**Задание 1**

Есть сайт, на котором рассчитывается рейтинг игроков различных сетевых игр.Игрок при регистрации указывает ник, а так же список игр, в которые он играет.

Задача – написать программу, которая:

- регистрирует игроков в системе (должна быть проверка,

занят ли ник);

- добавляет рейтинг игроку, в случае его выигрыша в игре;

- выводит список игр, в которые играют все игроки на сайте;

- выводит рейтинг по имени игрока и игре;

- выводит 10 лучших игроков в определенной игре;

- выводит 10 лучших игроков с учетом всех игр.

***class Playground***

package com.company;  
  
import java.util.ArrayList;  
import java.util.Scanner;  
  
//Класс "Игровая площадка онлайн"  
public class Playground {  
  
 //поля  
 ArrayList<String> playerList = new ArrayList<>(); //список никнеймов игроков  
 ArrayList<String> gameList = new ArrayList<>();// список игр на площадке  
 ArrayList<GameTable> ratingList = new ArrayList<>();//список рейтингов игроков  
  
  
 //Сканер для ввода игроков  
 Scanner in = new Scanner(System.*in*);  
  
 //методы  
 // 1. добавить игрока в список игроков  
 public String addPlayer(String *playerName*) {  
 String *newName* = *playerName*;  
 if (playerList.size() == 0) {  
 playerList.add(*playerName*);  
 } else  
 for (String *item* : playerList) {  
 if (*item*.equals(*newName*)) {  
 int *rand* = (int) (Math.*random*() \* 1000);  
 *newName* += "@" + *rand*;  
 System.*out*.println("Ваш новый никнейм: " + *newName*);  
 playerList.add(*newName*);  
 return *newName*;  
 }  
 playerList.add(*newName*);  
 break;  
 }  
 return *newName*;  
 }  
  
 // 2. добавить игру в список игр  
 public boolean addGame(String *gameName*) {  
 for (String *game* : this.gameList) {  
 if (*game* == *gameName*) {  
 return false;  
 }  
 }  
 this.gameList.add(*gameName*);  
 return true;  
 }  
  
 // 3. показать список игроков  
 public void showPlayerList() {  
 for (int *i* = 0; *i* < playerList.size(); *i*++) {  
 System.*out*.println(*i* + 1 + ")" + "." + playerList.get(*i*));  
 }  
 }  
  
 // 4. выбрать игрока по индексу  
 public int indexPlayer() {  
 System.*out*.println("Выберите пользователя: ");  
 showPlayerList();//отобразить список игроков на экране  
 return in.nextInt();// номер,который вводится с клавиатуры  
 }  
  
  
 // 5. показать список игр  
 public void showGameList() {  
 for (int *i* = 0; *i* < gameList.size(); *i*++) {  
 System.*out*.println(*i* + 1 + ")" + "." + gameList.get(*i*));  
 }  
 }  
  
 // 6. выбрать игру по индексу  
 public int indexGame() {  
 System.*out*.println("Выберите игру: ");  
 showGameList();//отобразить список игр на экране  
 return in.nextInt();// номер,который вводится с клавиатуры  
 }  
  
 // 7. добавить рейтинг игроку в игре  
 public int getRating(int *userIndex*, int *gameIndex*) {  
 int *res* = 0;  
 GameTable *p1*; // запись таблицы  
 for (GameTable *table* : ratingList) {  
 *p1* = *table*;  
 if (*p1*.playerIndex == *userIndex* && *p1*.gameIndex == *gameIndex*)  
 *res* += *p1*.rating;  
  
 }  
 return *res*;  
 }  
}

***class GameTable***

package com.company;  
  
//класс "Игровая таблица"  
public class GameTable implements Comparable<GameTable> {  
  
 // поля  
 int playerIndex; // индекс игрока в списке игроков  
 int gameIndex; // индекс игры в списке игр  
 int rating; // рейтинг пользователя  
  
  
 // Конструкторы  
 // 1. Конструктор с 3-мя параметрами  
 public GameTable(int *userIndex*, int *gameIndex*, int *rating*) {  
 this.playerIndex = *userIndex*;  
 this.gameIndex = *gameIndex*;  
 this.rating = *rating*;  
 }  
  
 //метод для сравнения объектов  
 @Override  
 public int compareTo(GameTable *obj*) {  
 return (this.rating - *obj*.rating);  
 }//возвращаем разницу рейтингов  
}

***class PlaygroundDatabase***

package com.company;  
  
import java.util.Collections;  
import java.util.Scanner;  
  
//Класс "База данных игровой площадки"  
public class PlaygroundDatabase {  
  
 //поля  
  
 private Playground ratings = new Playground(); // рейтинг  
 private Playground player = new Playground(); // игрок  
 private Playground game = new Playground(); // игра  
  
 //сканер для ввода пунктов меню  
 Scanner in = new Scanner(System.*in*);  
  
 //методы  
 // 1. добавить игрока игрока  
 public String addPlayer(String *name*) {  
 return player.addPlayer(*name*);  
 }  
  
 // 2. добавить игру  
 public boolean addGame(String *name*) {  
 return game.addGame(*name*);  
 }  
  
 // 3. добавить рейтинг победителю  
 public void addRatingWinner(int *userIndex*, int *gameIndex*) {  
 GameTable *p1*;  
 //поиск пользователя в игре  
 int *index* = -1;  
 for (int *i* = 0; *i* < ratings.ratingList.size(); *i*++) {  
 *p1* = ratings.ratingList.get(*i*);  
 if (*p1*.gameIndex == *gameIndex* && *p1*.playerIndex == *userIndex*) {  
 *index* = *i*;  
 }  
 }  
 if (*index* == -1) {  
 ratings.ratingList.add(new GameTable(*gameIndex*, *userIndex*, 0));  
 *index* = ratings.ratingList.size() - 1;  
 }  
 int *ratingOld* = ratings.ratingList.get(*index*).rating;  
 int *ratingNew* = *ratingOld* + 1;  
 ratings.ratingList.set(*index*, new GameTable(*gameIndex*, *userIndex*, *ratingNew*));  
 }  
  
  
 // 4. выбрать игрока по индексу  
 public int indexPlayer() {  
 return player.indexPlayer();  
 }  
  
 // 5. выбрать игру по индексу  
 public int indexGame() {  
 return game.indexGame();  
 }  
  
 // 6. показать список игр  
 public void showGamesAll() {  
 game.showGameList();  
 }  
  
 // 7. показать список игроков  
 public void showPlayerAll() {  
 player.showPlayerList();  
 }  
  
 // 7. добавить рейтинг игроку в игре  
 public int getRatingList(int *userIndex*, int *gameIndex*) {  
 return ratings.getRating(*userIndex*, *gameIndex*);  
 }  
  
 // 8. вывод 10 лучших игроков в определенной игре  
 public void showBestByGame(int *gameIndex*) {  
 Playground *table* = new Playground();  
 GameTable *p1*;  
 //проход по таблице рейтингов  
 for (int *i* = 0; *i* < ratings.ratingList.size(); *i*++) {  
 *p1* = ratings.ratingList.get(*i*);  
 if (*p1*.gameIndex == *gameIndex*)  
 *table*.ratingList.add(*p1*);  
 }  
  
 Collections.*sort*(*table*.ratingList);  
 Collections.*reverse*(*table*.ratingList);  
 System.*out*.println("рейтинг игроков в игре " + game.gameList.get(*gameIndex*));  
 for (int *i* = 0; *i* < 10; *i*++) {  
 if (*i* < *table*.ratingList.size()) {  
 *p1* = *table*.ratingList.get(*i*);  
 System.*out*.print(player.playerList.get(*p1*.playerIndex) + " рейтинг: ");  
 System.*out*.println(*p1*.rating);  
 }  
 }  
 }  
  
 // 9. вывод 10 лучших игроков во всех играх  
 public void showBestByAllGame() {  
 Playground *table* = new Playground();  
 GameTable *p1*;  
 //проход по таблице пользователей, и создание рейтинга каждого пользователя  
 for (int *i* = 0; *i* < player.playerList.size(); *i*++) {  
 *p1* = new GameTable(0, *i*, 0);  
 *table*.ratingList.add(*p1*);  
  
 }  
 // проход по таблице рейтингов  
 for (int *i* = 0; *i* < ratings.ratingList.size(); *i*++) {  
 *p1* = ratings.ratingList.get(*i*);  
 int *user* = *p1*.playerIndex;  
 int *ratingOld* = *table*.ratingList.get(*user*).rating;  
 int *ratingNew* = *ratingOld* + *p1*.rating;  
 *table*.ratingList.set(*user*, new GameTable(*user*, *user*, *ratingNew*));  
 }  
 Collections.*sort*(*table*.ratingList);//сортировка  
 Collections.*reverse*(*table*.ratingList);//меняется парядок от максимального к минимальному  
 System.*out*.println("Общий рейтинг игроков на игровой площадке: ");  
 for (int *i* = 0; *i* < 10; *i*++) {  
 if (*i* < *table*.ratingList.size()) {  
 *p1* = *table*.ratingList.get(*i*);  
 System.*out*.print(player.playerList.get(*p1*.playerIndex) + " рейтинг: ");  
 System.*out*.println(*p1*.rating);  
 }  
  
 }  
 }  
  
 // 10. вывод меню  
 public void showMenu() {  
 Menu.*showMenu*();  
 }  
}

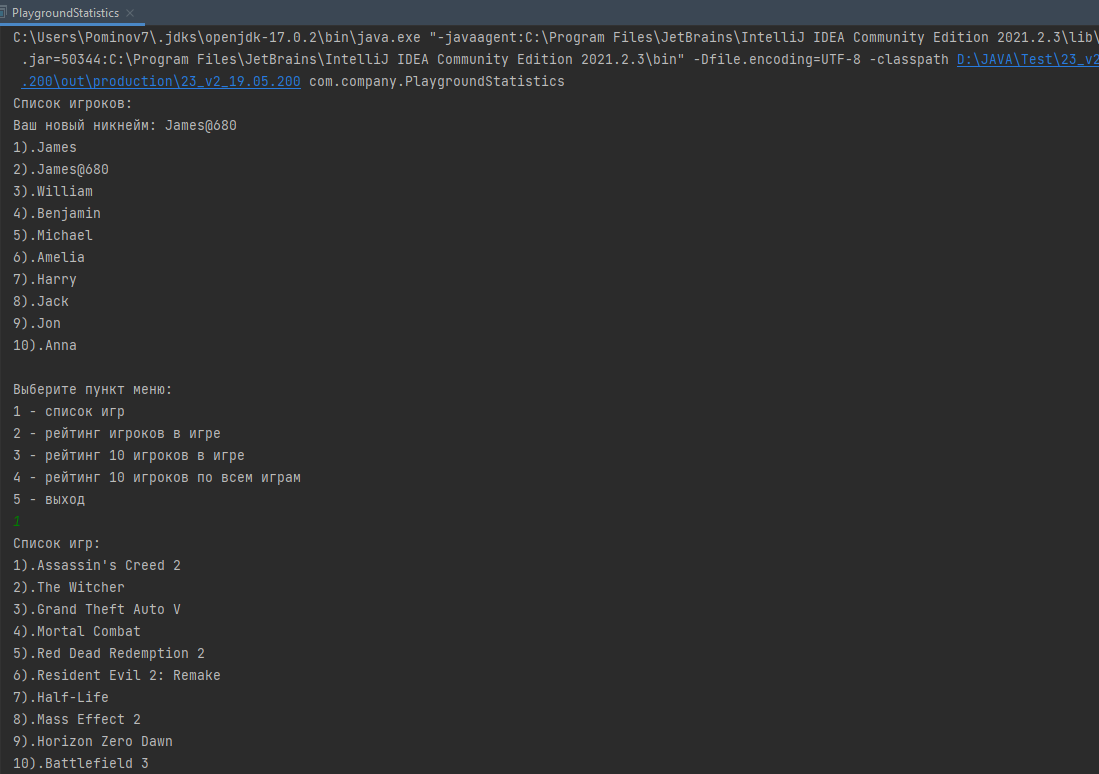
***class Menu***

package com.company;  
  
import java.util.Random;  
import java.util.Scanner;  
  
// Класс "Игровое меню"  
public class Menu {  
 public static void showMenu() {  
 Scanner *in* = new Scanner(System.*in*);  
  
 //Создали базу данных игровой площадки  
 PlaygroundDatabase *base* = new PlaygroundDatabase();  
  
 //Создали переменную для обработки меню  
 String *menu*;  
  
 //Создаем 10 игр и добавляем их в список игр  
 *base*.addGame("Assassin's Creed 2");  
 *base*.addGame("The Witcher");  
 *base*.addGame("Grand Theft Auto V");  
 *base*.addGame("Mortal Combat");  
 *base*.addGame("Red Dead Redemption 2");  
 *base*.addGame("Resident Evil 2: Remake");  
 *base*.addGame("Half-Life");  
 *base*.addGame("Mass Effect 2");  
 *base*.addGame("Horizon Zero Dawn");  
 *base*.addGame("Battlefield 3");  
  
 //Создаем 10 игроков и добавляем их в список игр  
 System.*out*.println("Список игроков: ");  
 *base*.addPlayer("James");  
 *base*.addPlayer("James");  
 *base*.addPlayer("William");  
 *base*.addPlayer("Benjamin");  
 *base*.addPlayer("Michael");  
 *base*.addPlayer("Amelia");  
 *base*.addPlayer("Harry");  
 *base*.addPlayer("Jack");  
 *base*.addPlayer("Jon");  
 *base*.addPlayer("Anna");  
 *base*.showPlayerAll();  
  
  
 //Рандомное присвоение рейтинга победителю  
 Random *rand* = new Random();  
 for (int *i* = 0; *i* < 1000; *i*++) {  
 *base*.addRatingWinner(*rand*.nextInt(10), *rand*.nextInt(10));  
 }  
  
 //меню  
 while (true) {  
 System.*out*.println("\nВыберите пункт меню: ");  
 System.*out*.println("1 - список игр ");  
 System.*out*.println("2 - рейтинг игроков в игре ");  
 System.*out*.println("3 - рейтинг 10 игроков в игре ");  
 System.*out*.println("4 - рейтинг 10 игроков по всем играм ");  
 System.*out*.println("5 - выход ");  
 *menu* = *in*.nextLine();  
  
 if ("1".equals(*menu*)) {  
 System.*out*.println("Список игр: ");  
 *base*.showGamesAll();  
 }  
  
 if ("2".equals(*menu*)) {  
  
 int *playerIndex* = *base*.indexPlayer();//выбор игрока  
 int *gameIndex* = *base*.indexGame();//выбор игры  
 System.*out*.println("Рейтинг: " + *base*.getRatingList(*playerIndex*, *gameIndex*));  
 }  
  
 if ("3".equals(*menu*)) {  
 int *gameIndex* = *base*.indexGame();  
 *base*.showBestByGame(*gameIndex*);  
 }  
  
 if ("4".equals(*menu*)) {  
 *base*.showBestByAllGame();  
 }  
  
 if ("5".equals(*menu*)) {  
 System.*out*.println("До встречи в новых играх! ");  
 break;  
 }  
 }  
 }  
}

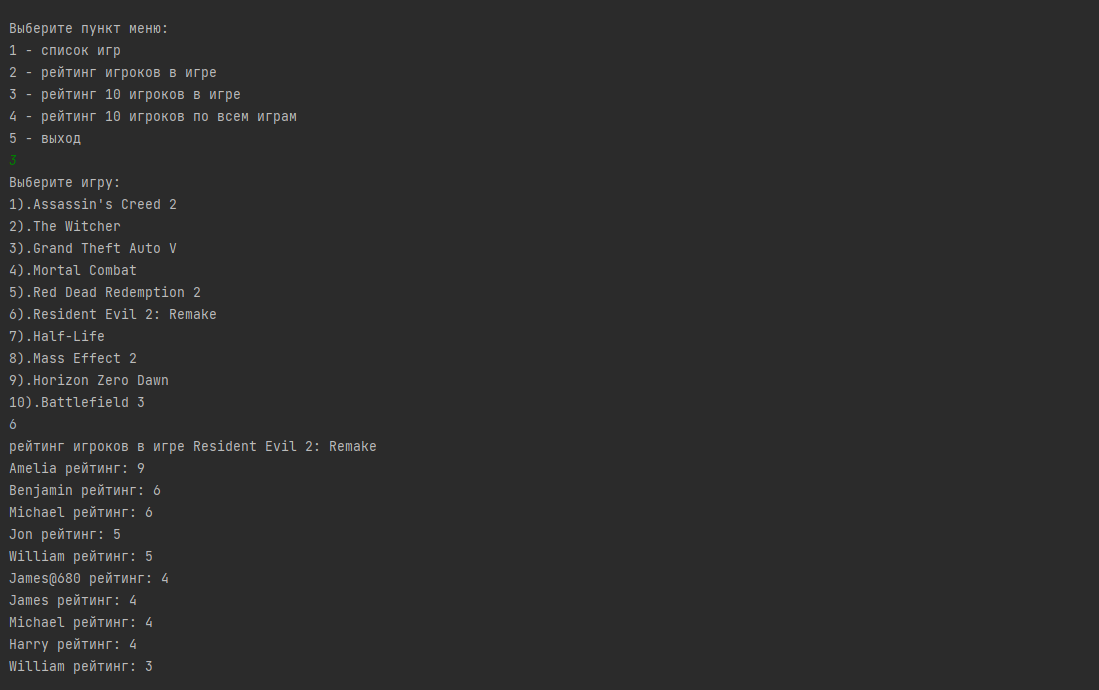
***class PlaygroundStatistics***

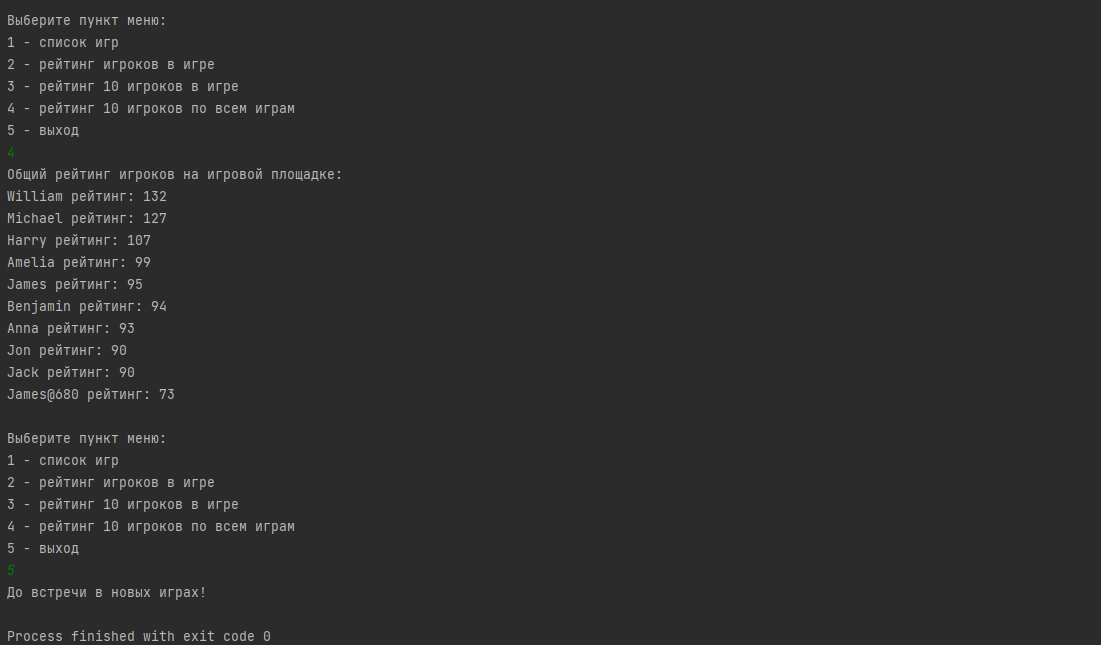
package com.company;  
  
//Класс "Статистика игровой площадки"  
  
public class PlaygroundStatistics {  
 public static void main(String[] *args*) throws Exception {  
  
 //Создали базу данных игровой площадки  
 PlaygroundDatabase *base* = new PlaygroundDatabase();  
  
 //вызвали меню игровой площадки  
 *base*.showMenu();  
 }  
}

***Тест:***

******

******

******

******