1. Тема роботи

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.

1 ВИМОГИ

1.1 Розробник: Мокрик Ярослав Любомирович; КН-108; варіант 19 (завдання 4).

1.2

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 (Прикладні задачі.

Список з 1-15 варіантів)​з сусіднім номером.

1 міняється з 2, 2 з 3, 3 з 4, 4 з 5 і т.д. Останній, 15 міняється з 1 варіантом

і далі аналогічно.

8. Продемонструвати послідовну та вибіркову 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.

1.3 Задача

Ввести текст. У кожному слові тексту k-ту літеру замінити заданим символом. Якщо k більше довжини слова, видати повідомлення та заміну не виконувати. Початкові дані та результат вивести у вигляді таблиці.

2 ОПИС ПРОГРАМИ

TODO: Дописати

2.1 Засоби ООП

Я застосував 3 класи, кожен з яких містить статичні методи. Таким чином, їх можна викликати, не створюючи екземпляр класу. До того ж, я використовував клас StringBuilder, щоб створити рядок, який можна змінювати в ході роботи програми.

2.2 Ієрархія та структура класів

Клас Main - головний клас. Керує роботою всієї програми і викликає статичні методи інших класів.

Клас Menu - клас меню. Керує показом головного меню, приймає ввід користувача і викликає відповідні методи.

Клас StringMangager - клас менеджера рядків. Керує вводом рядка, числа і символа. Генерує новий рядок відповідно до вимог завдання.

2.3 Важливі фрагменти програми

Цей метод генерує рядок відповідно до вимог програми.

public static void computeResult() {

res = new StringBuilder();

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

if ((i+1) % k != 0 || i == 0)

res.append(str.charAt(i));

else

res.append(c);

if (Main.debug)

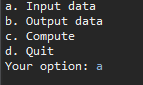
System.out.println(res);

}

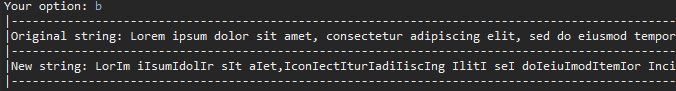
}

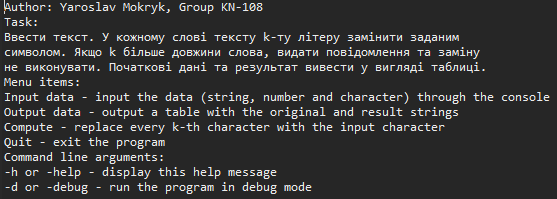
3 ВАРІАНТИ ВИКОРИСТАННЯ

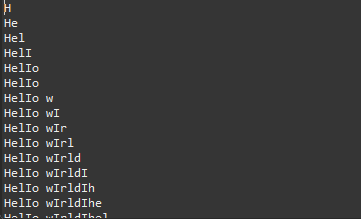
Головне меню:



Введення даних:  
Виведення в форматі таблиці:

Повідомлення аргументу -h (-help):

Обчислення результату в режимі -d (-debug):



ВИСНОВКИ Я навчився розробляти власні утилітарні класи. Я навчився виконувати прикладні завдання з використанням масивів та рядків. Я навчився реалізовувати діалогову програму в консолі Java.