Міністерство освіти і науки України

Національний університет “Львівська політехніка ”

Кафедра ЕОМ



**ЗВІТ**

до лабораторної роботи №5

З дисципліни: «Кросплатформні засоби програмування»

На тему: «Файли у Java»

Варіант - 7

Виконала:

ст. гр. КІ-305

Дзера А. Р.

Прийняв:

доц. каф. ЕОМ

Іванов Ю. С.

Львів – 2023

**Мета роботи:** оволодіти навиками використання засобів мови Java для роботи з потоками і файлами.

**Завдання**

1. Створити клас, що реалізує методи читання/запису у текстовому і двійковому форматах результатів роботи класу, що розроблений у лабораторній роботі №4. Написати програму для тестування коректності роботи розробленого класу.

2. Для розробленої програми згенерувати документацію.

3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.

5. Дати відповідь на контрольні запитання.

Варіант завдання –

**Код програми**

**Файл Equations.java**

package KI305.Dzera.Lab5;  
  
*/\*\*  
 \* Клас Equations містить методи для розрахунку математичних виразів та обробки винятків,  
 \* які можуть виникнути під час розрахунків.  
 \*/*public class Equations {  
 */\*\*  
 \* Метод calculate виконує розрахунок виразу для заданого значення x у градусах.  
 \*  
 \** ***@param*** *x значення x у градусах для розрахунку виразу  
 \** ***@return*** *результат розрахунку виразу для заданого значення x  
 \** ***@throws*** *CalcException виняток, який виникає у випадку некоректних вхідних даних або математичних помилок під час розрахунків  
 \*/* public double calculate(double x) throws CalcException  
 {  
 double y;  
 double sinValue = 0;  
 double rad = x \* Math.*PI* / 180.0; //переводимо x в радіани  
  
 try  
 {  
 //double xInRadians = Math.toRadians(x);  
 sinValue = Math.*sin*(7 \* Math.*toRadians*(x) - 1);  
 y = (1.0/Math.*tan*(rad)/sinValue);  
  
 //якщо результат не є числом, то генеруємо виключення  
 if (Double.*isNaN*(y) || Double.*isInfinite*(y) || Double.*isInfinite*(-y) || x == 90 || x == -90)  
 throw new ArithmeticException();  
 }  
 catch (ArithmeticException ex)  
 {  
 // створимо виключення вищого рівня з поясненням причини виникнення помилки  
 if (x==90 || x==-90)  
 throw new CalcException("Exception reason: Illegal value of X for tg calculation");  
 else if (sinValue == 0) {  
 throw new CalcException("Exception reason: sin(7x-1) = 0");  
 } else  
 throw new CalcException("Unknown reason of the exception during exception calculation");  
 }  
 return y;  
 }  
}

**Файл CalcException.java**

package KI305.Dzera.Lab5;  
  
*/\*\*  
 \* Клас CalcException представляє виняток, який може виникнути при виконанні операцій калькулятора.  
 \* Цей клас є підкласом класу ArithmeticException і дозволяє визначити власні повідомлення про помилки.  
 \*/*class CalcException extends ArithmeticException{  
 */\*\*  
 \* Конструктор без параметрів. Створює новий об'єкт CalcException з порожнім повідомленням про помилку.  
 \*/* public CalcException(){  
  
 }  
  
 */\*\*  
 \* Конструктор з параметром. Створює новий об'єкт CalcException зі зазначеним повідомленням про помилку.  
 \*  
 \** ***@param*** *cause Повідомлення про причину виникнення помилки.  
 \*/* public CalcException(String cause)  
 {  
 super(cause);  
 }  
}

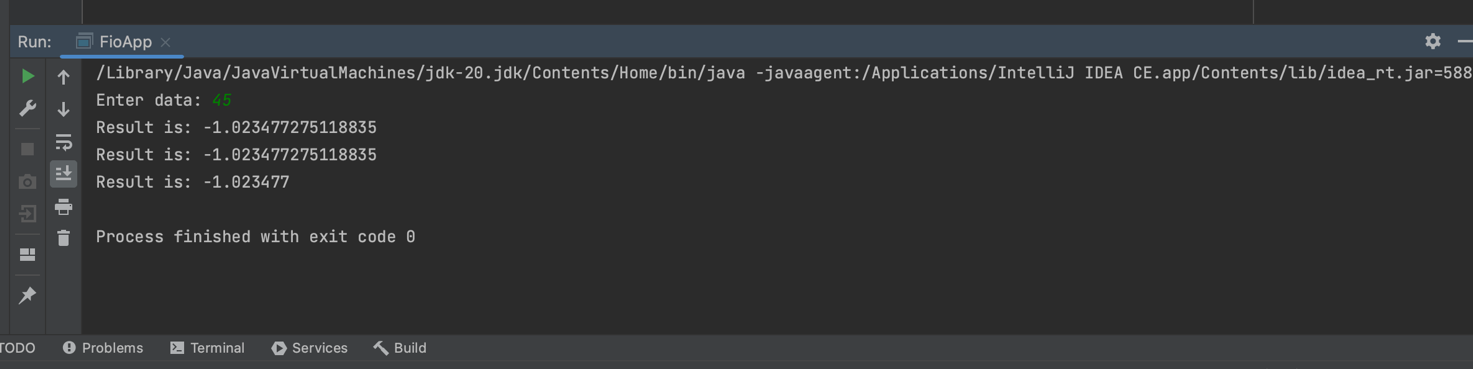
**Файл FioApp.java**

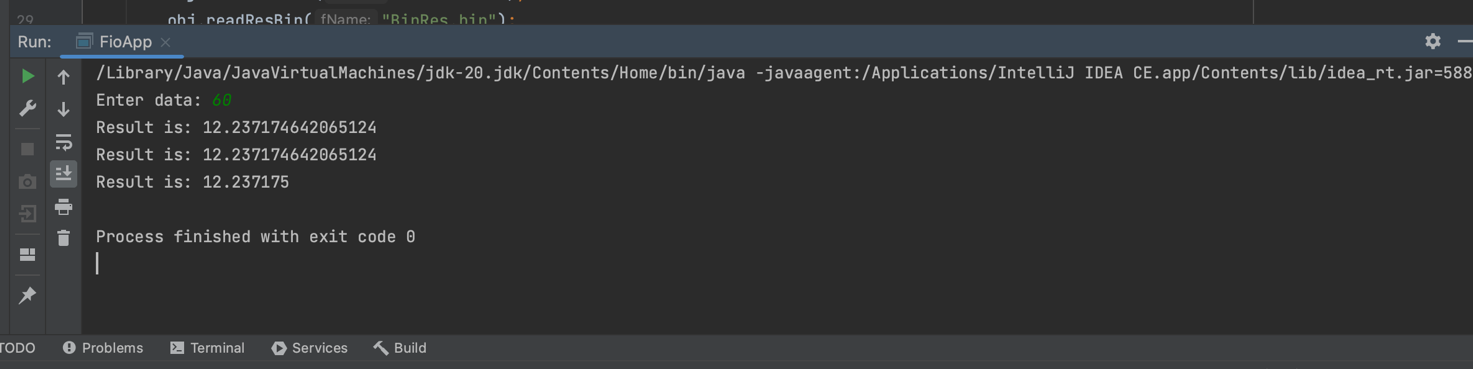
package KI305.Dzera.Lab5;  
  
import java.io.\*;  
import java.util.\*;  
  
*/\*\*  
 \* Клас FioApp представляє програму для використання класу CalcWFio. Програма демонструє введення  
 \* даних, їх обробку та збереження результатів у текстовий та бінарний файл, а також читання результатів  
 \* з цих файлів та їх виведення на екран.  
 \*/*public class FioApp {  
 */\*\*  
 \* Основний метод програми, який виконує введення даних, обробку та збереження результатів у файлі,  
 \* а також читання результатів з файлів та їх виведення на екран.  
 \*  
 \** ***@param*** *args параметри командного рядка (не використовуються у цій програмі)  
 \** ***@throws*** *FileNotFoundException виняток, який виникає, якщо не вдається знайти вказаний файл  
 \** ***@throws*** *IOException виняток, який виникає при помилках вводу/виводу під час роботи з файлами  
 \*/* public static void main (String[] args) throws FileNotFoundException, IOException{  
 CalcWFio obj = new CalcWFio();  
 Scanner s = new Scanner(System.*in*);  
 System.*out*.print("Enter data: ");  
 double data = s.nextDouble();  
 obj.calculate(data);  
 System.*out*.println("Result is: " + obj.getResult());  
 obj.writeResTxt("textRes.txt");  
 obj.writeResBin("BinRes.bin");  
 obj.readResBin("BinRes.bin");  
 System.*out*.println("Result is: " + obj.getResult());  
 obj.readResTxt("textRes.txt");  
 System.*out*.println("Result is: " + obj.getResult());  
 }  
}

**Файл CalcWFio.java**

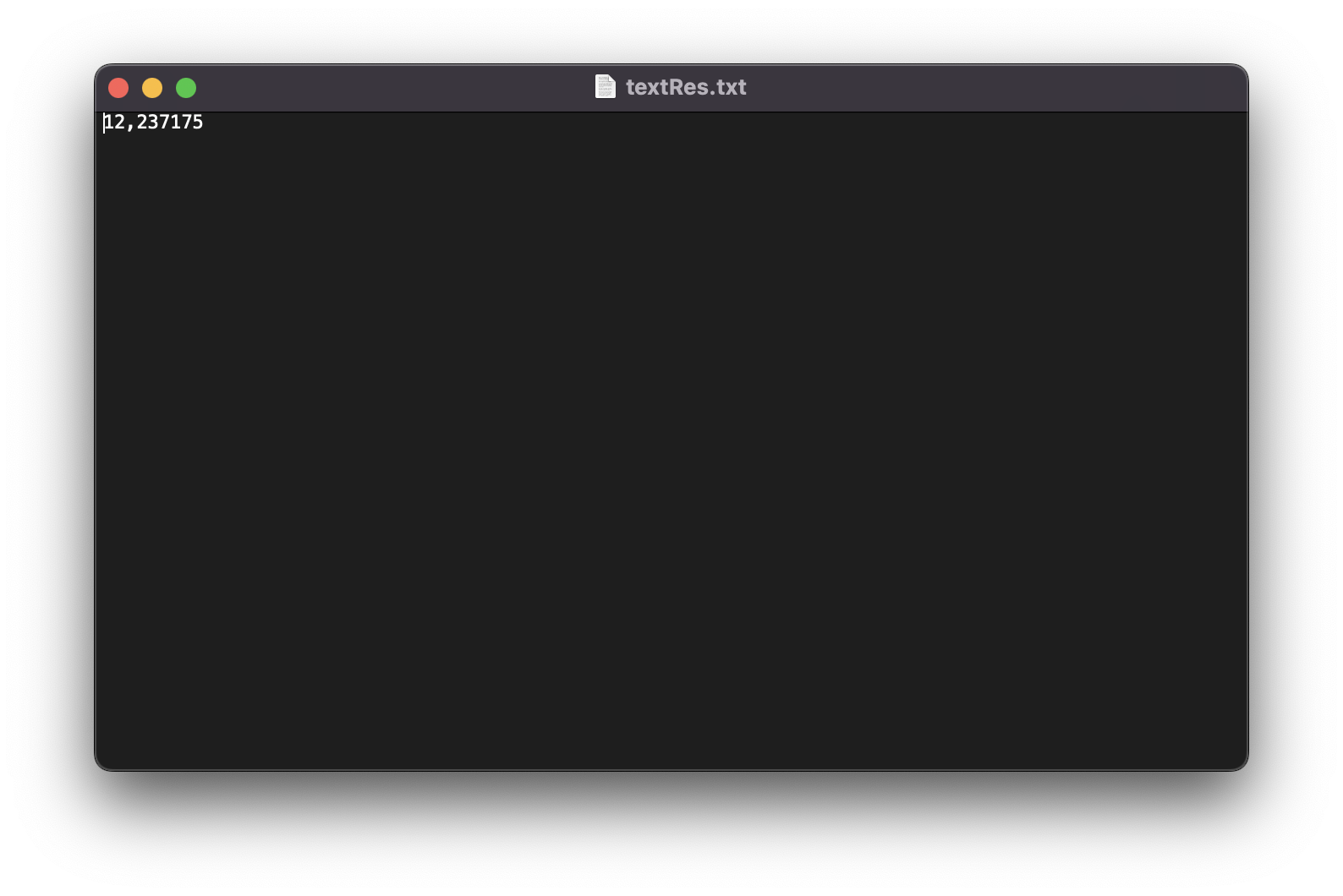
package KI305.Dzera.Lab5;  
  
import java.io.\*;  
import java.util.\*;  
  
*/\*\*  
 \* Клас CalcWFio містить методи для розрахунку математичного виразу та зберігання результатів  
 \* у текстових та бінарних файлах.  
 \*/*public class CalcWFio {  
  
 */\*\*  
 \* Метод відповідає за запис результату у текстовий файл.  
 \*  
 \** ***@param*** *fName ім'я текстового файлу, в який буде записаний результат  
 \** ***@throws*** *FileNotFoundException виняток, який виникає, якщо не вдається знайти вказаний файл  
 \*/* public void writeResTxt(String fName) throws FileNotFoundException {  
 PrintWriter f = new PrintWriter(fName);  
 f.printf("%f ",result);  
 f.close();  
 }  
  
 */\*\*  
 \* Метод відповідає за читання результату з текстового файлу.  
 \*  
 \** ***@param*** *fName ім'я текстового файлу, з якого буде зчитано результат  
 \*/* public void readResTxt(String fName) {  
 try {  
 File f = new File(fName);  
 if (f.exists()) {  
 Scanner s = new Scanner(f);  
 result = s.nextDouble();  
 s.close();  
 } else {  
 throw new FileNotFoundException("File " + fName + "not found");  
 }  
 }  
 catch (FileNotFoundException ex){  
 System.*out*.print(ex.getMessage());  
 }  
 }  
  
 */\*\*  
 \* Метод відповідає за запис результату у бінарний файл.  
 \*  
 \** ***@param*** *fName ім'я бінарного файлу, в який буде записаний результат  
 \** ***@throws*** *FileNotFoundException виняток, який виникає, якщо не вдається знайти вказаний файл  
 \** ***@throws*** *IOException виняток, який виникає при помилках вводу/виводу під час роботи з файлами  
 \*/* public void writeResBin(String fName) throws FileNotFoundException, IOException {  
 DataOutputStream f = new DataOutputStream(new FileOutputStream(fName));  
 f.writeDouble(result);  
 f.close();  
 }  
  
 */\*\*  
 \* Метод відповідає за читання результату з бінарного файлу.  
 \*  
 \** ***@param*** *fName ім'я бінарного файлу, з якого буде зчитано результат  
 \** ***@throws*** *FileNotFoundException виняток, який виникає, якщо не вдається знайти вказаний файл  
 \** ***@throws*** *IOException виняток, який виникає при помилках вводу/виводу під час роботи з файлами  
 \*/* public void readResBin(String fName) throws FileNotFoundException, IOException  
 {  
 DataInputStream f = new DataInputStream(new FileInputStream(fName));  
 result = f.readDouble();  
 f.close();  
 }  
  
 */\*\*  
 \* Метод виконує розрахунок математичного виразу для заданого значення x у градусах.  
 \*  
 \** ***@param*** *x значення x у градусах для розрахунку виразу  
 \*/* public void calculate(double x) {  
 result = ((1.0/Math.*tan*(x \* Math.*PI* / 180.0))/Math.*sin*(7 \* Math.*toRadians*(x) - 1));;  
 }  
  
 */\*\*  
 \* Метод повертає результат останнього розрахунку.  
 \*  
 \** ***@return*** *результат розрахунку  
 \*/* public double getResult() {  
 return result;  
 }  
 private double result; //змінна для збереження результату розрахунку  
}

**Результати роботи програми**

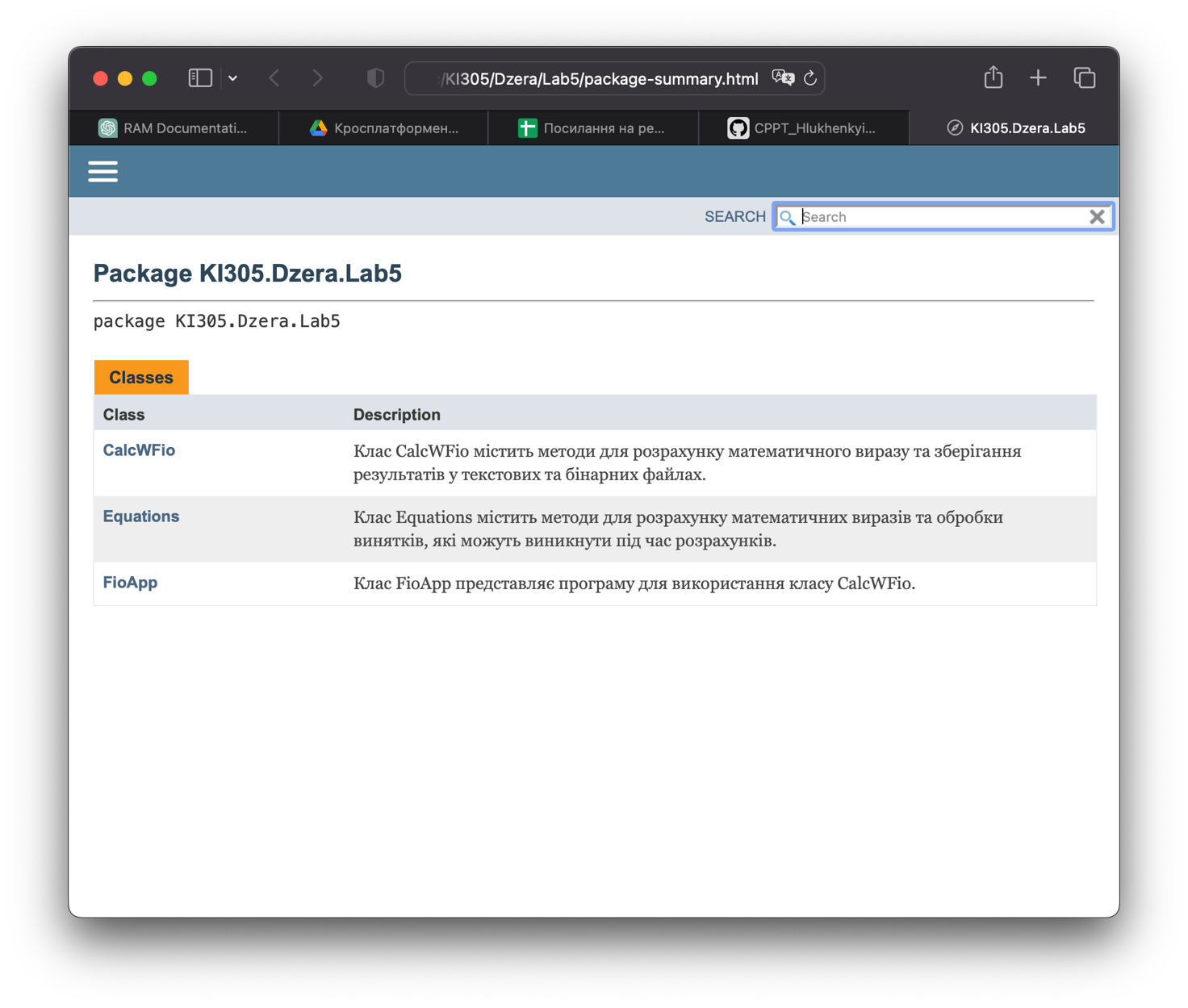


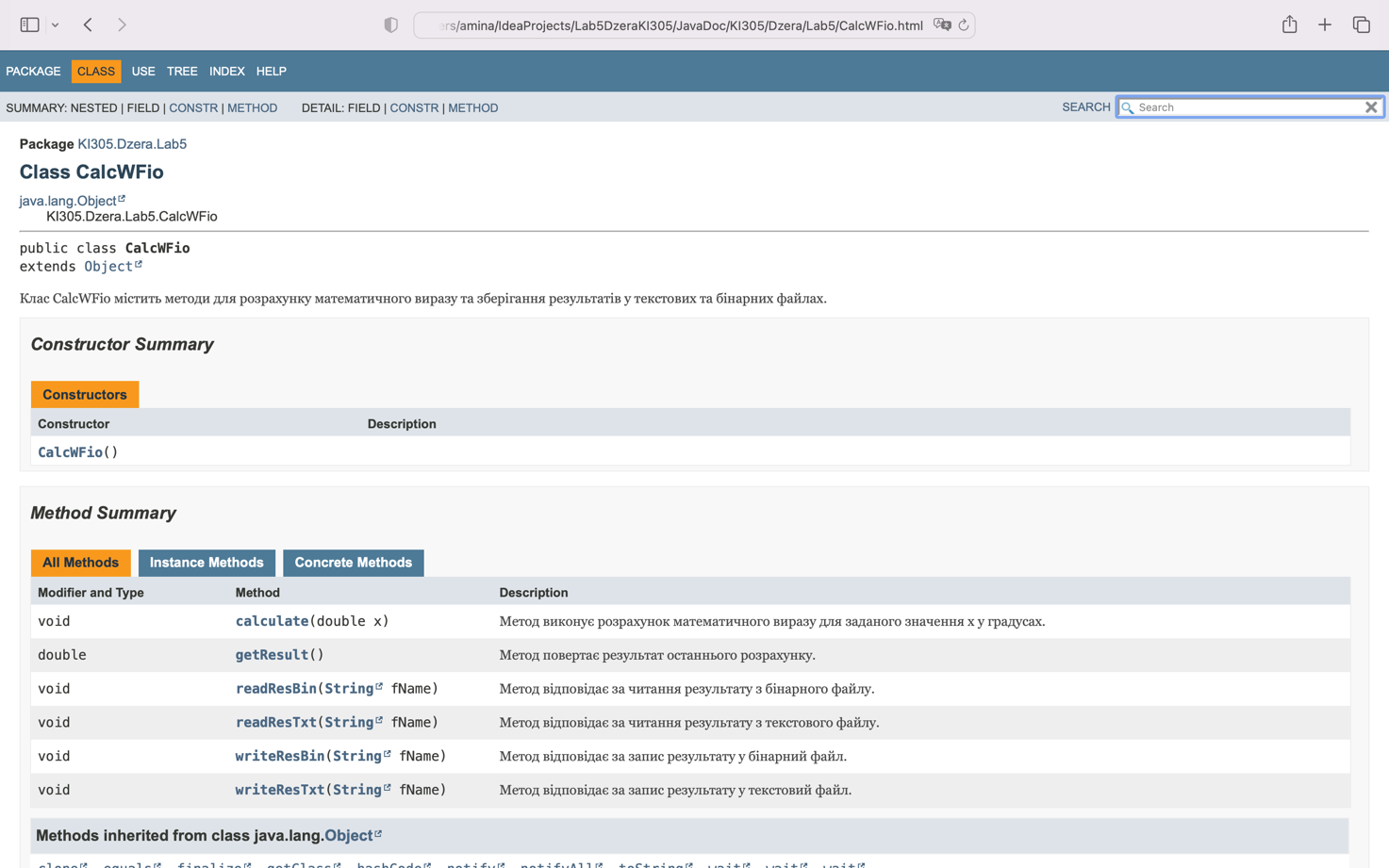


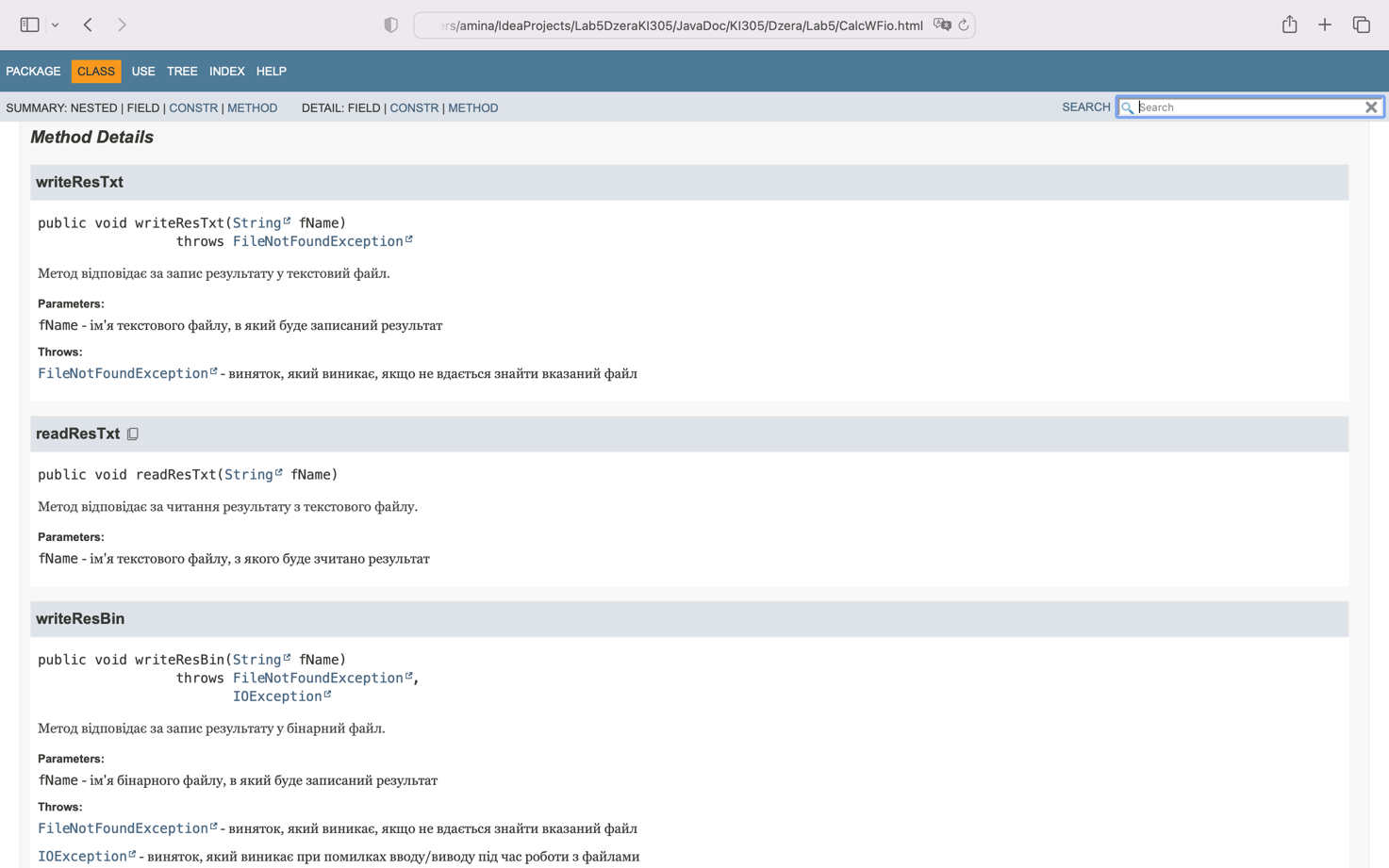
**Вміст файла textRes.txt**

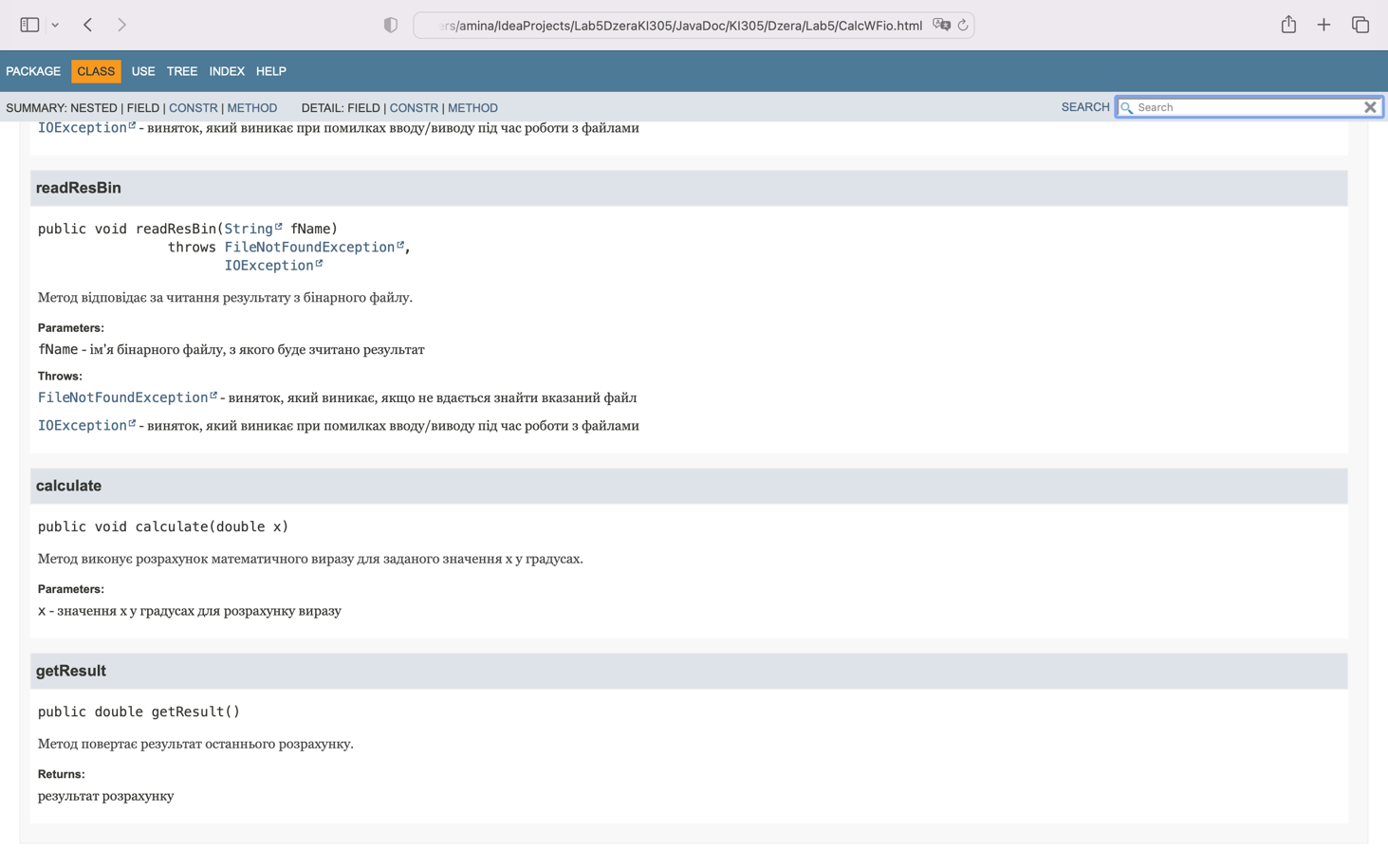


**Фрагмент згенерованої документації**

****

****

****

****

**Контрольні питання**

1. Розкрийте принципи роботи з файловою системою засобами мови Java.

Java надає низку класів для роботи з файловою системою. Основні операції включають в себе читання, запис і видалення файлів. Це можна робити засобами класів *File*, *FileReader*, *FileWriter*, *BufferedReader*, і *BufferedWriter*. *File* використовується для представлення шляху до файлу або каталогу. *FileReader* і *FileWriter* використовуються для читання та запису текстових файлів відповідно. *BufferedReader* і *BufferedWriter* забезпечують буферизоване читання і запис, що може покращити продуктивність при роботі з файлами.

1. Охарактеризуйте клас Scanner.

*Scanner* ⎯ це клас в мові java, який дозволяє зчитувати вхідні дані з різних джерел, таких як консоль, файли тощо. Він надає методи для розбору примітивних типів даних і рядків.

1. Наведіть приклад використання класу Scanner.

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Введіть ваше ім'я: ");

String name = scanner.nextLine();

System.out.println("Привіт, " + name + "!");

scanner.close();

}

}

1. За допомогою якого класу можна здійснити запис у текстовий потік?

Для запису у текстовий потік можна використовувати класи *FileWriter* або *PrintWriter*.

1. Охарактеризуйте клас PrintWriter.

*PrintWriter* ⎯ це клас, який дозволяє записувати рядки та інші типи даних у текстовий файл. Він забезпечує методи, схожі на System.out.println() для запису даних у файл.

1. Розкрийте методи читання/запису двійкових даних засобами мови Java.

Для читання і запису двійкових даних можна використовувати класи *FileInputStream* і *FileOutputStream*, які спадають від класів InputStream і OutputStream, відповідно.

1. Призначення класів DataInputStream і DataOutputStream.

*DataInputStream* і *DataOutputStream* ⎯ це класи в Java, які дозволяють зчитувати і записувати примітивні типи даних у бінарний формат.

1. Який клас мови Java використовується для здійснення довільного доступу до файлів.

Для довільного доступу до файлів використовується клас *RandomAccessFile*.

1. Охарактеризуйте клас RandomAccessFile.

*RandomAccessFile* дозволяє читати і записувати дані в файлі як текстові, так і бінарні. Він надає можливість переміщення вказівника у файлі і зчитування/запис даних з конкретної позиції.

1. Який зв’язок між інтерфейсом DataOutput і класом DataOutputStream?ʼ

*DataOutputStream* реалізує інтерфейс DataOutput. Це означає, що він надає всі методи, визначені в інтерфейсі DataOutput, які дозволяють записувати дані примітивних типів у бінарний потік.

**Висновок**

Під час виконання даної лабораторної роботи я оволоділа навичками використання засобів мови Java для роботи з потоками і файлами.