ФЕДЕРАЛЬНОЕ Государственное АВТОНОМНОЕ образовательное УЧРЕЖДЕНИЕ ВЫСШЕГО образования

**БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ**

**ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»**

**(НИУ «БелГУ»)**

**ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ**

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

**Отчет**

по дисциплине «Разработка программных приложений на языках высокого уровня»

специальности 09.03.03 «Прикладная информатика»

**Лабораторная работа № 3**

студента очного отделения

3 курса 12002005 группы

Огородникова Льва Олеговича

Проверил:

старший преподаватель

Лифиренко Максим Вячеславович

БЕЛГОРОД 2023

**Цель работы:** изучить Java Code Convention, получить практические навыки по логированию с помощью Log4J.

(Вариант № 15)

**Листинг класса Main**

package com.company;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
import java.text.ParseException;  
  
public class Main  
{  
 */\*\* Метод main предназначен для выбора режима ввода данных,  
 \* выбора задачи и вызова соответствующего метода  
 \** ***@param*** *args Не используется.  
 \** ***@exception*** *FileNotFoundException При ошибке открытия файла.\*/* public static void main(String[] args) throws FileNotFoundException, ParseException  
 {  
 System.*out*.println("Введите 1 для режима работы ввода параметров задач" +  
 " с клавиатуры или 2 для ввода параметров из файла");  
 Scanner scanner = new Scanner(System.*in*);  
 int inputMode = scanner.nextInt();  
 if (inputMode == 1 || inputMode == 2)  
 {  
 Tasks tasks = new Tasks();  
 tasks.task1(inputMode);  
 tasks.task2(inputMode);  
 tasks.task3(inputMode);  
 tasks.task4(inputMode);  
 tasks.task5(inputMode);  
 tasks.task6(inputMode);  
 tasks.task7(inputMode);  
 tasks.task8(inputMode);  
 }  
 else  
 {  
 System.*out*.println("Нет такого режима работы программы");  
 }  
 }  
}

**Листинг класса Tasks**

package com.company;  
import java.io.File;  
import org.apache.logging.log4j.LogManager;  
import org.apache.logging.log4j.Logger;  
import java.util.Scanner;  
import java.io.IOException;  
import java.io.FileNotFoundException;  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.TimeZone;  
  
*/\*\* Класс Tasks описывает 8 задач 2-ой лабораторной работы. "  
 \** ***@author*** *Огородников Лев  
 \*/*public class Tasks  
{  
  
 */\*\* Метод task1 вычисляет значение выражения по формуле 15 варианта.  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* private static Logger *logger* = LogManager.*getLogger*(Tasks.class.getName());  
 public void task1(int inputMode)  
 {  
 double x=0;  
 double y=0;  
 double result;  
 if (inputMode == 1)  
 {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.print("Введите x ");  
 x = scanner.nextDouble();  
 System.*out*.print("Введите y ");  
 y = scanner.nextDouble();  
 }  
 else  
 {  
 try {  
 String path = "rc/task\_1\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 x = scanner.nextDouble();  
 System.*out*.println("Ввод x из файла, x = " + x);  
 y = scanner.nextDouble();  
 System.*out*.println("Ввод y из файла, y = " + y);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_1\_data\" отсутствует");  
 }  
 }  
 result = x \* Math.*log*(x) + y/(Math.*cos*(x)-x/3);  
 System.*out*.println("Результат: " + result);  
 *logger*.debug("task1 закончил работу");  
 }  
  
 */\*\*  
 \* Вычисляет значение выражения 2x^4 - 3x^3 + 4x^2 - 5x + 6 для заданного значения x.  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task2(int inputMode)  
 {  
 double x=0;  
 if (inputMode == 1)  
 {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.print("Введите x: ");  
 x = scan.nextDouble();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_2\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 x = scanner.nextDouble();  
 System.*out*.println("Ввод из файла, x = " + x);}  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_2\_data\" отсутствует");  
 }  
 }  
 double x2 = x \* x;  
 double x4 = x2 \* x2;  
 double x3 = x2 \* x;  
 double result = (2 \* x4) - (3 \* x3) + (4 \* x2) - (5 \* x) + 6;  
 System.*out*.println("Результат выражения = " + result);  
 *logger*.info("какая-то информация о задании");  
 }  
  
 */\*\* Метод task3 считает количество целых чисел среди a, b, c  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task3(int inputMode)  
 {  
 double a=0;  
 double b=0;  
 double c=0;  
 if (inputMode == 1)  
 {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.print("Введите число a: ");  
 a = scan.nextDouble();  
 System.*out*.print("Введите число b: ");  
 b = scan.nextDouble();  
 System.*out*.print("Введите число c: ");  
 c = scan.nextDouble();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_3\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 a = scanner.nextDouble();  
 System.*out*.println("Ввод a из файла, a = " + a);  
 b = scanner.nextDouble();  
 System.*out*.println("Ввод b из файла, b = " + b);  
 c = scanner.nextDouble();  
 System.*out*.println("Ввод c из файла, c = " + c);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_3\_data\" отсутствует");  
 }  
 }  
 int count = 0;  
 if (a == Math.*floor*(a) && !Double.*isInfinite*(a))  
 {  
 count++;  
 }  
 if (b == Math.*floor*(b) && !Double.*isInfinite*(b))  
 {  
 count++;  
 }  
 if (c == Math.*floor*(c) && !Double.*isInfinite*(c))  
 {  
 count++;  
 }  
 System.*out*.println("Количество целых чисел = " + count);  
 *logger*.warn("Предупреждение!");  
 }  
  
 */\*\*Метод task4 определяет, можно ли построить треугольник с заданными длинами сторон,  
 \* и является ли он остроугольным.  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task4(int inputMode)  
 {  
 double a=0;  
 double b=0;  
 double c=0;  
 if (inputMode == 1)  
 {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.print("Введите длину стороны a: ");  
 a = scan.nextDouble();  
 System.*out*.print("Введите длину стороны b: ");  
 b = scan.nextDouble();  
 System.*out*.print("Введите длину стороны c: ");  
 c = scan.nextDouble();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_4\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 a = scanner.nextDouble();  
 System.*out*.println("Ввод стороны a из файла, a = " + a);  
 b = scanner.nextDouble();  
 System.*out*.println("Ввод стороны b из файла, b = " + b);  
 c = scanner.nextDouble();  
 System.*out*.println("Ввод стороны c из файла, c = " + c);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_4\_data\" отсутствует");  
 }  
 }  
 if (a + b > c && a + c > b && b + c > a)  
 {  
 double maxSide = Math.*max*(Math.*max*(a, b), c);  
 double sumSquares = 0;  
 if (a == maxSide)  
 {  
 sumSquares = b \* b + c \* c;  
 }  
 else if (b == maxSide)  
 {  
 sumSquares = a \* a + c \* c;  
 }  
 else  
 {  
 sumSquares = a \* a + b \* b;  
 }  
 if (sumSquares > maxSide \* maxSide)  
 {  
 System.*out*.print("Треугольник остроугольный");  
 }  
 else if (sumSquares == maxSide \* maxSide)  
 {  
 System.*out*.print("Треугольник прямоугольный");  
 }  
 else  
 {  
 System.*out*.print("Треугольник тупоугольный");  
 }  
 }  
 else  
 {  
 *logger*.error("Треугольник нельзя построить");  
 }  
 }  
  
 */\*\*Метод task5 Программа выводит значение числа N в зависимости от остатка от деления на 4  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task5(int inputMode)  
 {  
 int N=0;  
 if (inputMode == 1)  
 {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.print("Введите натуральное число N: ");  
 N = scan.nextInt();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_5\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 N = scanner.nextInt();  
 System.*out*.println("Ввод натурального число N из файла, N = " + N);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_5\_data\" отсутствует");  
 }  
 }  
 int remainder = N % 4;  
 double k = N/4;  
 switch (remainder)  
 {  
 case 0:  
 System.*out*.println("N = " + N \* k);  
 break;  
 case 1:  
 System.*out*.println("N = " + N \* k + 1);  
 break;  
 case 2:  
 System.*out*.println("N = " + N \* k + 2);  
 break;  
 case 3:  
 System.*out*.println("N = " + N \* k + 3);  
 break;  
 default:  
 System.*out*.println("Некорректное значение числа");  
 break;  
 }  
 }  
  
 */\*\* Метод task6 выводит на экран все числа Пифагора, меньшие N..  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task6(int inputMode)  
 {  
 int N=0;  
 if (inputMode == 1)  
 {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.print("Введите натуральное число N: ");  
 N = scan.nextInt();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_6\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 N = scanner.nextInt();  
 System.*out*.println("Ввод натурального число N из файла, N = " + N);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_6\_data\" отсутствует");  
 }  
 }  
 for (int a = 1; a < N; a++)  
 {  
 for (int b = a + 1; b < N; b++)  
 {  
 int c = (int) Math.*sqrt*(a \* a + b \* b);  
 if (c \* c == a \* a + b \* b && c < N)  
 {  
 System.*out*.println(a + "^2 + " + b + "^2 = " + c + "^2");  
 }  
 }  
 }  
 }  
  
 */\*\* Метод task7 находит сумму n членов ряда  
 \** ***@param*** *inputMode параметр передает режим ввода данных.  
 \*/* public void task7(int inputMode)  
 {  
 int n=0;  
 float x=0;  
 float sum = 0;  
 if (inputMode == 1)  
 {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.println("Введите число n ");  
 n = scanner.nextInt();  
 System.*out*.println("Введите число x ");  
 x = scanner.nextFloat();  
 }  
 else  
 {  
 try {  
 String path = "src/task\_7\_data.txt";  
 Scanner scanner = new Scanner(new File(path));  
 n = scanner.nextInt();  
 System.*out*.println("Ввод из файла числa n " + n);  
 x = scanner.nextFloat();  
 System.*out*.println("Ввод из файла числa x " + x);  
 }  
 catch (IOException e)  
 {  
 *logger*.fatal("Файл \"task\_7\_data\" отсутствует");  
 }  
 }  
 for (int i = 1; i <= n; i++)  
 {  
 long fact = 1;  
 for (int j = 2; j <= 2\*i-1; j++)  
 {  
 fact = fact \* j;  
 }  
 sum += Math.*pow*(-1,i-1)\*(Math.*pow*(x,2\*i-1)/fact);  
 }  
 System.*out*.println("Сумма n членов ряда = " + sum);  
 }  
  
 */\*\*  
 Метод task8 для получения расписания звонков на весь учебный день в учебном заведении.  
 Задаются параметры начала учебного дня, продолжительности пары, продолжительности  
 обычного и большого перерывов (и их место в расписании), количество пар.* ***@param*** *inputMode параметр передает режим ввода данных.* ***@exception*** *FileNotFoundException При ошибке открытия файла.* ***@exception*** *ParseException При ошибке в парсинге даты.  
 \*/* public void task8(int inputMode) throws FileNotFoundException, ParseException {  
 long lessonDuration, standardInterval, bigInterval, time;  
 int indexBigInterval, countLessons;  
 Date startDay;  
 SimpleDateFormat formater = new SimpleDateFormat("HH:mm");  
 formater.setTimeZone(TimeZone.*getTimeZone*("GMT"));  
 if (inputMode == 1)  
 {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.print("Начало учебного дня: ");  
 startDay = formater.parse(scanner.nextLine());  
 time = startDay.getTime();  
 System.*out*.print("Продолжительность пары: ");  
 lessonDuration = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.print("Продолжительность обычного перерыва: ");  
 standardInterval = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.print("Продолжительность большого перерыва: ");  
 bigInterval = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.print("Его позиция: ");  
 indexBigInterval = scanner.nextInt();  
 System.*out*.print("Кол-во пар: ");  
 countLessons = scanner.nextInt();  
 System.*out*.println();  
 }  
 else  
 {  
 Scanner scanner = new Scanner(new File("src/task\_8\_data.txt"));  
 startDay = formater.parse(scanner.nextLine());  
 time = startDay.getTime();  
 System.*out*.println("Начало учебного дня: " + formater.format(new Date(time)));  
 lessonDuration = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.println("Продолжительность пары: " + formater.format(new Date(lessonDuration)));  
 standardInterval = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.println("Продолжительность обычного перерыва: " + formater.format(new Date(standardInterval)));  
 bigInterval = formater.parse(scanner.nextLine()).getTime();  
 System.*out*.println("Продолжительность большого перерыва: " + formater.format(new Date(bigInterval)));  
 indexBigInterval = scanner.nextInt();  
 System.*out*.println("Его позиция: " + indexBigInterval);  
 countLessons = scanner.nextInt();  
 System.*out*.println("Кол-во пар: " + countLessons);  
 System.*out*.println();  
 time = startDay.getTime();  
 }  
 for (int i = 1; i <= countLessons; i++)  
 {  
 System.*out*.println(i + " пара");  
 System.*out*.println(formater.format(new Date(time)));  
 time += lessonDuration;  
 System.*out*.println(formater.format(new Date(time)));  
 System.*out*.println();  
 if (i != countLessons)  
 {  
 System.*out*.println("Перерыв:");  
 System.*out*.println(formater.format(new Date(time)));  
 if (i != indexBigInterval)  
 {  
 time += standardInterval;  
 System.*out*.println(formater.format(new Date(time)));  
 }  
 else  
 {  
 time += bigInterval;  
 System.*out*.println(formater.format(new Date(time)));  
 }  
 System.*out*.println();  
 }  
 }  
 }  
}

**Листинг файла log4j2.xml**

<?xml version="1.0" encoding="UTF-8"?>  
<Configuration status="warn">  
 <Appenders>  
 <Console name="LogToConsole" target="SYSTEM\_OUT">  
 <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %-5level %logger{36} - %msg%n"/>  
 </Console>  
  
 <File name="LogToFile1" fileName="logs/app1.log" append="false">  
 <PatternLayout>  
 <Pattern>%-1p %d{yyyy-MM-dd HH:mm:ss} %c{1}:%L - %m%n</Pattern>  
 </PatternLayout>  
 </File>  
  
 <File name="LogToFile2" fileName="logs/app2.log" append="false">  
 <PatternLayout>  
 <Pattern>%-1p %d{yyyy-MM-dd HH:mm:ss} %c{1}:%L - %m%n</Pattern>  
 </PatternLayout>  
 </File>  
  
 </Appenders>  
 <Loggers>  
 <!-- avoid duplicated logs with additivity=false -->  
 <Logger name="com.company.Tasks" level="debug" additivity="false">  
 <AppenderRef ref="LogToFile1" level="debug"/>  
 <AppenderRef ref="LogToFile2" level="warn"/>  
 <AppenderRef ref="LogToConsole"/>  
 </Logger>  
 <Root level="error">  
  
 </Root>  
 </Loggers>  
</Configuration>

Результаты работы программы представлены на рисунках 1–8.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 – Первая задача

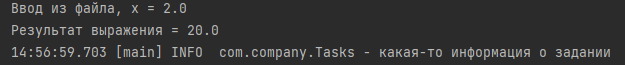


Рисунок 2 –Вторая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 – Третья задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 – Четвертая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 – Пятая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6 – Шестая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 – Седьмая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8 – Восьмая задача

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 – Файл логирования уровня debug и выше

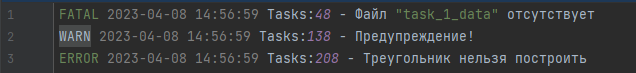


Рисунок 10 – Файл логирования уровня warn и выше

Вывод: были изучены Java Code Convention, и получены практические навыки по логированию с помощью Log4J.