

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТУ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Інститут комп’ютерних наук та інформаційних технологій
Кафедра систем штучного інтелекту

Лабораторна робота №2

з дисципліни

«Об’єктно-орієнтоване програмування»

Виконала:

студентка групи КН-108

Семич Тамара

Прийняла:

Грабовська Н.Р.

Львів – 2019 р.

Зміст

1. Тема лабораторної роботи.
2. Мета роботи.
3. Вимоги.
4. Висновок.

Тема : Розробка власних контейнерів. Ітератори. Серіалізація/десеріалізація об'єктів. Бібліотека класів користувача.

Мета:

- Набуття навичок розробки власних контейнерів.
- Використання ітераторів
- Тривале зберігання та відновлення стану об'єктів.
- Ознайомлення з принципами серіалізації/десеріалізації об'єктів.
- Використання бібліотек класів користувача.

Вимоги:

1. Розробити клас-контейнер, що ітерується для збереження початкових даних Вашого варіанту завдання з попередньої роботи (Прикладні задачі. Список з 1-15 варіантів) 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 .
6. Реалізувати і продемонструвати тривале зберігання/відновлення розробленого контейнера за допомогою серіалізації/десеріалізації .
7. Обмінятися відкомпільованим (без початкового коду) службовим класом (Utility Class) рішення одного варіанту задачі .
8. Продемонструвати послідовну та вибірккову обробку елементів розробленого контейнера за допомогою власного і отриманого за обміном службового класу.
9. Реалізувати та продемонструвати порівняння, сортування та пошук елементів у контейнері.
10. Розробити консольну програму та забезпечити діалоговий режим роботи з користувачем для демонстрації та тестування рішення.

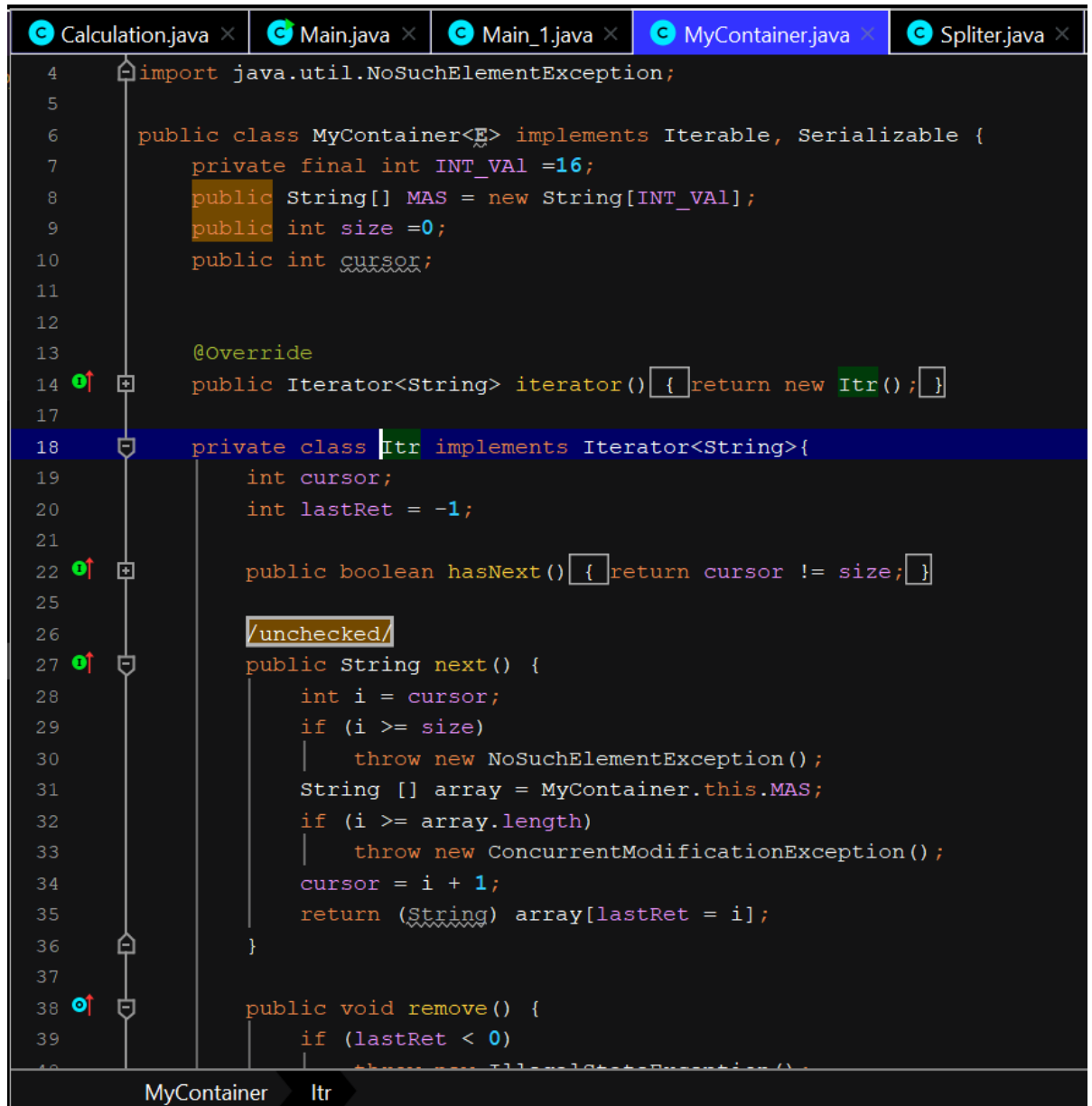
Розробник: Семич Тамара, КН-108, номер варіанту індивідуального завдання- 9.

Ієрархія та структура об'єктів:

1. Клас Main, який містить функцію main.
2. Клас Main_1, який містить функцію main_1, що зв'язує цю лабораторну роботу з попередньою .
3. Клас Spliter, який містить функцію spliter, що забезпечує поділ тесту на слова (альтернатива методу split).
4. Клас Calculation, який рахує скільки раз повторюється кожне слово.
5. Клас MyContainer, який містить всі функції з вимог і внутрішній клас Itr.

Важливі фрагменти коду:

Клас MyContainer:



```
4 import java.util.NoSuchElementException;
5
6 public class MyContainer<E> implements Iterable, Serializable {
7     private final int INT_VAL = 16;
8     public String[] MAS = new String[INT_VAL];
9     public int size = 0;
10    public int cursor;
11
12
13    @Override
14    public Iterator<String> iterator() { return new Itr(); }
15
16    private class Itr implements Iterator<String>{
17        int cursor;
18        int lastRet = -1;
19
20        public boolean hasNext() { return cursor != size; }
21
22        /unchecked/
23        public String next() {
24            int i = cursor;
25            if (i >= size)
26                throw new NoSuchElementException();
27            String [] array = MyContainer.this.MAS;
28            if (i >= array.length)
29                throw new ConcurrentModificationException();
30            cursor = i + 1;
31            return (String) array[lastRet = i];
32        }
33
34        public void remove() {
35            if (lastRet < 0)
36                throw new IllegalStateException();
37        }
38    }
39}
```

```
56 // Метод для повернення елемента за індексом
57 public String set(int index) {
58     checkIndex(index);
59     return MAS[index];
60 }
61
62 //Метод який перевіряє чи входить цей індекс в наш масив
63 private void checkIndex(int index) {
64     if (index >= size || index < 0)
65         throw new IllegalArgumentException();
66 }
67
68 //Додаємо елемент за індексом
69 public void add(int index, String value) {
70     if (index > MAS.length) {
71         addSize();
72     }
73     MAS[index] = value;
74     if (index > size) {
75         size++;
76     }
77 }
78 //Додаємо елемент до кінця списку
79 public void add(String string) {
80     if (size > MAS.length) {
81         addSize();
82     }
83     MAS[size] = string;
84     size++;
85 }
86
87 private void addSize() {
88     String[] MAS_1 = new String[MAS.length + 1];
89     for (int i = 0; i < MAS.length; i++) {
90         MAS_1[i] = MAS[i];
91     }
92     MAS = MAS_1;
93 }
```

MyContainer > ltr

Висновок: на цій лабораторній роботі я вивчила основні принципи роботи з контейнером. Навчилася створювати його та використовувати.