**Задание.**

Написать игру «камень-ножницы-бумага» с возможностями:

* сыграть раунд против компьютера (ввод выбора пользователя, вывод результата)
* результат каждой игры должен записывать в файл (кто победил и какие выборы были у соперников)
* вывод результатов всех игр на консоль (из файла)

Бонусное задание:

* Реализовать методы статистического анализа истории игр из файла, такие как вывод наиболее частого выбора, вывод кол-ва побед/проигрышей компьютера или игрока и т.д.

**Program code**

Class GamePlace

package com.company;  
  
  
import java.io.\*;  
import java.nio.charset.StandardCharsets;  
import java.util.Objects;  
  
// Class "GamePlace"  
public class GamePlace {  
  
 // Fields  
  
 private final String PAPER = "PAPER"; // PAPER  
 private final String SCISSORS = "SCISSORS"; // SCISSORS  
 private final String STONE ="STONE"; // STONE  
 private static final String *file* = "game.txt"; // file to write  
  
 private Gamer gamer;  
 private Computer computer;  
 private int gamerScore; // keep player score  
 private int computerScore; // keep computer score  
 private int numberOfGames; // counts how many times it has been played  
 private int draw; //  
  
  
  
 // Constructor  
 public GamePlace() {  
 gamer = new Gamer();  
 computer = new Computer();  
 gamerScore = 0;  
 computerScore = 0;  
 numberOfGames = 0;  
 draw = 0;  
 }  
  
  
 //Method to start game  
 public void startTheGame() throws IOException {  
 System.*out*.println("Game: STONE, SCISSORS, PAPER is started");  
 *writeGameResult*("Game: STONE, SCISSORS, PAPER is started");  
 String userMove = gamer.makeARunningPlayer();  
 String computerMove = Computer.*randomComputerChoice*();  
 System.*out*.println("Your turn " + userMove + ".");  
 System.*out*.println("Computer turn " + computerMove);  
 // if player win  
 if((userMove.equals("PAPER")) && (computerMove.equals("STONE")) ||  
 (userMove.equals("SCISSORS")) && (computerMove.equals("PAPER")) ||  
 (userMove.equals("STONE")) && (computerMove.equals("SCISSORS"))) {  
 System.*out*.println(" You choice " + userMove + " beats "  
 + " computer choice " + computerMove + " You win");  
 gamerScore++;  
 *writeGameResult*(" You choice " + userMove + " beats "  
 + " computer choice " + computerMove + " You WIN");  
 // if computer and player choice similar things = draw  
 } if (userMove.equals(computerMove)) {  
 draw++;  
 System.*out*.println("Draw");  
 *writeGameResult*( "Draw" );  
 // if computer win  
 } else  
 System.*out*.println("computer choice " + computerMove + " beats "  
 + " You choice " + userMove + " You lose");  
 computerScore++;  
 *writeGameResult*("computer choice " + computerMove + " beats "  
 + " You choice " + userMove + " You LOSE");  
 numberOfGames++; // count game  
 if (gamer.playAgain()) {  
 System.*out*.println();  
 startTheGame();  
 } else {  
 printGameStats();  
 }  
 }  
   
  
 // Method to print  
 public void printGameStats() throws IOException {  
 int wins = gamerScore;  
 int loses = computerScore;  
 // int ties = numberOfGames - gamerScore - computerScore;  
 // double percentageWon = (wins + ((double) ties) / 2) / numberOfGames;  
 System.*out*.println("Wins = " + wins);  
 System.*out*.println("Loses = " + loses);  
 System.*out*.println("Draw = " + draw);  
 // System.out.println("Ties = " + ties);  
 //System.out.println("Percent of Won " + percentageWon);  
 *writeGameResult*("Wins = " + wins);  
 *writeGameResult*("Loses = " + loses);  
 *writeGameResult*("Draw = " + draw);  
 // writeGameResult("Ties = " + ties);  
 // writeGameResult("Percent of Won " + percentageWon);  
 }  
  
 // Getters  
 public String getPAPER() {  
 return PAPER;  
 }  
 public String getSCISSORS() {  
 return SCISSORS;  
 }  
 public String getSTONE() {  
 return STONE;  
 }  
  
  
  
  
 @Override  
 public String toString() {  
 return "GamePlace{" +  
 "PAPER='" + PAPER + '\'' +  
 ", SCISSORS='" + SCISSORS + '\'' +  
 ", STONE='" + STONE + '\'' +  
 '}';  
 }  
   
  
 public static void writeGameResult(String s) throws IOException {  
 OutputStream out = null;  
 try {  
 out = new FileOutputStream(*file*, true);  
 out.write((s + "\n").getBytes(StandardCharsets.*UTF\_8*));  
 } catch (FileNotFoundException ex) {  
 ex.printStackTrace();  
 }  
 finally {  
 try {  
 if(out != null) {  
 out.close();  
 }  
 } catch(IOException ex) {  
 System.*out*.println("Failed to close file: " + ex.getMessage());  
 }  
 }  
 }  
  
 public void inputResult() {  
 try (FileInputStream file = new FileInputStream("GamesResult.txt")) {  
 int c;  
 while ((c = file.read()) != -1) {  
 System.*out*.print((char) c);  
 }  
 } catch (IOException ex) {  
 ex.printStackTrace();  
 }  
 }  
  
  
} // Class "GamePlace"

Class Computer

package com.company;  
  
import java.util.Random;  
  
// Class "Computer"  
public class Computer {  
  
 private static final String *file* = "game.txt";  
  
 public static String randomComputerChoice() {  
 String [] arr = {"STONE", "SCISSORS", "PAPER"};  
 Random r = new Random();  
 String choice = arr[r.nextInt(3)];  
 System.*out*.println(choice);  
 return choice;  
 }  
  
} // Class "Computer"

Class Gamer

package com.company;  
  
import java.io.IOException;  
import java.util.Scanner;  
  
// Class "Gamer"  
public class Gamer {  
  
// String inputName = new String(String.valueOf(System.in));  
 private Scanner inputPlayer;  
  
 //  
  
 public Scanner getInputPlayer() {  
 return inputPlayer;  
 }  
  
 // Constructor  
 public Gamer() {  
 inputPlayer = new Scanner(System.*in*);  
 }  
  
 //Method to player make choice: STONE, PAPER or SCISSORS  
 public String makeARunningPlayer() throws IOException {  
 System.*out*.println("Make choice: 1 - \"STONE\", 2 - \"SCISSORS\", 3 - \"PAPER\" ?");  
 GamePlace.*writeGameResult*("Make choice: 1 - \"STONE\", 2 - \"SCISSORS\", 3 - \"PAPER\" ?");  
 char input = inputPlayer.next().charAt(0);  
 if (input == '1' || input == '2' || input == '3') {  
 switch(input) {  
 case '1':  
 return "STONE";  
 case '2':  
 return "SCISSORS";  
 case '3':  
 return "PAPER";  
 }  
 }  
  
 return makeARunningPlayer();  
 }  
  
 // Method to ask player want he play the game again  
 public boolean playAgain() {  
 System.*out*.println("Do you want play again? 1 - \"YES\" or 2 - \"NO\"");  
 char userInput = inputPlayer.next().charAt(0);  
 return userInput == '1';  
 }  
  
  
  
} // Class "Gamer"

Main

package com.company;  
  
import java.io.IOException;  
  
public class Main {  
  
 public static void main(String[] args) throws IOException {  
  
 GamePlace game = new GamePlace();  
 game.startTheGame();  
  
 }  
}

Test

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

TXT variant

Изображение выглядит как текст

Автоматически созданное описание