Розробка власних контейнерів. Ітератори

Мета роботи: набутти навичок розробки власних контейнерів. Використання ітераторів.

Вимоги

1. Розробити клас-контейнер, що ітерується для збереження початкових даних завдання л.р. №3 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.

3. В класі ітератора відповідно до Interface Iterator 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.

Розробник: Каркуша Дмитро Андрійович КІТ119а №10.

Опис програми

Засоби ООП: клас, метод классу, інтерфейс ітератору та його переписані методи.

Структура класів: два публічних класа Main, Container та їх статичні публічні методи.

Важливі фрагменти програми:

Методи контейнеру:

```
public class Container {
    private String[] array;
    private int size;

public String toString() // повертає вміст контейнера у вигляді рядка;
    {
    String newArray = "";
    for (String string : array)
        {
        newArray += string + " ";
        }
    return newArray;
    }

public void addElement(String string) //додає вказаний елемент до кінця контейнеру;
    {
    String newArray[] = new String[size + 1];
    for (int i = 0; i < size; i++)
```

```
{
newArray[i] = array[i];
}
newArray[size] = string;
size++;
array = newArray;
}
public void clear() //видаляє всі елементи з контейнеру;
{
for (int i = 0; i < array.length; i++) {
array[i]=null;

}
size =0;
}
public boolean removeElement(String string) // видаляє перший випадок вказаного елемента з контейнера;
{

boolean flag = false;
String [] new_array = new String[size-1];
for(int i=0;i<size;i++) {
if(array[i].equals(string))
flag = true;
}
if(flag) {
for(int i=0,j=0;i<size;i++) {
if(array[i].equals(string))
i++;
new_array[j]=array[i];
j++;
}
size--;
array = new_array;
return flag;
}
else
{
```

```
                return flag;
                }
            }
    public Object[] toArray()  //повертає масив, що містить всі елементи у контейнері;
            {
        return array;
            }
    public int size()  //повертає кількість елементів у контейнері;
            {
        return size;
            }
    public boolean containsAll(Container arr) //повертає true, якщо контейнер містить всі елементи з зазначеного у параметрах;
            {
            int count = 0;
        for (int i = 0; i < array.length; i++) {
            for (int j = 0; j < arr.array.length; j++) {
                if(arr.array[j].equals(array[i]))
                        count++;
                }
            }
        if(count == arr.array.length)
                return true;
            else
                return false;
            }
    public boolean contains(String str) //повертає true, якщо контейнер містить вказаний елемент;
            {
        boolean flag = false;
        for (int i=0;i<array.length;i++) {
                if(array[i].equals(str))
                        flag=true;
                }
            return flag;
            }
    public Container(String... str) {
                if(str.length!=0) {
                    size = str.length;
```

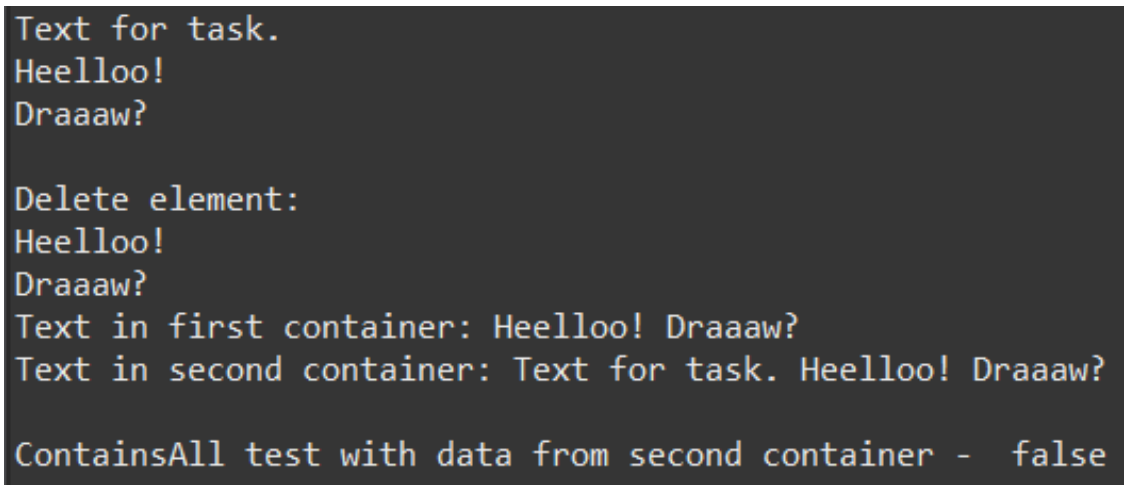
```
                array = new String[size];
                for (int i=0;i<size;i++) {
                    array[i]=str[i];
                }
            }
        }
    public Iterator<String> getIterator()
    {
        return new MyIterator<String>();
    }
    private class MyIterator<String> implements Iterator {
        int index;

        @Override
        public boolean hasNext() {
            return index < size ? true : false;
        }

        @Override
        public Object next() {
            return array[index++];
        }

/*Method that removes from the underlying collection the last element returned by this iterator*/
        @Override
        public void remove() {
            Container.this.removeElement(array[--index]);
        }
    }
}
```

Результат роботи програми.



Висновки

Набув навичок розробки власних контейнерів. Навчився використовувати ітератори у програмі.