Національний технічний університет

«Харківський політехнічний інститут»

**Лабораторна робота № 5**

з дисципліни «ООП»

Виконав студент групи КІТ - 320

Радєвич Владислав Романович

Перевірив викладач

Жилин Володимир Анатолійович

Харків 2021

**Розробка власних контейнерів. Ітератори**

Мета: набуття навичок розробки власних контейнерів та використання ітераторів

1 ВИМОГИ

* 1. **Розробник**

Інформація про розробника:

Радєвич Владислав Романович;

студент групи КІТ – 320;

номер варіанту.

* 1. **Загальне завдання**

Розробити клас-контейнер, що ітерується для збереження початкових даних завдання л.р. №3 у вигляді масиву рядків з можливістю додавання, видалення і зміни елементів.

* 1. **Задача**

1. 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.

1. В класі ітератора відповідно до Interface Iterator реалізувати методи:

* public boolean hasNext();
* public String next();
* public void remove().

1. Продемонструвати роботу ітератора за допомогою циклів while и for each
2. Забороняється використання контейнерів (колекцій) і алгоритмів з Java Collections Framework
3. ОПИС ПРОГРАМИ
   1. **Засоби ООП**

Програма призначена для збереження початкових даних завдання л.р. №3 у вигляді масиву рядків з можливістю додавання, видалення і зміни елементів.

* 1. **Ієрархія та структура класів**

public final class Main – головний клас

class Text – клас для вирішення завдання лаб.роб №3

public class Collection implements Iterable<String> - клас контейнер в якому реалізовані всі необхідні методи з завдання

public Iterator<String> iterator() – клас ітератор

* 1. **Важливі фрагменти програми**

Файл Main.java

package ua.khpi.oop.radeivych05;

import java.util.Scanner;

import java.util.Iterator;

public final class Main {

public static void main(String[] args) {

String tmp = "";

Text arg = new Text();

Collection collection = new Collection();

collection.add("abc,ad");

collection.add("cba,ad 2");

// System.out.println(collection);

for(var str : collection)

{

System.out.println(str);

}

Iterator<String> it = collection.iterator();

// while(it.hasNext()) {

// System.out.println(it.next());

// }

it.remove();

System.out.println(collection);

// Iterator<String> it = collection.iterator();

// while(it.hasNext()) {

// System.out.println(it.next());

// }

}

}

class Text {

String text;

String output1;

String output2;

String output3;

String outputFinal;

public void ViewBefore() {

System.out.printf("–С—Г–ї–Њ: \n%s\n---------\n", this.text);

}

public void DeleteNumbers() {

output1 = text;

StringBuffer strArgs = new StringBuffer(output1);

strArgs.ensureCapacity(32);

int length = output1.length();

for (int i = 0; i < length - 1; i++) {

while(strArgs.charAt(i) == '0' || strArgs.charAt(i) == '1' || strArgs.charAt(i) == '2'

|| strArgs.charAt(i) == '3' || strArgs.charAt(i) == '4' || strArgs.charAt(i) == '5'

|| strArgs.charAt(i) == '6' || strArgs.charAt(i) == '7' || strArgs.charAt(i) == '8'

|| strArgs.charAt(i) == '9' && strArgs.charAt(i) != ' ') {

strArgs.deleteCharAt(i);

length--;

}

}

output1 = strArgs.toString();

}

public void AddSpace() {

output2 = output1;

StringBuffer strArgs = new StringBuffer(output2);

strArgs.ensureCapacity(32);

int length = output2.length();

/\*

\* –Ф–Њ–±–∞–≤–ї–µ–љ–љ—П –њ—А–Њ–±i–ї—Г

\*/

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

if (strArgs.charAt(i) != ' ' && i != 0 && strArgs.charAt(i - 1) != ' ' && i != strArgs.length()) {

strArgs.insert(i + 1, ' ');

length++;

}

}

output2 = strArgs.toString();

}

public void DeleteSpace() {

output3 = output2;

StringBuffer strArgs = new StringBuffer(output3);

strArgs.ensureCapacity(32);

int length = output3.length();

/\*

\* –Т–Є–і–∞–ї–µ–љ–љ—П –њ—А–Њ–±i–ї—Г

\*/

for (int i = 0; i < length - 1; i++) {

if (i != 0 && i < strArgs.length() && strArgs.charAt(i) == ' ' && strArgs.charAt(i + 1) == ' ') {

while (strArgs.charAt(i) == ' ' && strArgs.charAt(i + 1) == ' ') {

strArgs.deleteCharAt(i + 1);

length--;

}

}

}

output3 = strArgs.toString();

outputFinal = output3;

}

public void ViewResultDelNum() {

System.out.printf("–°—В–∞–ї–Њ –њ—Ц—Б–ї—П –≤–Є–і–∞–ї–µ–љ–љ—П —Г—Б—Ц—Е —З–Є—Б–µ–ї: %s \n", output1);

}

public void ViewResultAdSp() {

System.out.printf("–°—В–∞–ї–Њ –њ—Ц—Б–ї—П –і–Њ–і–∞–≤–∞–љ–љ—П –±—А–∞–Ї—Г—О—З–Є—Е –њ—А–Њ–±—Ц–ї—Ц–≤: %s \n", output2);

}

public void ViewResultDelSp() {

System.out.printf("–°—В–∞–ї–Њ –њ—Ц—Б–ї—П –≤–Є–і–∞–ї–µ–љ–љ—П –Ј–∞–є–≤–Є—Е –њ—А–Њ–±—Ц–ї—Ц–≤: %s \n", output3);

}

public void ViewAfter() {

System.out.printf("–Ч–∞–≥–∞–ї—М–љ—Ц–є —А–µ–Ј—Г–ї—М—В–∞—В: %s \n\n", outputFinal);

}

}

Файл Collection.java

package ua.khpi.oop.radeivych05;

import java.util.Iterator;

public class Collection implements Iterable<String> {

private String [] mass = new String[10];

private int Size = 0;

private int ActualSize = 10;

public void setMass(String[] mass) {

this.mass = mass;

this.Size = mass.length;

this.ActualSize= mass.length;

}

public String[] getMass() {

var masst = new String[Size];

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

masst[i] = mass[i];

}

return masst;

}

public String toString() {

StringBuilder strB = new StringBuilder();

for(var str : this)

strB.append(str+" ");

if(!(strB.length()==0))

strB.deleteCharAt(strB.length()-1);

return strB.toString();

}

public void add(String string) {

if(Size==ActualSize) {

String [] tempMass = new String[ActualSize+10];

ActualSize += 10;

for(int i = 0; i<Size; i++)

tempMass[i]=mass[i];

mass=tempMass;

}

mass[Size++]= string;

}

public void clear() {

mass = new String[10];

Size = 0;

ActualSize = 10;

}

public boolean remove(String string) {

for(int i = 0; i<Size; i++)

if(mass[i].equals(string)) {

delete(i);

return true;

}

return false;

}

public Object[] toArray() {

return getMass();

}

public int size() {

return Size;

}

public boolean contains(String string) {

for(var str : this) {

if(str.equals(string))

return true;

}

return false;

}

public boolean containsAll(Collection container) {

for(var str : container)

if(!contains(str)) {

return false;

}

return true;//TODO

}

private void delete(int i) {

String [] tempMass = new String[ActualSize];

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

tempMass[j]=mass[j];

}

for(int j = i+1;j<Size;j++) {

tempMass[j]=mass[j];

}

mass = tempMass;

Size -= 1;

}

public String get(int i) {

if(i<Size)

return mass[i];

else

return null;

}

public Iterator<String> iterator() {

Iterator<String> MyIterator = new Iterator<String>(){

private int i = 0;

public String next() {

return mass[i++];

}

public boolean hasNext() {

return i<Size;

}

public void remove() {

delete(i-1);

}

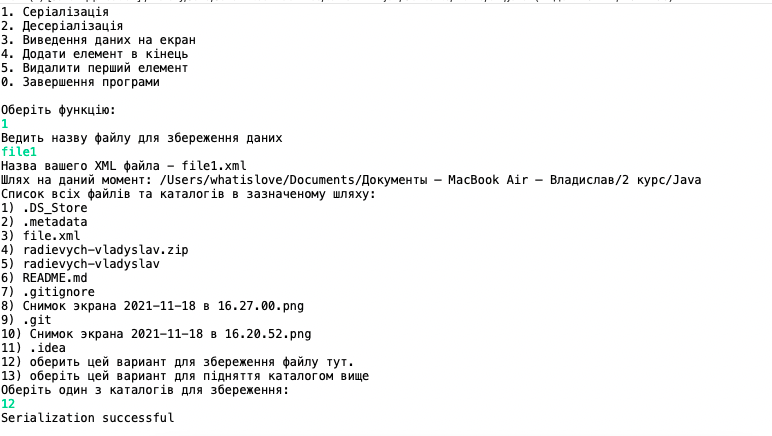
};

return MyIterator;

}

}

Демонстрація використання



ВИСНОВКИ

Протягом лабораторної роботи я навчився розроблювати власні контейнери та ітератори.