|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Отчет по выполнению практической работы 7

**Тема. ИСПОЛЬЗОВАНИЕ СТАНДАРТНЫХ КОНТЕЙНЕРНЫХ КЛАССОВ ПРИ ПРОГРАММИРОВАНИИ НА JAVA**

Дисциплина Программирование на языке Джава

Выполнил

|  |  |
| --- | --- |
| студент | Болотов Михаил |
|  | Фамилия И.О. |
| группа | ИКБО-06-19 |
|  | Номер группы |

Москва 2020

**Содержание**

[​ Теория 2](#__RefHeading___Toc4128_811265792)

[​ Задание 2](#__RefHeading___Toc4136_811265792)

[​ Код 3](#__RefHeading___Toc35859_811265792)

[​ Drunk.java 3](#__RefHeading___Toc42618_811265792)

[​ Скриншот 5](#__RefHeading___Toc3748_811265792)

[​ Заключение 5](#__RefHeading___Toc3750_811265792)

[​ Библиографический список 5](#__RefHeading___Toc3752_811265792)

# Теория

Java Collections Framework — это набор связанных классов и интерфейсов, реализующих широко используемые структуры данных — коллекции. На вершине иерархии в Java Collection Framework располагаются 2 интерфейса: Collection и Map. Эти интерфейсы разделяют все коллекции, входящие в фреймворк на две части по типу хранения данных: простые последовательные наборы элементов и наборы пар «ключ — значение» (словари).

Vector — реализация динамического массива объектов. Позволяет хранить любые данные, включая null в качестве элемента. Vector появился в JDK версии Java 1.0, но, как и Hashtable, эту коллекцию не рекомендуется использовать, если не требуется достижения потокобезопасности. Потому как в Vector, в отличии от других реализаций List, все операции с данными являются синхронизированными. В качестве альтернативы часто применяется аналог — ArrayList.

Stack — данная коллекция является расширением коллекции Vector. Была добавлена в Java 1.0 как реализация стека LIFO (last-in-first-out). Является частично синхронизированной коллекцией (кроме метода добавления push()). После добавления в Java 1.6 интерфейса Deque, рекомендуется использовать именно реализации этого интерфейса, например ArrayDeque.

# Задание

Напишите программу в виде консольного приложения, которая моделирует карточную игру «пьяница» и определяет, кто выигрывает. В игре участвует 10 карт, имеющих значения от 0 до 9, большая карта побеждает меньшую; карта «0» побеждает карту «9».

Карточная игра “ В пьяницу”. В этой игре карточная колода раздается поровну двум игрокам. Далее они открывают по одной верхней карте, и тот, чья карта старше, забирает себе обе открытые карты, которые кладутся под низ его колоды. Тот, кто остается без карт, - проигрывает.

Для простоты будем считать, что все карты различны по номиналу и что самая младшая карта побеждает самую старшую карту (“шестерка берет туз”).

Игрок, который забирает себе карты, сначала кладет под низ своей колоды карту первого игрока, затем карту второго игрока (то есть карта второго игрока оказывается внизу колоды).

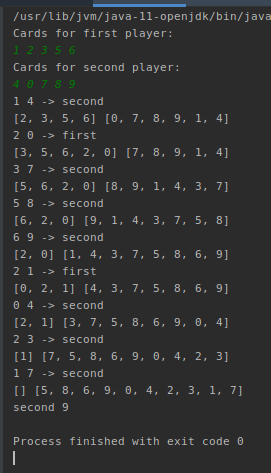
Используйте для организации хранения структуру данных Queue.

# Код

### Drunk.java

package dev.ky3he4ik.pr.pr7;  
  
import java.util.\*;  
  
// 2  
public class Drunk {  
 private Queue<Integer> firstPlayer;  
 private Queue<Integer> secondPlayer;  
 private String winner;  
 private int turns;  
  
 public Drunk() {  
 firstPlayer = new LinkedList<>();  
 boolean[] a = new boolean[10];  
 System.out.println("Cards for first player: ");  
 Scanner scanner = new Scanner(System.in);  
 for (int i = 0; i < 5; i++) {  
 int t = scanner.nextInt();  
 if (t < 0 || t >= 10 || a[t]) {  
 System.err.println("Invalid cards for first player!");  
 System.exit(1);  
 }  
 a[t] = true;  
 firstPlayer.add(t);  
 }  
 secondPlayer = new LinkedList<>();  
 System.out.println("Cards for second player: ");  
 for (int i = 0; i < 5; i++) {  
 int t = scanner.nextInt();  
 if (t < 0 || t >= 10 || a[t]) {  
 System.err.println("Invalid cards for second player!");  
 System.exit(1);  
 }  
 a[t] = true;  
 secondPlayer.add(t);  
 }  
 turns = 0;  
 }  
  
 public void play() {  
 while (turns <= 106 && !firstPlayer.isEmpty() && !secondPlayer.isEmpty()) {  
 int p1card = firstPlayer.poll();  
 int p2card = secondPlayer.poll();  
 boolean p1won;  
 if (p1card > p2card)  
 p1won = !(p1card == 9 && p2card == 0);  
 else  
 p1won = p1card == 0 && p2card == 9;  
 if (p1won) {  
 firstPlayer.add(p1card);  
 firstPlayer.add(p2card);  
 } else {  
 secondPlayer.add(p1card);  
 secondPlayer.add(p2card);  
 }  
  
 System.out.println("" + p1card + ' ' + p2card + " -> " + (p1won ? "first" : "second"));  
 System.out.println(firstPlayer.toString() + " " + secondPlayer.toString());  
  
 turns++;  
 }  
 if (firstPlayer.isEmpty())  
 winner = "second";  
 else  
 winner = "first";  
 }  
  
 public String getWinner() {  
 return winner;  
 }  
  
 public int getTurns() {  
 return turns;  
 }  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 Drunk drunk = new Drunk();  
 drunk.play();  
 if (drunk.getTurns() > 106)  
 System.out.println("botva");  
 else  
 System.out.println(drunk.getWinner() + ' ' + drunk.getTurns());  
 }  
}

# Скриншот



# Заключение

В данной практической работе я освоил на практике приемы работы со стандартными контейнерными классами Java Collection Framework.

# Библиографический список

1. Зорина Н.В. Курс лекций по Объектно-ориентированному программированию на Java, МИРЭА, Москва, 2016
2. Программирование на языке Java: работа со строками и массивами. Методические указания. [Электронный ресурс] : Учебно-методические пособия — Электрон. дан. — СПб. : ПГУПС, 2015. — 24 с.
3. Кожомбердиева, Г.И. Программирование на языке Java: создание графического интерфейса пользователя: учеб. пособие. [Электронный ресурс] : Учебные пособия / Г.И. Кожомбердиева, М.И. Гарина. — Электрон. дан. — СПб.: ПГУПС, 2012. — 67 с.
4. Вишневская, Т.И. Технология программирования. Часть 1. [Электронный ресурс] / Т.И. Вишневская, Т.Н. Романова. — Электрон. дан. — М. : МГТУ им. Н.Э. Баумана, 2007. — 59 с.