NetCracker Learning center

Учебное практическое задание № 2

Разработать программу с использованием интерфейсов и переопределить методы Java.

выполнил студент

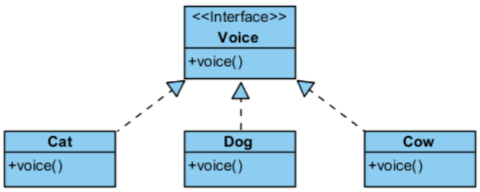
Яромир Водзяновский

Долгопрудный, 2021

Цель работы: Сформировать навыки проектирования и реализации интерфейсов Java, закрепить знания в области разработки классов java и научиться переопределять методы eduals(), hashCode(), toString().

1 Задача №1

Задание: Напишите программу, реализующую следующую диаграмму классов:



Main.java

```
package task_1;

class Cow implements Voice {
    @Override
    public void voice() {
        System.out.println("Myy-myy");
    }
}

class Dog implements Voice {
    @Override
    public void voice() {
        System.out.println("Gav-gav");
    }
}

class Cat implements Voice {
    @Override
    public void voice() {
        System.out.println("Mow-mow");
    }
}
public class Main {
    public static void main(String[] args) {
        Cat cat = new Cat();
        Dog dog = new Dog();
        Cow cow = new Cow();

        System.out.println("Cat: ");
        cat.voice();
        System.out.println("Dog: ");
        dog.voice();
        System.out.println("Cow: ");
        cow.voice();

    }
}
```

Voice.java

```
package task_1;

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

2 Задача №2: Игра в кости

Задание: Переработать задачу про игру в кости под использование интерфейсов. Играют 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.

Bones.java

```
package task_2;
```

```
public interface Bones {
    int Play(int K);
    int[] maxArray(int[] Array);
    int[][] Substitute(int[][] PlayerList, int maxScoreIndex);
    void game();
}
```

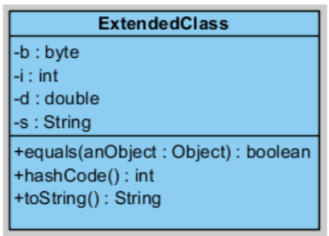
Main.java

```
package task_2;

import java.util.Scanner;

class BonesImpl implements Bones {
    @Override
    public int Play(int K) {
        int score = 0;
        for (int j=0; j<K; j++) {
            score += (int) (Math.random() * 6 + 1);
        }
        return score;
    }

    @Override
    public int[] maxArray(int[] Array) {
        int index = 0;
        int max = 0;
        int[] maxArray = new int[2];
        for (int i = 0; i<Array.length; i++) {
            if (max <= Array[i]) {
                max = Array[i];
                index = i;
            }
        }
        maxArray[0] = index;
        maxArray[1] = max;
        return maxArray;
    }

    @Override
    public int[][] Substitute(int[][] PlayerList, int maxScoreIndex) {
        int[] save_winner = PlayerList[maxScoreIndex];
        for (int k=0; k<maxScoreIndex; k++) {
            PlayerList[maxScoreIndex - k] = PlayerList[maxScoreIndex - k-1];
        }
        PlayerList[0] = save_winner;
        return PlayerList;
    }

    @Override
    public void game() {

        Scanner in = new Scanner(System.in);
        int N;
        int K;
        int maxScoreIndex;

        System.out.print("Enter number of players: ");
        N = in.nextInt();

        System.out.print("Enter number of cubes: ");
        K = in.nextInt();
        in.close();

        int[] Scores = new int[N];
```

```
        int[][] PlayerList = new int[N][2];
        int[] TotalScores = new int[N];
        int WinnerIndex = 0;


        for (int i=0; i<N; i++) {
            PlayerList[i][0] = i;
            PlayerList[i][1] = 0;
        }

        for (int round=0; round<7; round++) {

            System.out.println("Round: " + (round+1));
            for (int player_num=0; player_num<N; player_num++) {
                Scores[player_num] = Play(K);
            }
            maxScoreIndex = maxArray(Scores)[0];

            PlayerList[maxScoreIndex][1] += 1;

            for (int k=0; k<N; k++) {
                System.out.println("Player " + PlayerList[k][0] + " has score: " + Scores[k]);
            }

            PlayerList = Substitute(PlayerList, maxScoreIndex);
        }

        System.out.println("Outcomes: ");

        for (int k=0; k<N; k++) {
            System.out.println("Player " + PlayerList[k][0] + " has number of victories: " + PlayerList[k][1]);
        }

        for (int i=0; i<N; i++) {
            TotalScores[i] = PlayerList[i][1];
        }

        WinnerIndex = maxArray(TotalScores)[0];

        System.out.println("Winner of the Game: Player " + PlayerList[WinnerIndex][0]);

    }
}

public class Main {
    public static void main(String[] args) {
        Bones bones = new BonesImpl();
        bones.game();
    }
}
```

3 Задача №3

Задание: Напишите программу, реализующую изображенный класс:



4 Задача №4, Вариант B

Задание: Создайте интерфейс Sleepyc методами sleep(), wakeUp() и ask(). Реализуйте интерфейс в классе SleepyImpl. Метод sleep() устанавливает флаг awake в false, метод wakeUp() в true. Метод ask() печатает в консоль “BOO!”, если флаг установлен в true, и “zzz...” в противном случае.

Sleepyc.java

```
package task_4;

public interface Sleepyc {
    void sleep();
    void wakeUp();
    void ask();
}
```

Main.java

```
package task_4;

public class Main {
    public static void main(String[] args) {
        Sleepyc sleepyc = new SleepyImpl();

        sleepyc.sleep();
        sleepyc.ask();

        sleepyc.wakeUp();
        sleepyc.ask();
    }
}
```

SleepyImpl.java

```
package task_4;

public class SleepyImpl implements Sleepyc {
    boolean awake;

    @Override
    public void sleep() {
        this.awake = false;
    }

    @Override
    public void wakeUp() {
        this.awake = true;
    }

    @Override
    public void ask() {
        if (this.awake == true) {
            System.out.println("BOO!");
        } else {
            System.out.println("zzz...");
        }
    }
}
```