("GhostIce"));// false coz explicit Constructor
        System.out.println(ghostIce == (ghost+ice).intern());// true coz intern return link with
the same content if possible
        System.out.println(ghostIce == new String("Ghost"+"Ice"));  // false
    }
    public void batonPass() {
        System.out.println("Gastrodon casts Baton Pass");
    }
}


public class Lab2 {
    public static void main(String[] args) {
        Gastrodon daughter = new Gastrodon();
        Shellos mother = new Shellos();
        Shellos sister = new Gastrodon();

        sister.foresight();
        daughter.painSplit(sister);
        ((Gastrodon)sister).swordsDance(); //!
        mother.painSplit(sister);
        mother.batonPass();
        daughter.foresight();
        daughter.focusEnergy(); //!
        daughter.camouflage();
        daughter.batonPass();
        mother.torrent(); //!
        mother.foresight();
        daughter.splash(); //!
        daughter.painSplit(mother);
        sister.batonPass();
        sister.stickyHold();
        mother.painSplit(daughter);
    }
}
```

2. Результат выполнения

```
/* output:
Shellos casts Foresight
Gastrodon attacks Shellos with Pain Split
!SwordsDance: false false true false
Shellos attacks Shellos with Pain Split
Shellos casts Baton Pass
Gastrodon casts Foresight
!FocusEnergy: -208 0 135 false
!Camouflage: false
Gastrodon casts Baton Pass
!Torrent: false
Shellos casts Foresight
!Splash: 169 39 -34
Gastrodon attacks Shellos with Pain Split
Gastrodon casts Baton Pass
!StickyHold: false true true true false true
Shellos attacks Gastrodon with Pain Split */
```

3. **Вывод:** Я разобрался с порядком инициализации полей класса, выяснил способы сравнения строк, немного познакомился с оптимизацией компилятора и представлением в чисел с плавающей точкой в виртуальной машине.