Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

до виконаної лабораторної роботи № 1

з дисципліни

“Прикладне програмування”

на тему:

***«Основи Java»***

Виконав

студент групи *ОІ-11 сп*

*Вальчевський П. В.*

Прийняв:

*Пелех Ю. М.*

Львів – 2023

## Лабораторна робота № 1

*Тема роботи:* «Основи Java».

*Мета роботи:* ***…***

***Порядок виконання роботи***

1. **Послідовність роботи**
2. Згідно з індивідуальним завданням напишіть програму (з використанням об’єктно-орієнтованого підходу), що відповідає наступним вимогам:
   * Програма визначає і опрацьовує задані числа Фібоначчі (1, 1, 2, 3, 5, … ) або Люка (1, 3, 4, 7, 11, …). Номер числа та інші вхідні дані користувач має можливість передати через командний рядок, або в іншому випадку вводить з клавіатури.
   * Створіть клас, який містить дані про номер та значення числа Фібоначчі або Люка, методи доступу до даних та інші методи, відповідно до індивідуального завдання.
   * У головній функції оголосіть об’єкти або масив об’єктів створеного класу, які зберігають та опрацьовують введені дані.
   * Програма виводить на екран вхідні дані і результати обчислень та перевірок із відповідними підказками.
   * Скомпілюйте та запустіть програму через командний рядок та за допомогою обраного середовища розробки.
3. Продокументуйте програму з п. 1 з використанням JavaDoc і згенеруйте на основі нього документацію.
4. **Індивідуальне завдання**

Номер завдання вибирався за порядковим номером у журналі групи. Мій варіант № 3.

1. Визначити, які числа серед перших N чисел Фібоначчі є кубами.
2. **Текст програми**

Посилання на репозиторій в GitHub: <https://github.com/ManFromLviv/nulp_prikladne_programuvannia_lr1.git>

*Файл FibonachiAndCubes.java (клас реалізації завдання)*

package lr1;

import java.util.ArrayList;

/\*\*

\*

\* @author Valchevskyi

\*/

public class FibonachiAndCubes {

private final ArrayList<Integer> fibonachi;

private final ArrayList<Integer> cubes;

/\*

\* @param N is count number Fibonachi

\*/

public FibonachiAndCubes(int N) {

fibonachi = new ArrayList<>();

cubes = new ArrayList<>();

setFibonachi(N);

}

/\*\*

\* @param N is count number Fibonachi

\*/

public final void setFibonachi(int N) {

if (fibonachi.size() != N) {

if (N > 2) {

// For append next number if row clear or row is one element.

if (fibonachi.isEmpty()) {

fibonachi.add(1);

fibonachi.add(1);

cubes.add(1);

} else if (fibonachi.size() == 1) {

fibonachi.add(1);

}

// Delete or append elements in Fibonachi and cubes rows.

if (fibonachi.size() > N) {

for (int i = fibonachi.size() - 1; i >= N; i--) {

if (isCube(fibonachi.get(i))) {

cubes.remove(cubes.size() - 1);

}

fibonachi.remove(i);

}

} else {

try {

for (int i = fibonachi.size(); i < N; i++) {

fibonachi.add(Math.addExact(fibonachi.get(i - 2), fibonachi.get(i - 1)));

if (isCube(fibonachi.get(i))) {

cubes.add(fibonachi.get(i));

}

}

} catch (ArithmeticException e) {

// Overflow int data type.

}

}

} else {

fibonachi.clear();

cubes.clear();

if (N > 0) {

fibonachi.add(1);

cubes.add(1);

if (N == 2) {

fibonachi.add(1);

}

}

}

}

}

/\*\*

\* @param number check value is cube

\* @return boolean is cube

\*/

private boolean isCube(int number) {

return number == Math.pow((int) Math.pow(number, 1.0 / 3), 3);

}

/\*\*

\* @return String copy array list of Fibonachi and cubes

\*/

@Override

public String toString() {

StringBuilder stringBuilder = new StringBuilder("Result:");

stringBuilder.append(toStringBuilderArrayList(fibonachi, "\n\tFibonachi"));

stringBuilder.append(toStringBuilderArrayList(cubes, "\n\tCubes"));

return stringBuilder.toString();

}

/\*\*

\* @param arrayList for output data of array list

\* @param nameArrayList for output name array list

\* @return StringBuilder copy of array list

\*/

private StringBuilder toStringBuilderArrayList(ArrayList<Integer> arrayList, String nameArrayList) {

StringBuilder stringBuilder = new StringBuilder(nameArrayList);

stringBuilder.append(" => N: ").append(arrayList.size());

stringBuilder.append(" => List: ");

if (!arrayList.isEmpty()) {

for (int i = 0; i < arrayList.size() - 1; i++) {

stringBuilder.append(arrayList.get(i)).append(", ");

}

stringBuilder.append(arrayList.get(arrayList.size() - 1));

} else {

stringBuilder.append("none");

}

return stringBuilder;

}

}

*Файл LR1.java (клас з функцією виконання)*

package lr1;

/\*\*

\*

\* @author Valchevskyi

\*/

public class LR1 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

FibonachiAndCubes variantThree = new FibonachiAndCubes(47);

System.out.println("Overflow in 47 element Fibonachi (max element - 46):");

System.out.println(variantThree.toString());

System.out.println("\nChange count element in Fibonachi:");

variantThree.setFibonachi(4);

System.out.println(variantThree.toString());

variantThree.setFibonachi(2);

System.out.println(variantThree.toString());

variantThree.setFibonachi(0);

System.out.println(variantThree.toString());

variantThree.setFibonachi(1);

System.out.println(variantThree.toString());

variantThree.setFibonachi(0);

System.out.println(variantThree.toString());

variantThree.setFibonachi(3);

System.out.println(variantThree.toString());

variantThree.setFibonachi(1);

System.out.println(variantThree.toString());

variantThree.setFibonachi(6);

System.out.println(variantThree.toString());

variantThree.setFibonachi(6);

System.out.println(variantThree.toString());

variantThree.setFibonachi(10);

System.out.println(variantThree.toString());

}

}

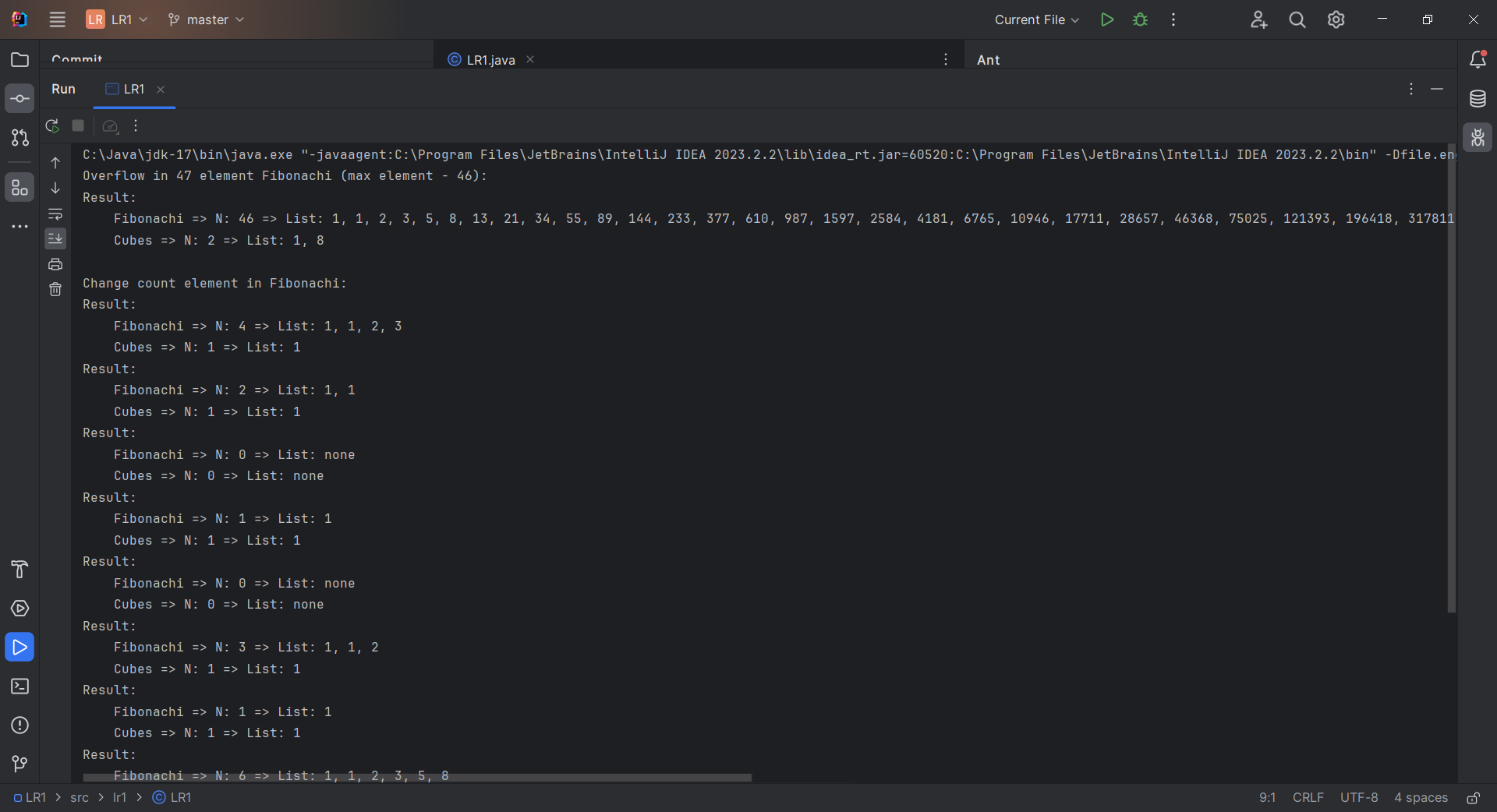
1. **Контрольний приклад та результати роботи програми**
2. 

Рисунок Результат програми виконання

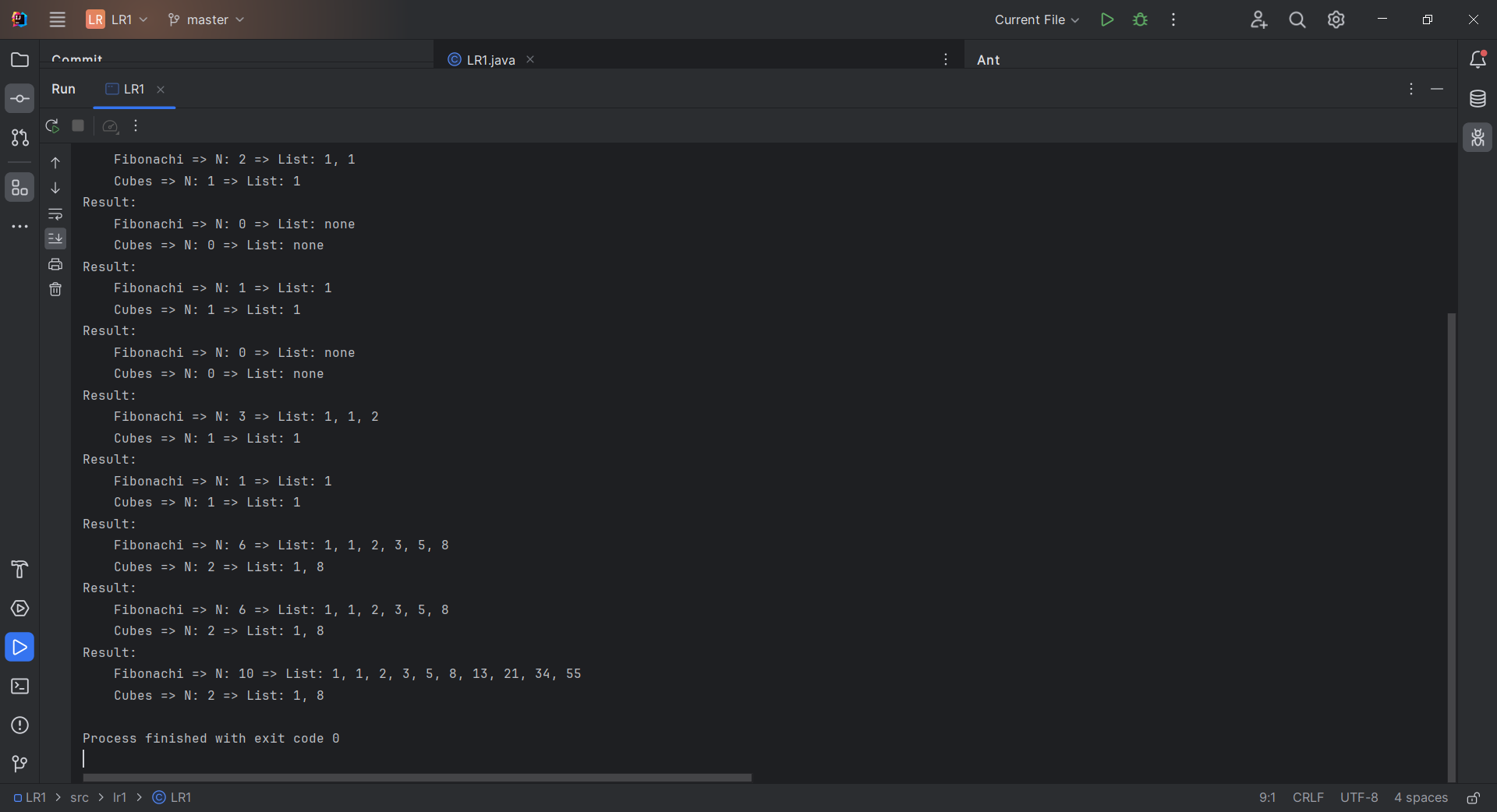


Рисунок 2 Результат програми виконання (продовження)

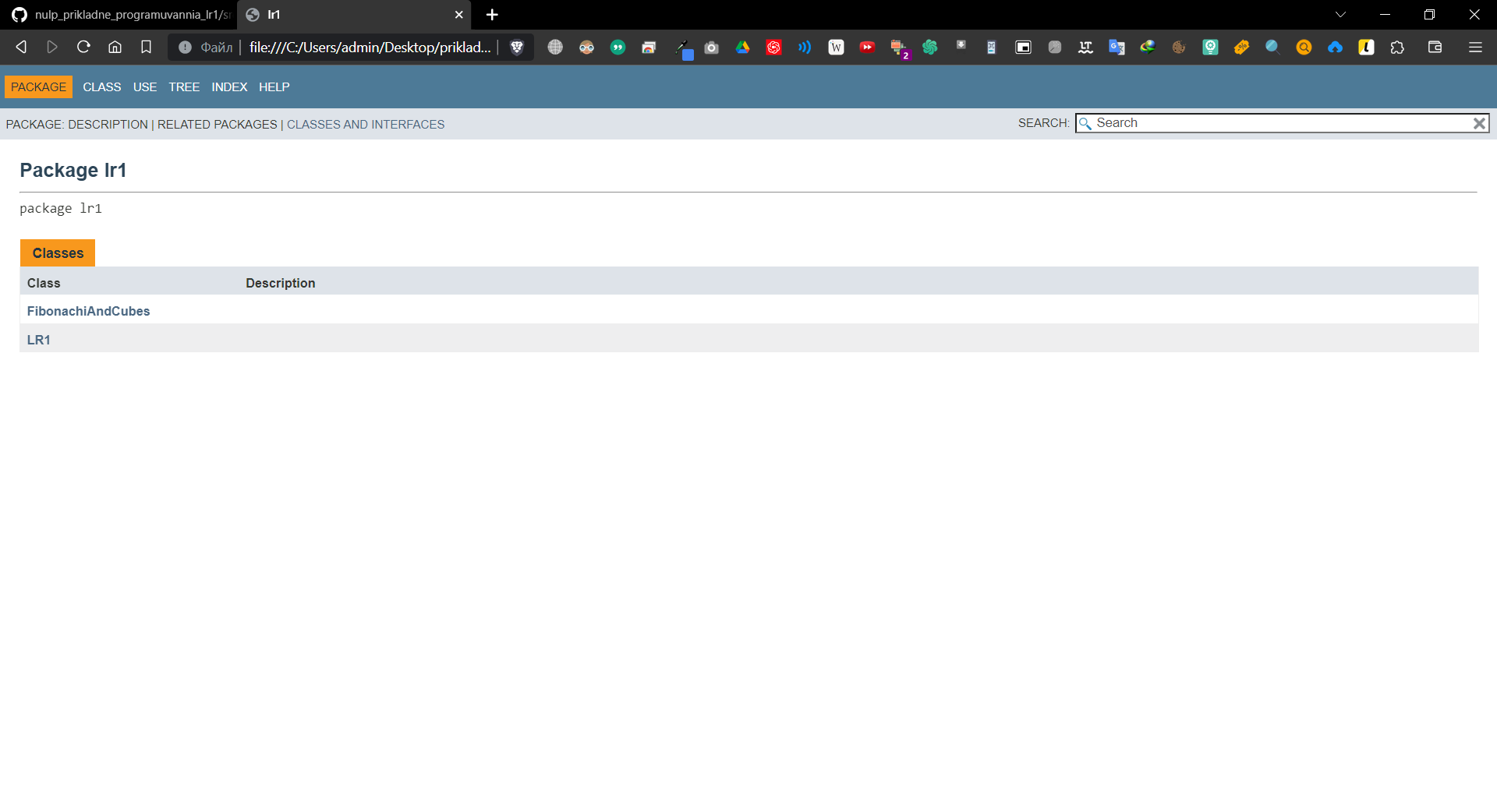


Рисунок 3 Початкова сторінка згенерованого JavaDoc

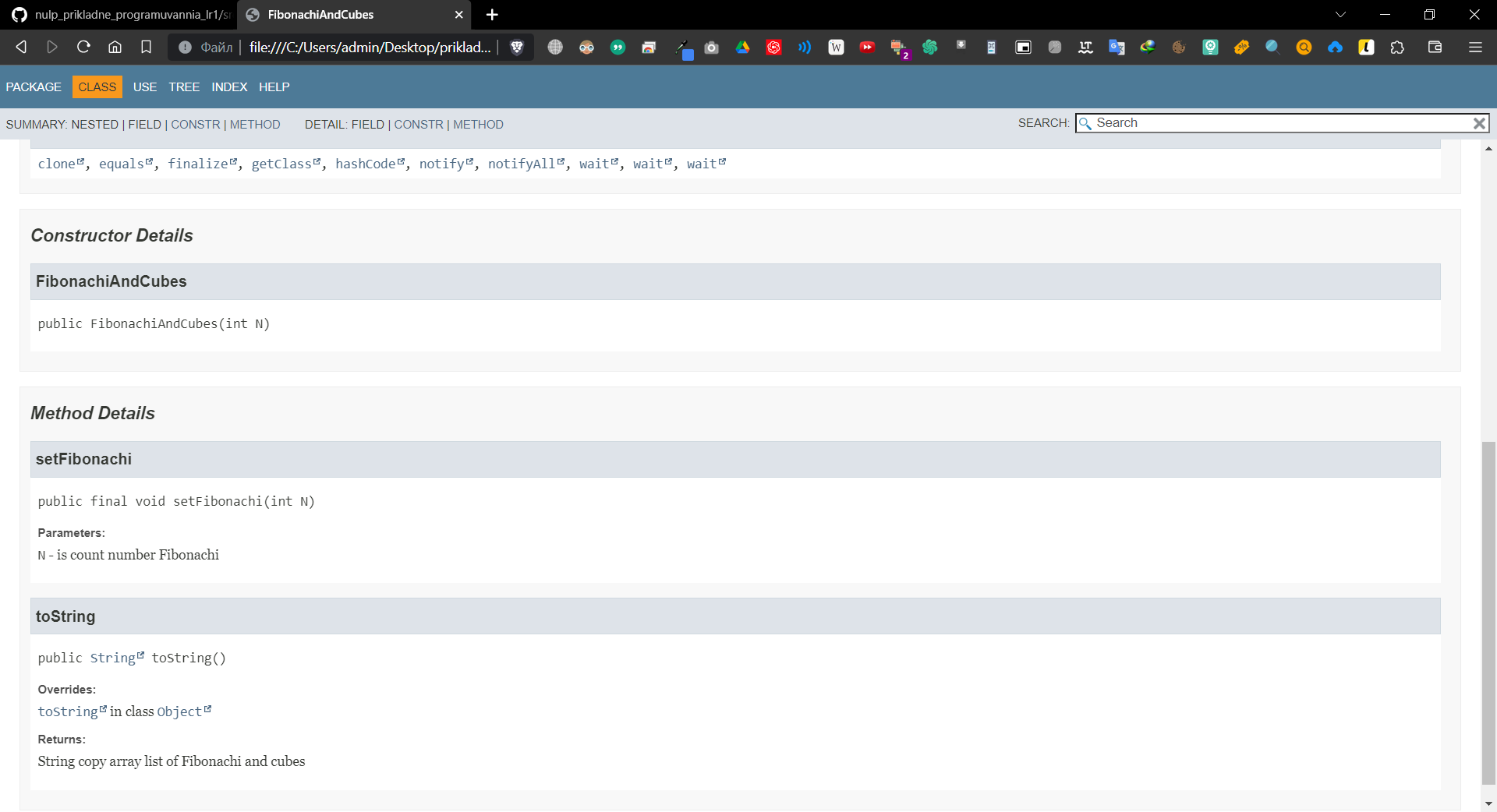


Рисунок Сторінка опису класу FibonachiAndCubes (файл реалізації завдання)

1. **Висновки**

У даній лабораторній роботі я реалізував поставлене індивідуальне завдання та зміг зробити клас у якому будуть зберігатися числа Фібоначі й їх куби.

Клас завдання містить два об’єкти ArrayList, один зберігає числа Фібоначі, а інший куби цих чисел.

При створенні об’єкта потрібно буде вказати кількість чисел у ряду Фібоначі N. При заданні кількості чисел в подальшому, потрібно буде використати метод void setFibonachi(int N). При заданні параметра кількості менш рівним нулю, програма буде встановлювати значення 0, а при заданні позитивного числа буде згенеровано цю кількість чисел у ряду. Якщо число кількості більш рівне 47, то програма встановить значення чисел 46, оскільки буде переповнення типу даних (для визначення переповнення використовується метод з бібліотеки Math – int addExact(int x, int y), який створює виняткову ситуацію). При зміні значення кількості вниз / вверх будуть вилучатись / долучатись числа ряду Фібоначі й куби цих чисел.

При генеруванні кожного числа йде перевірка чи є задане поточне число є кубом за допомогою методу boolean isCube(int number), якщо це є куб, то його буде додано в об’єкт ArrayList, який зберігає куби.

Для виводу результату був перевантажений метод toString() у якому використовувався об’єкт StringBuilder, але ще використовувався метод, який конвертував ArrayList у StringBuilder – toStringBuilderArrayList(ArrayList<Integer> arrayList, String nameArrayList).

Паралельно був опис параметрів методів для генерування JavaDoc, який було успішно згенеровано.

Аналізуючи результати виконання програми було визначено, що при максимальній кількості чисел у ряду Фібоначі (46) є всього 2 куби – 1, 8 (від 1, 2). Також, було перевірено, коли йде зміна кількості чисел у ряду Фібоначі (починаючи з 0 і вверх та навпаки) – програма продовжує працювати коректно. З цього можна зробити висновки, що програма працює коректно.

Для цієї лабораторної роботи було створено репозиторій на GitHub: <https://github.com/ManFromLviv/nulp_prikladne_programuvannia_lr1.git> .