Урок 10. Программы со сложной логикой, пишем крестики-нолики.

Цель задания:

Формирование компетенций по работе со сложными программами, их анализ, самостоятельное написание по заданному техническому заданию

Задание:

методический материал:

класс X0Game

```
package myclass26x0;

import java.util.Scanner;

public class X0Game {
   FieldX0 gameField;
   Scanner scanner = new Scanner(System.in);
   char whoMakeNextTurn;
   boolean gameOver = false;

   void setupNewGame() {
       System.out.println("Will play new X0 game");
       this.gameField = new FieldX0();
       this.gameField.initField();
   }

   void play() {
       this.setupNewGame();
       System.out.println("Who will make first turn: ");
       char first = this.scanner.next().charAt(0);
       if (first == 'X' || first == '0') {
           this.whoMakeNextTurn = first;
       } else {
           System.out.println("I recognizing only X and 0 (zero). So first will be X");
           this.whoMakeNextTurn = 'X';
       }

       while (!gameOver) {
           turn();
           this.gameOver = this.gameField.isGameOver(this.whoMakeNextTurn);
           if (this.gameOver) {
               System.out.println(this.whoMakeNextTurn + " win!");
           }
           if (this.whoMakeNextTurn == 'X') {
               this.whoMakeNextTurn = '0';
           } else {
               this.whoMakeNextTurn = 'X';
           }
       }
       System.out.println("Game over.");
   }

   void turn() {
       System.out.println(this.whoMakeNextTurn + ", your turn. ");
       System.out.print("Chose row: ");
       int rowNumber = this.scanner.nextInt();
       System.out.print("Chose column: ");
       int colNumber = this.scanner.nextInt();
       int rowIndex = rowNumber - 1;
       int colIndex = colNumber - 1;
       if (this.gameField.isPlaceFree(rowIndex, colIndex)) {
           this.gameField.setValue(rowIndex, colIndex, whoMakeNextTurn);
```

```
            this.gameField.printField();
        } else  {
            System.out.println("Wrong number (maybe this place is not free?). Make turn
again");
            turn();
        }
    }
}
```

Класс FieldX0

```
package myclass26x0;

public class FieldX0 {
    char[][] field;
    int size = 9;
    int countToWin = 5;

    void initField() {
        this.field = new char[size][size];
        for (int row = 0; row < size; row++) {
            for (int col = 0; col < size; col++) {
                field[row][col] = ' ';
            }
        }
        System.out.println("Field initialized");
        this.printField();
    }

    void printField() {
        System.out.print("   ");
        for (int i = 1; i <= size; i++) {
            System.out.print(i + "  ");
        }
        System.out.println();
        for (int row = 0; row < size; row++) {
            int rowNumber = row + 1;
            System.out.print(rowNumber + " ");
            for (int col = 0; col < size; col++) {
                System.out.print("[" + this.field[row][col] + "]");
            }
            System.out.println();
        }
    }

    boolean isPlaceFree(int rowIndex, int colIndex) {
        if (rowIndex<0 || rowIndex>=size || colIndex<0 || colIndex>=size) {
            return false;
        } else if (this.field[rowIndex][colIndex] == ' ') {
            return true;
        } else {
            return false;
        }
    }

    void setValue(int rowIndex, int colIndex, char value) {
        this.field[rowIndex][colIndex] = value;
    }

    boolean isGameOver(char player) {
        for (int row=0; row < this.size; row++) {
            for (int col=0; col < this.size; col++) {
                if (checkRightDirection(row, col, player)) {
                    return true;
                } else if (checkDownDirection(row, col, player)) {
```

```
                return true;
            } else if (checkRightDiagonal(row, col, player)) {
                return true;
            } else if (checkLeftDiagonal(row, col, player)) {
                return true;
            }
        }
    }
    return false;
}

boolean checkRightDirection(int row, int col, char player) {
    if (col > this.size - this.countToWin) {
        return false;
    }
    for (int i = col; i < col + this.countToWin; i++) {
        if (this.field[row][i] != player) {
            return false;
        }
    }
    return true;
}

boolean checkDownDirection(int row, int col, char player) {
    if (row > this.size - this.countToWin) {
        return false;
    }
    for (int i = row; i < row + this.countToWin; i++) {
        if (this.field[i][col] != player) {
            return false;
        }
    }
    return true;
}

boolean checkRightDiagonal(int row, int col, char player) {
    if (row > this.size - this.countToWin) {
        return false;
    }
    if (col > this.size - this.countToWin) {
        return false;
    }
    for (int sdvig = 0; sdvig < this.countToWin; sdvig++) {
        int rowToCheck = row + sdvig;
        int colToCheck = col + sdvig;
        if (this.field[rowToCheck][colToCheck] != player) {
            return false;
        }
    }
    return true;
}

boolean checkLeftDiagonal(int row, int col, char player) {
    if (row > this.size - this.countToWin) {
        return false;
    }
    if (col < this.countToWin - 1) {
        return false;
    }
    for (int sdvig = 0; sdvig < this.countToWin; sdvig++) {
        int rowToCheck = row + sdvig;
        int colToCheck = col - sdvig;
        if (this.field[rowToCheck][colToCheck] != player) {
            return false;
        }
    }
    return true;
```

```
    }
}
```

1. Напишите сами крестики-нолики, не подглядывая в наш код.
2. Напишите “змейку”. Есть поле 20х20, есть змейка длинной, пусть, 3 квадрата. Выводится поле с положением змейки. Пользователь вводит, куда сделать следующий шаг - повернуть, или не двигаться. Рисуется поле с новым положением змейки.
3. Доработайте змейку, что б при врезании в стену(край поля) - игра оканчивалась
4. Доработайте змейку, что б на поле были фрукты. Если змейка их съедает - она растет. Если съела все фрукты - игрок победил.
5. Доработайте змейку, пусть на поле еще могут быть стены. в них тоже нельзя врезаться

Критерии оценивания:

1 балл- создан новый проект в IDE

2 балла - написана общая структура программы

3 балла - написана программа “змейка” без дополнительного функционала прописанного в пункте 3,4,5 данного задания

4 балла - написана программа “змейка” с дополнительным функционалом, включающая в себя, как минимум 3,4 пункт технического задания

5 баллов - все технические задания выполнены корректно, в полном объеме, дополнительный функционал прописанный в пункте 3, 4, 5 выполнен корректно, в полном объеме

Задание считается выполненным при условии, что слушатель получил оценку не менее 3 баллов