

ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №2

## Java Base Libraries

Выполнила:  
Юлия Прохорова

## Содержание

1. Цель работы	2
2. <b>Voice</b>	2
3. <b>Игра в кости</b>	3
4. <b>Extended Class</b>	5
5. <b>Black</b>	7
6. Вывод	8

## 1. Цель работы

Сформировать навыки проектирования и реализации интерфейсов Java, закрепить навыки в области разработки классов java и научиться переопределять методы equals(), hashCode(), toString().

## 2. Voice

- 1) **Задание.** Разработать программу с использованием интерфейсов и переопределить методы Java.
- 2) Реализация программы:
- 3) Voice.java:

```
public interface Voice {  
    void voice();  
}
```

- 4) Animals.java

```
class Cat implements Voice {  
  
    @Override  
    public void voice() {  
        System.out.println("Meow-meow");  
    }  
}  
  
class Dog implements Voice {  
  
    @Override  
    public void voice() {  
        System.out.println("Gav-gav");  
    }  
}  
  
class Cow implements Voice {  
  
    @Override  
    public void voice() {  
        System.out.println("Mu-mu");  
    }  
}  
  
public class Animals {  
  
    public static void main(String[] args) {  
        Cat cat = new Cat();  
        Dog dog = new Dog();  
        Cow cow = new Cow();  
  
        System.out.print("Cat say: ");  
        cat.voice();  
  
        System.out.print("Dog say: ");  
        dog.voice();  
  
        System.out.print("Cow say: ");  
        cow.voice();  
    }  
}
```

- 5) Результаты тестов:

```
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\voice> java Animals
Cat say: Meow-meow
Dog say: Gav-gav
Cow say: Mu-mu
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\voice> |
```

- 6) Чтобы перейти к реализации программы на Github, нажмите на название задачи в самом начале ее описания.

### 3. Игра в кости

- 1) **Задача.** Переработать задачу про игру в кости под использование интерфейсов. Игроют N игроков (компьютер в списке последний). Подкидываются одновременно K кубиков. Выигрывает тот, у кого большая сумма очков. Кто выиграл, тот и кидает первым в следующем кону. Игра идет до 7 выигрышей. Начинаете игру Вы.
- 2) Реализация программы:
- 3) Game.java:

```
import java.util.Scanner;

public interface Game {

    int getPlayers (Scanner in);
    int getBones (Scanner in);
    int[][] createArray (int n);
    int game(int[][] players, int n, int k);

}
```

- 4) Main.java:

```
import java.util.Random;
import java.util.Scanner;

class Bones implements Game{

    @Override
    public int getPlayers (Scanner in) {
        int n;
        System.out.println("Print the number of the players with the computer");
        //Scanner in = new Scanner(System.in);
        n = in.nextInt();
        //in.close();
        return n;
    }

    @Override
    public int getBones (Scanner in) {
        int k;
        System.out.println("Print the number of the bones");
        //Scanner inn = new Scanner(System.in);
        k = in.nextInt();
        //inn.close();
        return k;
    }

    @Override
    public int[][] createArray (int n) {
        int[][] players = new int[2][n];
        for (int i = 0; i < n; i++) {
            players[0][i] = i+1;
            players[1][i] = 0;
        }
    }
}
```

```

    }
    return players;
}

@Override
public int game(int[][] players, int n, int k) {

    int[] current = new int [n];
    int winner = 0;
    int max = 0;
    int flag = 0;

    final Random random = new Random();

    while (flag == 0) {
        for (int i = 0; i < n; i++){
            for (int j = 0; j < k; j++) {

                current[i] += random.nextInt(5) + 1;
            }
            if ( current[i] > max) {
                max = current[i];
                winner = i;
            }
        };
        players[1][winner]++;
        if (players[1][winner]==7){
            flag = 1;
        }
        else {
            int number = players[1][winner];
            for(int i = winner; i > 0; i--) {
                players[0][i] = players[0][i-1];
                players[1][i] = players[1][i-1];
            }
            players[0][0] = winner+1;
            players[1][0] = number;
            for (int i=0; i < n; i++) {
                current[i] = 0;
            }
        }
    };
    return players[0][0];
}

}

public class Main {

    public static void main (String[] args) {

        Scanner in = new Scanner(System.in);
        Bones bones = new Bones();
        int n = bones.getPlayers(in);
        int k = bones.getBones(in);
        in.close();

        int[][] players = bones.createArray(n);

        int winner = bones.game(players, n, k);

        System.out.println("The winner is " + winner + " player");

    };
}

```

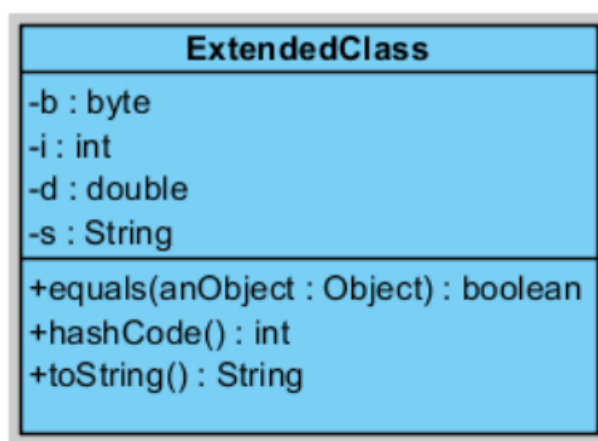
5) Результаты тестов:

```
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\bones> java Main
Print the number of the players with the computer
8
Print the number of the bones
3
The winner is 2 player
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\bones> java Main
Print the number of the players with the computer
7
Print the number of the bones
1
The winner is 1 player
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\bones>
```

6) Чтобы перейти к реализации программы на Github, нажмите на название задачи в самом начале ее описания.

## 4. Extended Class

1) **Задача.** Написать программу, реализующую изображенный класс:



2) Реализация программы:

3) ExtendedClass.java :

```
import java.util.Objects;

public class ExtendedClass {
    byte b;
    int i;
    double d;
    String s;

    @Override
    public boolean equals(Object anObject) {
        if (anObject == this) {
            return true;
        } else if (anObject == null || anObject.getClass() != this.getClass()) {
            return false;
        }

        ExtendedClass extendedClass = (ExtendedClass) anObject;
        if (this.b == extendedClass.b && this.i == extendedClass.i && this.d == extendedClass.d && thi
            return true;
    }
}
```

```

    }
    else {
        return false;
    }
}

@Override
public int hashCode() {
    int result = Objects.hashCode(b);
    result = 31 * result + Objects.hashCode(i);
    result = 31 * result + Objects.hashCode(d);
    result = 31 * result + Objects.hashCode(s);
    return result;
}

@Override
public String toString() {
    return "byte: " + b + "\n" + "int: " + i + "\n" + "double: " + d + "\n" + "String: " + s ;
}
}

```

4) Main.java :

```

import java.util.Objects;

public class Main {
    public static void main(String[] args) {

        ExtendedClass extended1 = new ExtendedClass();
        extended1.b = 1;
        extended1.i = 1;
        extended1.d = 0.2;
        extended1.s = "extended";

        ExtendedClass extended2 = new ExtendedClass();
        extended2.b = 1;
        extended2.i = 1;
        extended2.d = 0.2;
        extended2.s = "extended";

        ExtendedClass extended3 = new ExtendedClass();
        extended3.b = 0;
        extended3.i = 1;
        extended3.d = 0.2;
        extended3.s = "extended3";

        String s = "String";

        System.out.println(extended1.equals(extended1));
        System.out.println(extended1.equals(extended2));
        System.out.println(extended1.equals(extended3));
        System.out.println(extended1.equals(s));

        System.out.println("-----");

        System.out.println(extended1.hashCode() == extended1.hashCode());
        System.out.println(extended1.hashCode() == extended2.hashCode());
        System.out.println(extended1.hashCode() == extended3.hashCode());
        System.out.println(extended1.hashCode() == s.hashCode());

        System.out.println("-----");
        System.out.println(extended3.toString());

    }
}

```

5) Результаты тестов:

```
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\extended> java Main
true
true
false
false
-----
true
true
false
false
-----
byte: 0
int: 1
double: 0.2
String: extended3
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\extended>
```

6) Чтобы перейти к реализации программы на Github, нажмите на название задачи в самом начале ее описания.

## 5. Black

1) **Задача.** Создать интерфейс Black с методами setColor(String color) и isBlack(). Реализовать интерфейс в классе BlackImpl. Метод setColor(String color) должен устанавливать текущий цвет в color. Метод isBlack() должен печатать в консоль "It is black если текущий цвет == "black и "it isn't black" в противном случае.

2) Реализация программы:

3) Black.java :

```
public interface Black{

    void setColor(String color);
    void isBlack();
}
```

4) BlackImpl.java :

```
public class BlackImpl implements Black {
    String color;

    @Override
    public void setColor(String color) {
        this.color=color;
    }

    @Override
    public void isBlack() {
        if (this.color == "black"){
            System.out.println("It is black");
        }
        else {
            System.out.println("It isn't black");
        }
    }
}
```

5) Main.java :

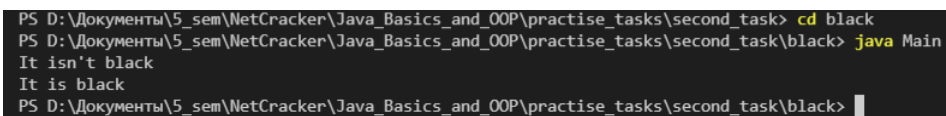
```
public class Main{
    public static void main(String[] args) {
        Black black = new BlackImpl();

        black.setColor("blue");
        black.isBlack();
    }
}
```



```
        black.setColor("black");  
        black.isBlack();  
    }  
}
```

6) Результаты тестов:



```
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task> cd black  
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\black> java Main  
It isn't black  
It is black  
PS D:\Документы\5_sem\NetCracker\Java_Basics_and_OOP\practise_tasks\second_task\black> █
```

7) Чтобы перейти к реализации программы на Github, нажмите на название задачи в самом начале ее описания.

## 6. Вывод

В ходе выполнения практического задания были сформированы навыки проектирования и реализации интерфейсов Java, закреплены навыки в области разработки классов java и переопределены методы equals(), hashCode(), toString().