Звіт

Автор: Лунгол О., КІТ101.8б

Дата: 08.11.2019

Лабораторна робота №5

РОЗРОБКА ВЛАСНИХ КОНТЕЙНЕРІВ. ІТЕРАТОРИ

**Мета.**

* Набуття навичок розробки власних контейнерів.
* Використання ітераторів.

**Вимоги:**

1. Розробити клас-контейнер, що [ітерується](http://docs.oracle.com/javase/8/docs/api/java/lang/Iterable.html) для збереження початкових даних завдання [л.р. №3](https://oop-khpi.github.io/#task_03_app) у вигляді **масиву рядків** з можливістю додавання, видалення і зміни елементів.
2. В контейнері реалізувати та продемонструвати наступні методи:
   * String toString() повертає вміст контейнера у вигляді рядка;
   * void add(String string) додає вказаний елемент до кінця контейнеру;
   * void clear() видаляє всі елементи з контейнеру;
   * boolean remove(String string) видаляє перший випадок вказаного елемента з контейнера;
   * Object[] toArray() повертає масив, що містить всі елементи у контейнері;
   * int size() повертає кількість елементів у контейнері;
   * boolean contains(String string) повертає true, якщо контейнер містить вказаний елемент;
   * boolean containsAll(Container container) повертає true, якщо контейнер містить всі елементи з зазначеного у параметрах;
   * public Iterator<String> iterator() повертає ітератор відповідно до [Interface Iterable](http://docs.oracle.com/javase/8/docs/api/java/lang/Iterable.html).
3. В класі ітератора відповідно до [Interface Iterator](http://docs.oracle.com/javase/8/docs/api/java/util/Iterator.html) реалізувати методи:
   * public boolean hasNext();
   * public String next();
   * public void remove().
4. Продемонструвати роботу ітератора за допомогою циклів while и for each.
5. Забороняється використання контейнерів (колекцій) і алгоритмів з [Java Collections Framework](https://docs.oracle.com/javase/8/docs/technotes/guides/collections/).

ЗАВДАННЯ ДО РОБОТИ

Ввести текст. Визначити та вивести, яких літер (голосних чи приголосних) більше в кожному реченні тексту. Результат вивести у вигляді таблиці.

ОПИС ПРОГРАМИ

**Опис змінних**

Scanner scan = **new** Scanner(System.***in***);//змінна для сканування введеної інформації

String str1 = **new** String();// створення нового порожнього рядка

**int** index1//індекс знаку «.»

**int** v = 0//змінна для голосних літер

**int** sign = 0;//змінна для знаків

Container container = **new** Container();//контейнер

**Ієрархія та структура класів**

**class** Lab04 – точка входу в програму

**class** Container – клас в якому знаходяться контейнері та клас з літераторами

**class** Function – клас з нашими основними функціями

**class** IteratorMine – вкладений клас

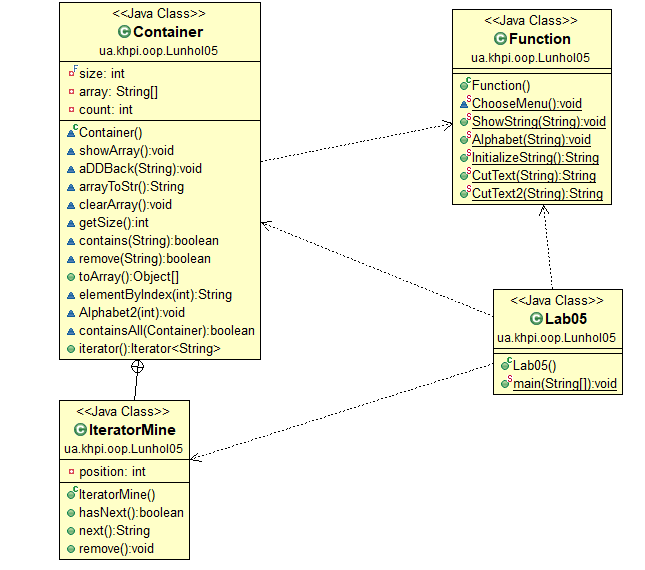


Рисунок 1 – Ієрархія класів

ТЕКСТ ПРОГРАМИ

Lab05.java

package ua.khpi.oop.Lunhol05;

import java.util.Arrays;

import java.util.Scanner;

import ua.khpi.oop.Lunhol05.Container.IteratorMine;

import static ua.khpi.oop.Lunhol04.Function.\*;

public class Lab05 {

/\*\*

\* @author ALEXANDRA

\* @version 4.0

\* Here start point of the program

\* Function in which menu actions are performed

\* until the user presses exit program does not end

\*

\* \*/

public static void main(String[] args) {

Container container = new Container();

String str1;

str1 = Function.InitializeString();

Function.ShowString(str1);

String str2 = "Hello world. Its me.";

System.out.println();

container.addBack(str1);

container.addBack(str2);

container.addBack(str1);

container.addBack(str2);

container.addBack(str1);

container.addBack(str2);

System.out.println("\n\n\n");

container.showArray();

System.out.println("\n\n\n");

container.Alphabet2(3);

System.out.println(container.arrayToStr());

System.out.println("\nWrite with iterator: ");

IteratorMine iter = (IteratorMine) container.iterator();//перемещаться по элементам

for (String s : container) {

System.out.println(s);

}

iter.next();

iter.remove();

System.out.println();

container.showArray();

}

}

Container.java

package ua.khpi.oop.Lunhol05;

import java.util.Arrays;

import java.util.Iterator;

import java.util.NoSuchElementException;

/\*\*

\* Contain elements in program.

\*/

class Container implements Iterable<String> {

/\*\*

\* First size for array.

\*/

private final int size = 10;

/\*\*

\* Array contains all data.

\*/

private String[] array = new String[size];

/\*\*

\* Counter of number elements.

\*/

private int count = 0;

/\*\*

\* Showing array`s data.

\*/

void showArray() {

if (count == 0) {

System.out.println("Empty mass");

} else {

System.out.println();

for (int i = 0; i < count; i++) {

System.out.println(array[i]);

}

}

}

/\*\*

\* First size for array.

\* @param str1

\*/

void addBack(final String str1) {

if (count == array.length) {

array = Arrays.copyOf(array, array.length \* 2);

array[count++] = str1;

} else {

array[count++] = str1;

}

}

/\*\*

\* function translates an array to a string

\* @param str1 our future string

\* \*\*/

String arrayToStr() {

StringBuilder str1 = new StringBuilder("");

if (count != 0) {

str1 = new StringBuilder(array[0]);

str1.append(" ");

for (int i = 1; i < count; i++) {

str1.append(array[i]);

str1.append(" ");

}

}

return str1.toString();

}

void clearArray() {

array = null;

count = 0;

}

int getSize() {

return count;

}

/\*\*

\* the function checks if the element that is being passed is contained in the container

\* \*\*/

boolean contains(final String str) {

boolean cont = false;

for (int i = 0; i < count; i++) {

if (cont) {

return cont;

} else {

String str1;

str1 = array[i];

cont = str.equals(str1);

if (i == count - 1) {

return cont;

}

}

}

return cont;

}

/\*\*

\* removal of any element from the container

\* \*\*/

boolean remove(final String str) {

boolean remov = false;

int coun = 0;

for (int i = 0; i < count; i++) {

if (remov) {

break;

} else {

remov = str.equals(array[i]);

coun++;

if (i == count - 1) {

break;

}

}

}

array[coun - 1] = null;

for (int i = 0; i < count; i++) {

array[coun - 1] = array[coun++];

}

count--;

return remov;

}

public Object[] toArray() {

if (array == null) {

return null;

}

return Arrays.copyOf(array, count);

}

String elementByIndex(final int index) {

return array[index];

}

/\*

\* function alphabet counting vowel and other

\* @param v our vowels

\* \*/

void Alphabet2(final int index) {

String str2 = new String();

String str1 = new String();

str1 = Function.CutText(array[index - 1]);

str2 = Function.CutText2(array[index - 1]);

System.out.println("==============================");

Function.Alphabet(str2);

System.out.println("==============================");

Function.Alphabet(str1);

}

boolean containsAll(final Container container) {

boolean result;

if (container.getSize() != count) {

return false;

}

for (int i = 0; i < count; i++) {

result = array[i].equals(container.elementByIndex(i));

if (!result) {

return false;

}

}

return true;

}

@Override

public Iterator<String> iterator() {

return new IteratorMine();

}

public class IteratorMine implements Iterator<String> {

/\*\*

\* вложеный клас

\* унаследованый от интерфейса Iterator<String>

\*/

private int position = 0;

@Override

public boolean hasNext() {

return position < count;

}

@Override

public String next() {

if (this.hasNext()) {

return array[position++];

} else {

throw new NoSuchElementException();

}

}

@Override

public void remove() {

int temp = position;

for (int i = position; i < count; i++) {

array[temp++] = array[i + 1];

}

count--;

}

}

}

Function.java

**package** ua.khpi.oop.Lunhol05;

**import** java.util.Scanner;

/\*

\* class Function in which all used functions are stored

\*

\* @author ALEXANDRA

\* @version 4.2

\* Here start point of the program

\* \*/

**public** **class** Function {

/\*

\* Function that show user point of menu that he can choose

\* \*/

**static** **void** ChooseMenu() {

System.***out***.println("\n\nChoose the right variant:");

System.***out***.println("1. Create string");

System.***out***.println("2. Show string");

System.***out***.println("3. Split text");

System.***out***.println("4. Counting now");

System.***out***.println("0. The End");

System.***out***.print("Write:");

}

/\*

\* Function show our string that we wrote

\* @param str is our string

\* \*/

**public** **static** **void** ShowString(String str){

System.***out***.println("Your string:");

System.***out***.println(str);

}

/\*

\* function alphabet counting vowel and other

\* @param v our vowels

\* \*/

**public** **static** **void** Alphabet (String str) {

**int** v = 0;

**int** sign = 0;

**char**[] vowels = {'a','e','i','o','u'};

**char** [] sentence = str.toLowerCase().toCharArray();

**for** (**char** letter : sentence) {

**if**(letter == ' ' || letter == '.' || letter == ',')

sign++;

**for** (**char** vowel : vowels) {

**if** (letter == vowel) {

v++;

}

}

}

System.***out***.println(str);

System.***out***.print("\tVowels:"+ v);

System.***out***.print("\tConsonants:" + (sentence.length -v-sign));

System.***out***.println();

System.***out***.println();

}

/\*

\* Function reads input string

\* @return str1 as our used string

\* \*/

**public** **static** String InitializeString(){

Scanner scan = **new** Scanner(System.***in***);

System.***out***.print("Enter text whith dots: ");

String str1 = **new** String();

str1 = scan.nextLine();

**return** str1;

}

/\*

\* Function truncates a string to a point and show the first sentence

\* @param index1 looking for dot

\* @return str

\* \*/

**public** **static** String CutText (String str) {

**int** index1=str.indexOf('.');

str = str.substring(index1+1);

**return** str;

}

/\*

\* Function truncates a string to a point and show the second sentence

\* @param index1 looking for dot

\* @return str

\* \*/

**public** **static** String CutText2 (String str) {

**int** index1=str.indexOf('.');

str = str.substring(0, index1+1);

**return** str;

}

}

РОБОТА ПРОГРАМИ

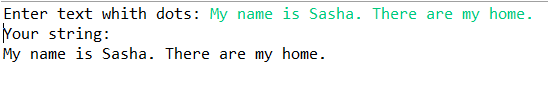


Рисунок 1 – Введення нового рядка

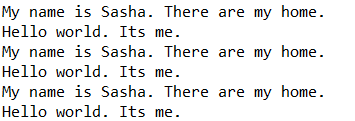


Рисунок 2 – Занесення в контейнер

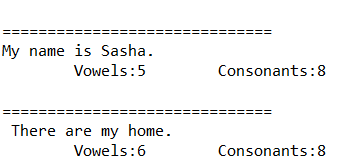


Рисунок 4 – Робота функції з контейнером



Рисунок 5 – З масиву в рядок

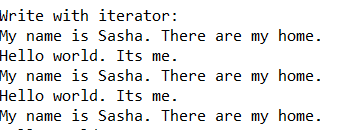


Рисунок 6 – Виведення за допомогою ітератора

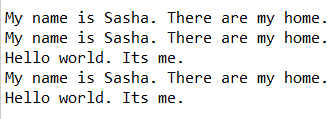


Рисунок 7 – Видалення вибраного елементу

ВИСНОВОК

В даній лабораторній роботі ми навчились створювати контейнери , та використовувати ітератори.