

Розробка власних контейнерів. Ітератори

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

Вимоги

- Розробити клас-контейнер, що ітерується для збереження початкових даних завдання [HYPERLINK](https://oop-khpi.gitlab.io/%D0%B7%D0%BD%D0%BC%D0%BA%D0%BD%D0%BD%D1%8F/task03/) "https://oop-khpi.gitlab.io/%D0%B7%D0%BD%D0%BC%D0%BA%D0%BD%D0%BD%D1%8F/task03/" л.р [HYPERLINK](https://oop-khpi.gitlab.io/%D0%B7%D0%BD%D0%BC%D0%BA%D0%BD%D0%BD%D1%8F/task03/) "https://oop-khpi.gitlab.io/%D0%B7%D0%BD%D0%BC%D0%BA%D0%BD%D0%BD%D1%8F/task03/". №3 у вигляді масиву **рядків** з можливістю додавання, видалення і зміни елементів.
 - 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.
 - В класі ітератора відповідно до Interface Iterator реалізувати методи:

- public boolean hasNext();
- public String next();
- public void remove().
- Продемонструвати роботу ітератора за допомогою циклів *while* и *for each*.
- Забороняється використання контейнерів (колекцій) і алгоритмів з [Java Collections Framework](#).

Розробник: Єлєсін Артем Олександрович

Група : KIT-1196

Опис програми

Засоби ООП: клас, метод классу.

Структура класів: один публічний клас Main, один утилітарний клас Helper, один клас колекція MyCollection

Важливі фрагменти програми:

```
static private MyCollection findArray(String ln){
    MyCollection line = new MyCollection();
    StringBuilder strB = new StringBuilder();
    for(int i = 0; i < ln.length();i++ ) {
        if(ln.charAt(i)==' ') {
            line.add(strB.toString());
            strB = new StringBuilder();
        }
        else
            strB.append(ln.charAt(i));
    }
    line.add(strB.toString());
```

```
    return line;
}

public class MyCollection 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(MyCollection 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;  
}
```

Результати роботи

```
Input line:  
    Проверка      про      вер ка  
Lines are greater than average:  
Проверка  
Line lenght:8  
Lines less than average:  
про  
Line lenght:3  
вер  
Line lenght:3  
ка  
Line lenght:2  
Первая  
Вторая  
Третья  
Четвертая  
Первая  
Вторая  
Третья  
Четвертая
```

Висновки

Оволодів навичками розробки власної колекції та ітератора.