МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения

ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
по дисциплине
«Программирование на языке Java»

Тема: Коллекции, очереди, списки в Java.

Выполнил студент группы ИКБО-16-20 Пак С.А.

Принял ассистент кафедры ИиППО Русляков А.А.

Практические работы выполнены «_____» 2021г.

«Зачтено» «_____» 2021г

Москва 2021

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ

1. Коллекции

Для хранения наборов данных в Java предназначены массивы. Однако их не всегда удобно использовать, прежде всего потому, что они имеют фиксированную длину. Эту проблему в Java решают коллекции. Однако суть не только в гибких по размеру наборах объектов, но в и том, что классы коллекций реализуют различные алгоритмы и структуры данных, например, такие как стек, очередь, дерево и ряд других.

Классы коллекций располагаются в пакете java.util, поэтому перед применением коллекций следует подключить данный пакет.

Хотя в Java существует множество коллекций, но все они образуют стройную и логичную систему. Во-первых, в основе всех коллекций лежит применение того или иного интерфейса, который определяет базовый функционал.

2. Класс ArrayList

Класс ArrayList представляет обобщенную коллекцию, которая наследует свою функциональность от класса AbstractList и применяет интерфейс List. Проще говоря, ArrayList представляет простой список, аналогичный массиву, за тем исключением, что количество элементов в нем не фиксировано.

Класс имеет следующие конструкторы:

- ArrayList(): создает пустой список;
- ArrayList(Collection<? extends E> col): создает список, в который добавляются все элементы коллекции col;
- ArrayList (int capacity): создает список, который имеет начальную емкость capacity.

Емкость в ArrayList представляет размер массива, который будет использоваться для хранения объектов. При добавлении элементов фактически происходит перераспределение памяти - создание нового массива и копирование в него элементов из старого массива. Изначальное задание емкости ArrayList позволяет снизить подобные перераспределения памяти, тем самым повышая производительность.

3. Класс LinkedList

Обобщенный класс LinkedList<E> представляет структуру данных в виде связанного списка. Он наследуется от класса AbstractSequentialList и реализует интерфейсы List, Dequeue и Queue. Класс LinkedList имеет следующие конструкторы:

- LinkedList(): создает пустой список;

- LinkedList(Collection<? extends E> col): создает список, в который добавляет все элементы коллекции col.

РЕШЕНИЕ ЗАДАЧИ

1. Постановка задачи

Вариант: 3

Задание: создать свою коллекцию, такую же как и ArrayList.

2. Программный код

Файл DynamicArray.java:

```
package ru.mirea.classes;

public class DynamicArray {
  private int size;
  private int[] array;

  /**
   * Меняет длину массива
   * @param size      новая длина массива
   */
  private void resize(int size) {
    int[] nArray = new int[size];

    System.arraycopy(this.array, 0, nArray, 0, this.array.length);
    this.array = nArray;
  }

  /**
   * Конструктор по умолчанию
   */
  public DynamicArray() {
    this.size = 1;
  }

  /**
   * Создаёт массив на определённое количество элементов
   * @param size        количество элементов в массиве
   */
  public DynamicArray(int size) {
    this.array = new int[size];
  }

  /**
   * Добавляет элемент в конец массива
   * @param element           добавляемый элемент
   */
  public void add(int element) {
```

```
    if ((++this.size) >= this.array.length) {
      this.resize(this.array.length * 2);
              System.out.println("Изменение длины массива: " +
this.array.length);
    }
    this.array[this.size - 1] = element;
  }

  /**
   * Возвращает элемент в массиве под определённым индексом
   * @param index         индекс элемента
   * @return              элемент массива
   * @throws Exception    неверный индекс для массива
   */
  public int get(int index) throws Exception {
    if (index >= array.length) {
       throw new Exception("[ERROR]: Индекса " + index + " нет в
массиве длины " + array.length);
    }
    return this.array[index];
  }

  /**
   * Определяет значение для элемента на заданной позиции
   * @param index             позиция элемента
   * @param element           значение элемента
   * @throws Exception        неверный индекс массива
   */
  public void set(int index, int element) throws Exception {
    if (index >= array.length) {
       throw new Exception("[ERROR]: Индекса " + index + " нет в
массиве длины " + array.length);
    }
    this.array[index] = element;
  }

  /**
   * Возвращает размер массива
   * @return        количество элементов в массиве
   */
  public int size() {
    return this.size;
  }
}
```

Файл App.java:

```
package ru.mirea;
```

```
import java.util.Scanner;
import ru.mirea.classes.DynamicArray;

public class App {
  private static final Scanner IN = new Scanner(System.in);

  public static void main(String[] args) throws Exception {
    System.out.print("Введите количество элементов массива: ");
    int size = IN.nextInt();

    DynamicArray array = new DynamicArray(size);

    System.out.print("Введите элементы массива: ");
    for (int i = 0; i < size; ++i) {
      array.add(IN.nextInt());
    }

    System.out.println();
    System.out.print("Массив: ");

    System.out.print(array.get(0));
    for (int i = 1; i < size; ++i) {
      System.out.print(" " + array.get(i));
    }
    System.out.println();

    System.out.println("Размер массива: " + array.size());

    int value = 55;
    array.add(value);
    array.add(value);

    System.out.println();
    System.out.print("Массив: ");

    System.out.print(array.get(0));
    for (int i = 1; i < array.size(); ++i) {
      System.out.print(" " + array.get(i));
    }
    System.out.println();

    System.out.println("Размер массива: " + array.size());

    array.set(0, 645);

    System.out.println();
    System.out.print("Массив: ");
```

```
    System.out.print(array.get(0));
    for (int i = 1; i < array.size(); ++i) {
      System.out.print(" " + array.get(i));
    }
    System.out.println();

    System.out.println("Размер массива: " + array.size());
  }
}
```

3. Вывод программы

Пользователь вводит количество элементов в массиве. Затем длина массива изменяется на 6 добавляются два числа 55. И в конце первый элемент массива изменяется на 645. Как видно истинный размер массива 5, а не 6 (рис.1).

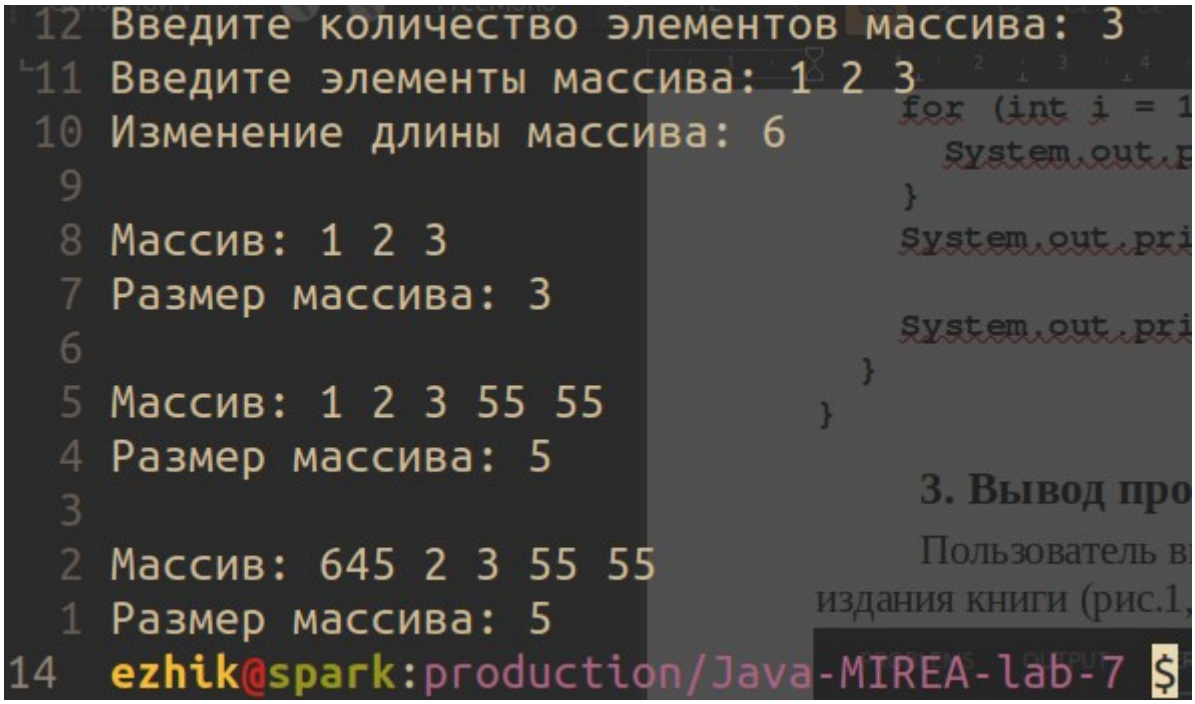

Рис.1 Выходные данные программы ч.1

ВЫВОД

В ходе выполнения изучил работу с различными коллекциями в Java.