# Міністерство освіти і науки України Національний університет «Львівська політехніка» Інститут комп'ютерних наук та інформаційних технологій Кафедра автоматизованих систем управління



# 3віт

до лабораторної роботи № 1 з дисципліни Прикладне програмування на тему:

"Основи Java"

Виконав: студент КН-105

Руденко Руслан

Прийняв:

Мельник Р.В.

#### Лабораторна робота № 1

# Тема роботи: "Основи Java"

**Мета роботи:** ознайомитись з основними компонентами інтегрованого середовища програмування мовою Java; навчитись створювати, відлагоджувати і виконувати програмні проекти;

#### Послідовність роботи:

- 1. 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.
- 2. Продокументуйте програму з п. 1 з використанням JavaDoc і згенеруйте на основі нього документацію.

# Завдання лабораторної роботи, 22 варіант:

Визначити суму N перших чисел Люка (спробуйте вивести і перевірити відповідну формулу).

# Текст програми:

#### Constants.java

```
package lab1;
/**
* Contains Golden ratio and max number in luke sequence constants
 * @author Ruslan
 * @version 1.0.0
 */
public class Constants {
    static int MAX FAST LUKE NUMBER = 67;
    static int MAX_LUKE_NUMBER = 89;
    static double GOLDEN_RATIO = (1 + Math.sqrt(5)) / 2.0;
}
Sequence.java
package lab1;
/**
* Base sequence class
 * @author Ruslan
 * @version 1.0.0
 */
public interface Sequence {
    void setAmountOfNumbers(int amountOfNumbers);
    int getAmountOfNumbers();
    long getSumOfFirstNNumbers();
}
```

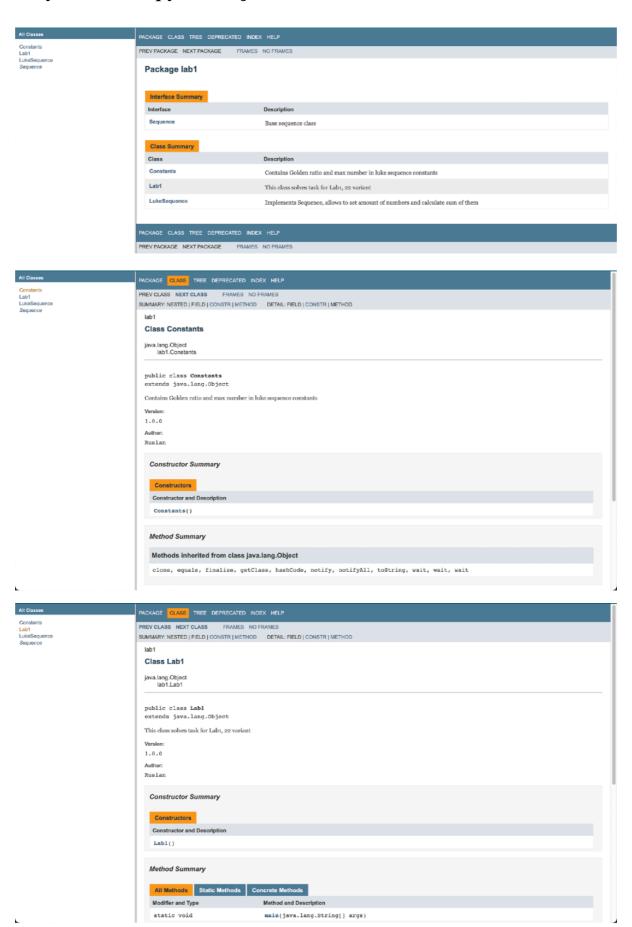
#### LukeSequence.java

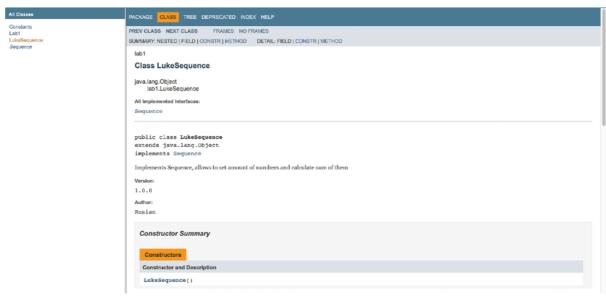
```
package lab1;
/**
 * Implements Sequence, allows to set amount of numbers and
calculate sum of them
 * @author Ruslan
 * @version 1.0.0
public class LukeSequence implements Sequence {
    private int amountOfNumbers;
    /**
     * amountOfNumbers setter
     * Note: when number is bigger than MAX LUKE NUMBER
     * it sets amountOfNumbers to MAX LUKE NUMBER
    @Override
    public void setAmountOfNumbers(int amountOfNumbers) {
        if (amountOfNumbers > Constants.MAX LUKE NUMBER) {
            System.out.printf("Maximum amount of numbers is %d\n",
Constants. MAX LUKE NUMBER);
            amountOfNumbers = Constants.MAX LUKE NUMBER;
        this.amountOfNumbers = amountOfNumbers;
    }
     * amountOfNumbers getter
     */
    @Override
    public int getAmountOfNumbers() {
        return amountOfNumbers;
    }
    /**
     * Returns sum of first N numbers of Luke sequence
    @Override
    public long getSumOfFirstNNumbers() {
        if (amountOfNumbers < 1) return 0;</pre>
        return getNLukeNumber(amountOfNumbers + 1) - 1;
    }
     * if n is less than 68 uses fast calculation via Golden ratio
     * otherwise uses standard iterative approach
     * @param n from 1 to 89
     */
```

```
private long getNLukeNumber(int n) {
       if (n <= Constants.MAX FAST LUKE NUMBER)</pre>
           return getNLukeNumberFast(n);
        long prev = 2;
        long cur = 1;
        long temp;
        for (int i = 1; i < n; i++) {
            temp = cur;
            cur += prev;
            prev = temp;
        }
        return cur;
    }
    /**
     * Calculates Luke sequence number using golden radio
     * \mbox{\it Cparam} \mbox{\it n} when over 67 gives not precise results
    private long getNLukeNumberFast(int n) {
        return Math.round(Math.pow(Constants.GOLDEN RATIO, n));
    }
}
Lab1.java
package lab1;
import java.util.Scanner;
/**
* This class solves task for Lab1, 22 variant
 * @author Ruslan
 * @version 1.0.0
 */
public class Lab1 {
    public static void main(String[] args) {
        new Lab1().run(args);
    }
    /**
     * Performs necessary calculations and outputs result for Lab1
    void run(String[] args) {
        Sequence lukeSequence = new LukeSequence();
        lukeSequence.setAmountOfNumbers(readLukeNumber(args));
        long sumOfLukeNumbers =
```

```
lukeSequence.getSumOfFirstNNumbers();
        System.out.printf(
                "Sum of first %d Luke sequence numbers: %d\n",
                lukeSequence.getAmountOfNumbers(),
                sumOfLukeNumbers);
    }
    /**
line * Reads number from arguments, if not able then from command
     */
    int readLukeNumber(String[] args) {
        try {
            return Integer.parseInt(args[0]);
        } catch (Exception e) {
            return readLukeNumberFromTerminal();
    }
    /**
     * Reads number from terminal
    int readLukeNumberFromTerminal() {
        System.out.println("Enter amount of Luke numbers to
calculate sum: ");
        Scanner input = new Scanner(System.in);
        return input.nextInt();
    }
}
```

# Результат генерування java doc:





Constants Lab1 All Methods Instance Methods Concrete Methods

Modifier and Type Method and Description

int getAmountOfNumbers ()
amountOfNumbers getter

long getSumOfFirstNNumbers()
Returns sum of first N numbers of Luke sequence

void setAmountOfNumbers (int amountOfNumbers)
amountOfNumbers etter Note: when number is bigger than MAX\_LUKE\_NUMBER it sets amountOfNumbers to
MAX\_LUKE\_NUMBER

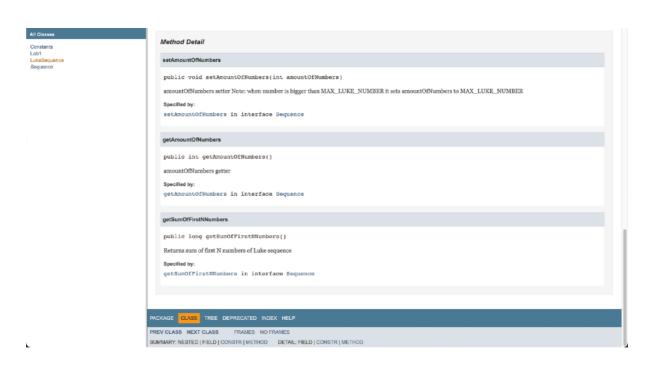
Methods inherited from class java.lang.Object

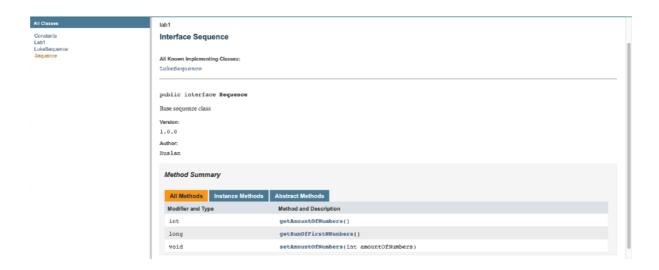
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

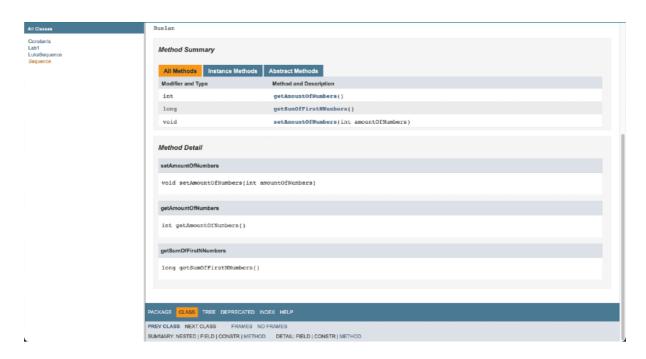
Constructor Detail

LukeSequence

public LukeSequence()







# Результати роботи програми:

Enter amount of Luke numbers to calculate sum: 5
Sum of first 5 Luke sequence numbers: 17
Process finished with exit code 0

Enter amount of Luke numbers to calculate sum: -10
Sum of first -10 Luke sequence numbers: 0

Process finished with exit code 0

Enter amount of Luke numbers to calculate sum: 100 Maximum amount of numbers is 89 Sum of first 89 Luke sequence numbers: 6440026026380244497

Process finished with exit code 0

#### Висновок

В ході лабораторної роботи я ознайомився з основними компонентами інтегрованого середовища **Intellij Idea** для програмування мовою Java. Навчився створювати, відлагоджувати і виконувати програмні проекти, вивчив принципи оголошення класу та об'єктів, рівні захисту елементів класу та застосував ці знання на практиці виконавши індивідуальне завдання.