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 лабораторной работе №3**

студента очного отделения

третьего курса 12002005 группы

Набойченко Николая Сергеевича

Проверил: Ст.преподаватель

Лифиренко Максим Вячеславович

**БЕЛГОРОД**, 2023

**Цель работы:** изучить основные понятия, синтаксис языка Java и общую структуру программ, получить практические навыки программирования на языке Java.

**Результаты выполнения работы (Вариант 13)**

1. Вычислить значения выражения по формуле:



На рисунке 1 представлен код задания 1.

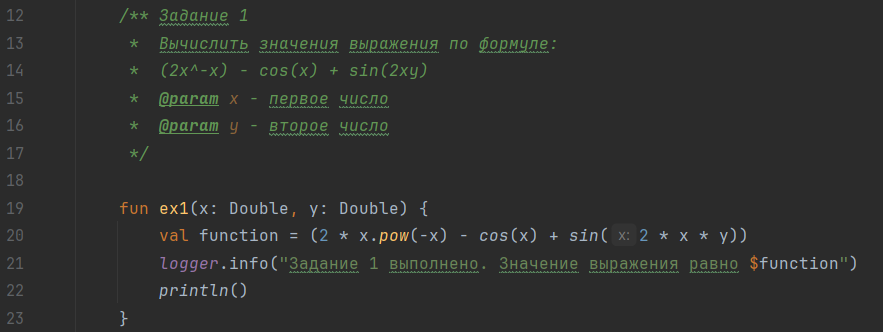


Рисунок 1 – Код задания 1

На рисунке 2 представлен результат задания 1.

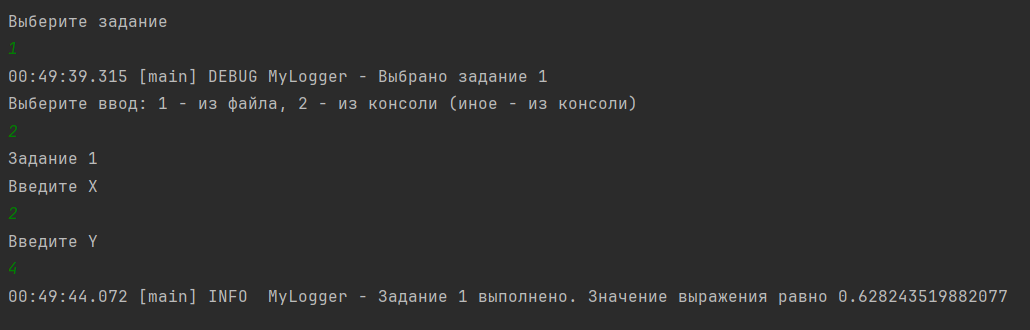


Рисунок 2 – Результат задания 1

2. Найти площадь равнобедренной трапеции с основаниями а и b и углом а при большем основании а.

На рисунке 3 представлен код задания 2.

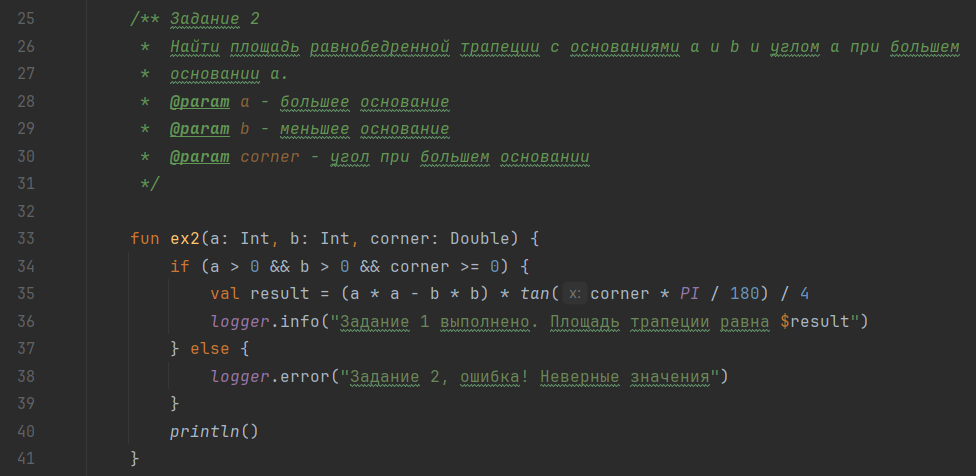


Рисунок 3 – Код задания 2

На рисунке 4 представлен результат задания 2.

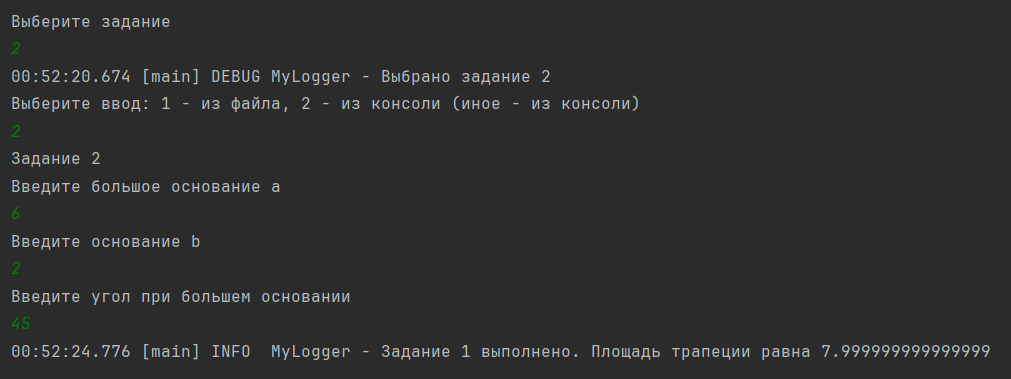


Рисунок 4 – Результат задания 2

3. Подсчитать количество отрицательных чисел среди чисел а, b, с.

На рисунке 5 представлен код задания 3.

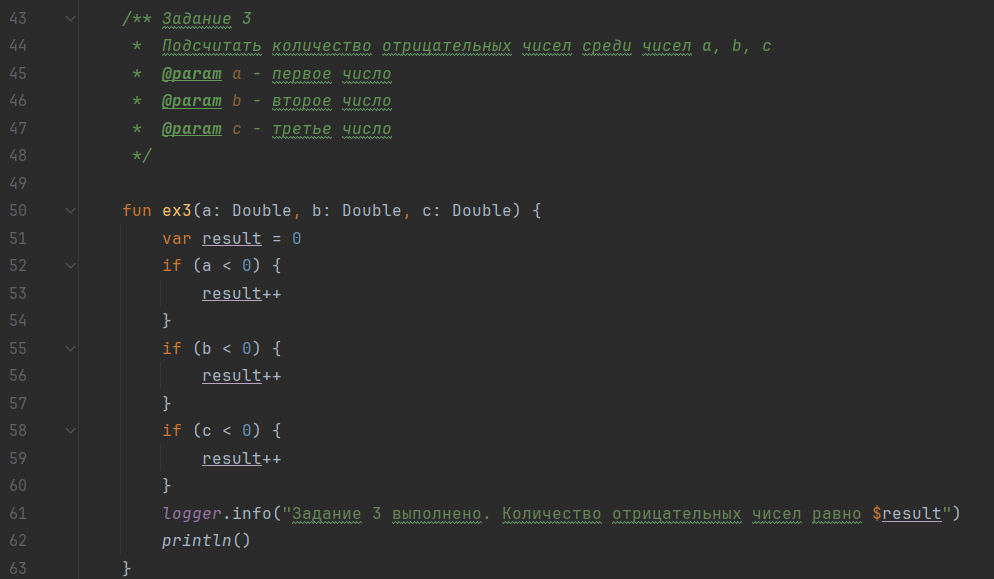


Рисунок 5 – Код задания 3

На рисунке 6 представлен результат задания 3.

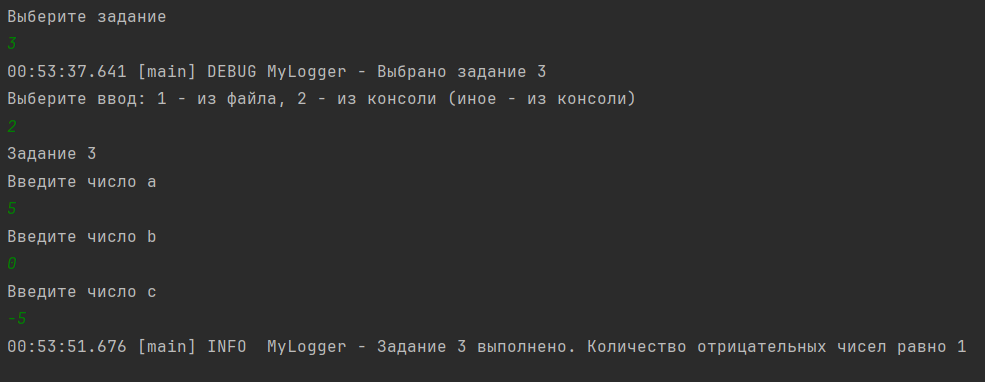


Рисунок 6 – Результат задания 3

4. Дана точка A (x, у). Определить, принадлежит ли она треугольнику с вершинами в точках (x1, у1), (х2, у2), (х3, у3).

На рисунке 7 представлен код задания 4.

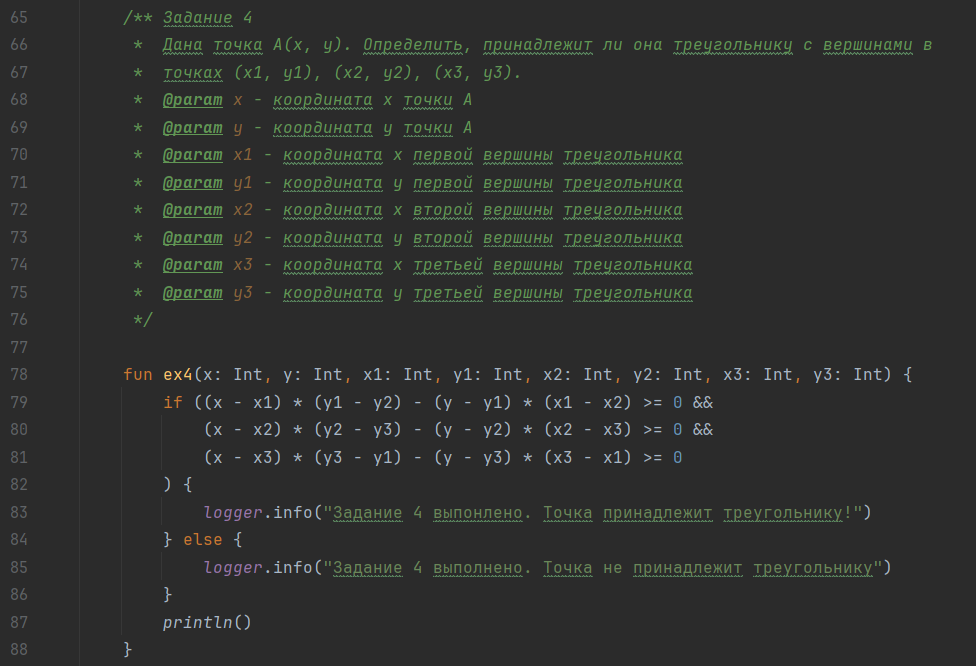


Рисунок 7 – Код задания 4

На рисунке 8 представлен результат задания 4.

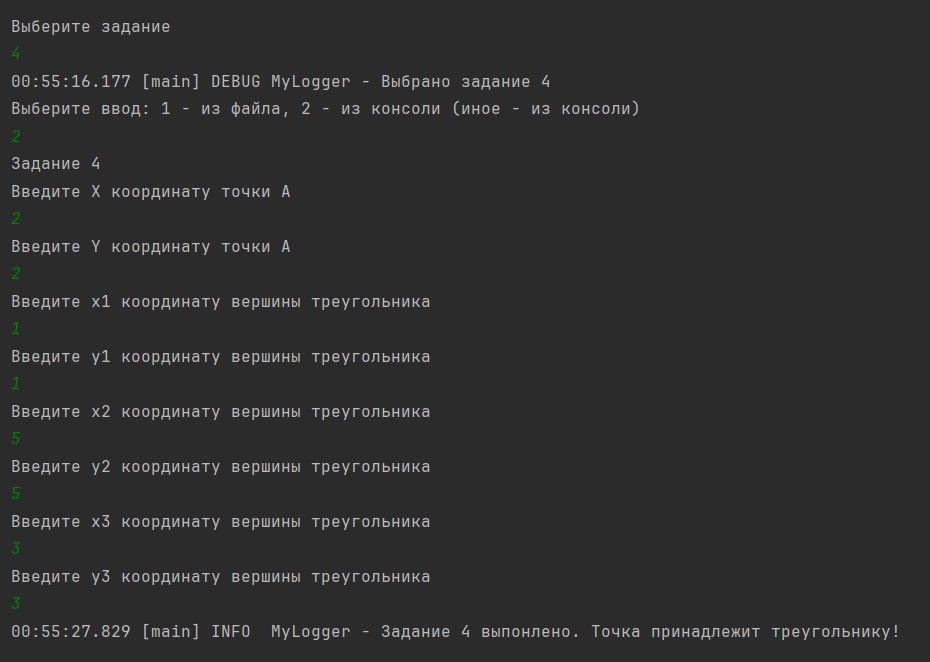


Рисунок 8 – Результат задания 4

5. Написать программу, которая по вводимому числу от 1 до 11 (номеру класса) выдает соответствующее сообщение «Привет, k-классник». Например, если k = 1, «Привет, первоклассник»; при k = 4: «Привет, четвероклассник». (switch)

На рисунке 9 представлен код задания 5.

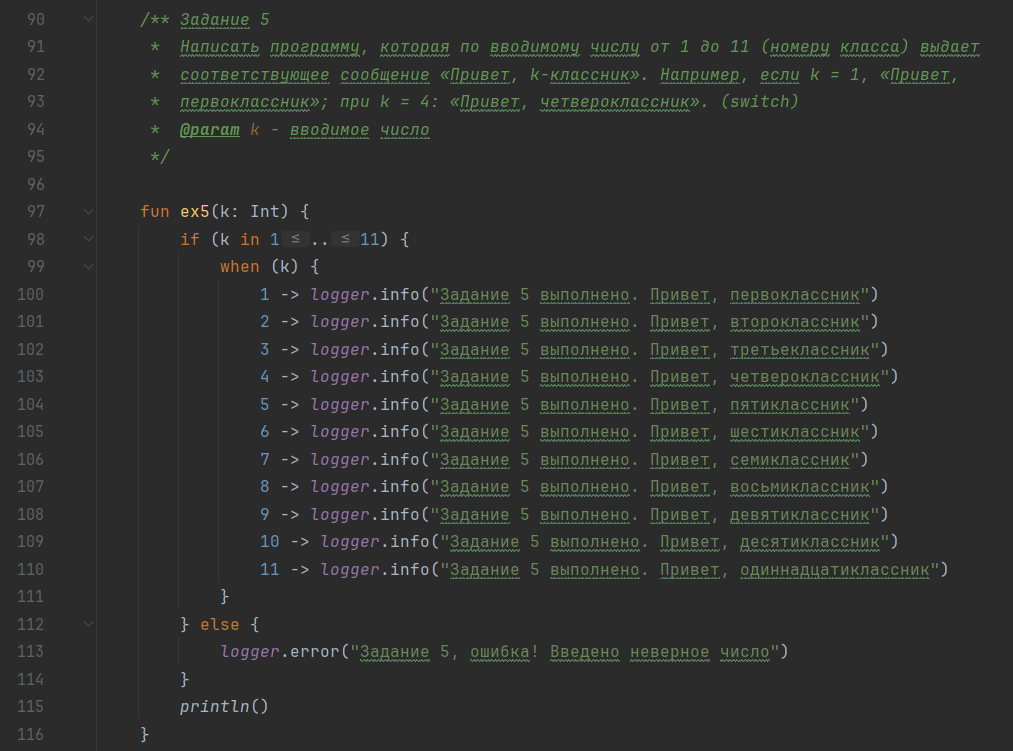


Рисунок 9 – Код задания 5

На рисунке 10 представлен результат задания 5.

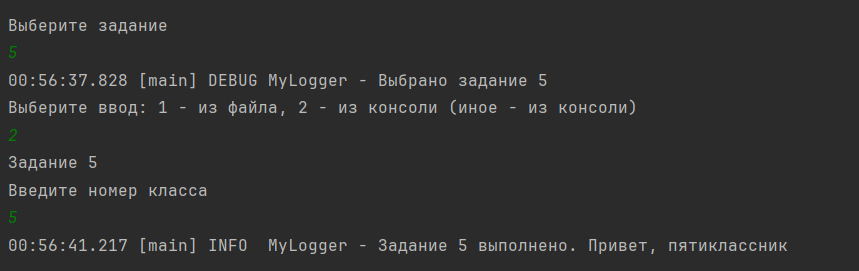


Рисунок 10 – Результат задания 5

6. Найти все делители натурального числа n.

На рисунке 11 представлен код задания 6.

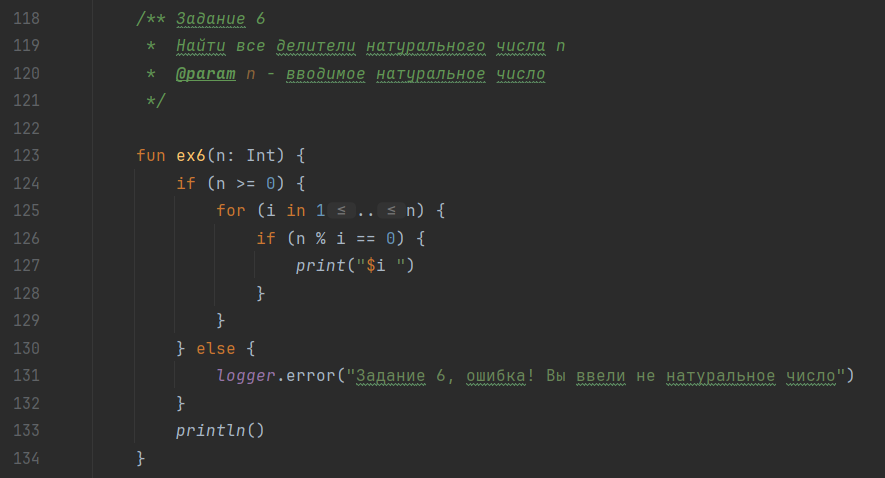


Рисунок 11 – Код задания 6

На рисунке 12 представлен результат задания 6.

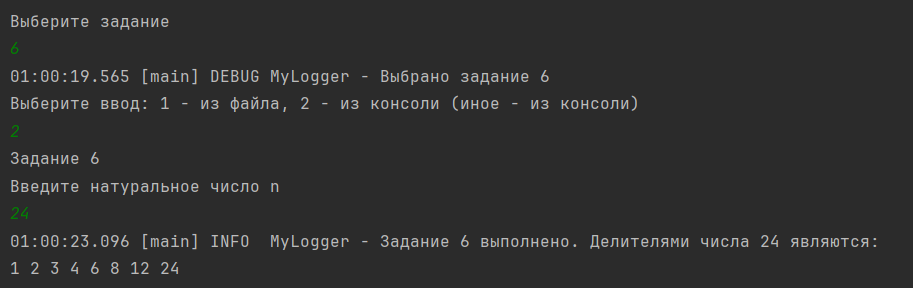
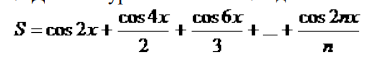


Рисунок 12 – Результат задания 6

7. Дано натуральное число n и действительное x. Найти сумму n членов ряда:



На рисунке 13 представлен код задания 7.

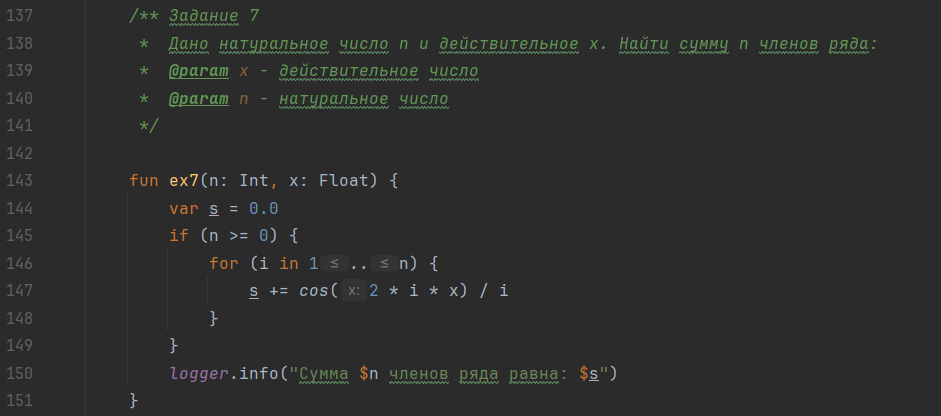


Рисунок 13 – Код задания 7

На рисунке 14 представлен результат задания 7.

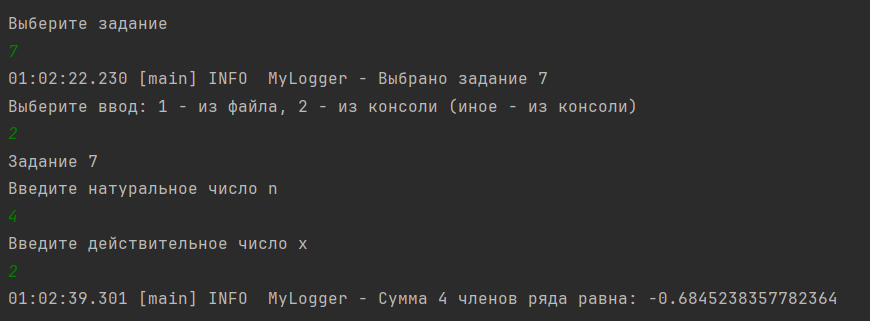


Рисунок 14 – Результат задания 7

8. Даны натуральные числа M, N, k. Написать программу для вычисления суммы целых положительных чисел, больших M, меньших N и кратных k. Полученное число вывести на экран

На рисунке 15 представлен код задания 8.

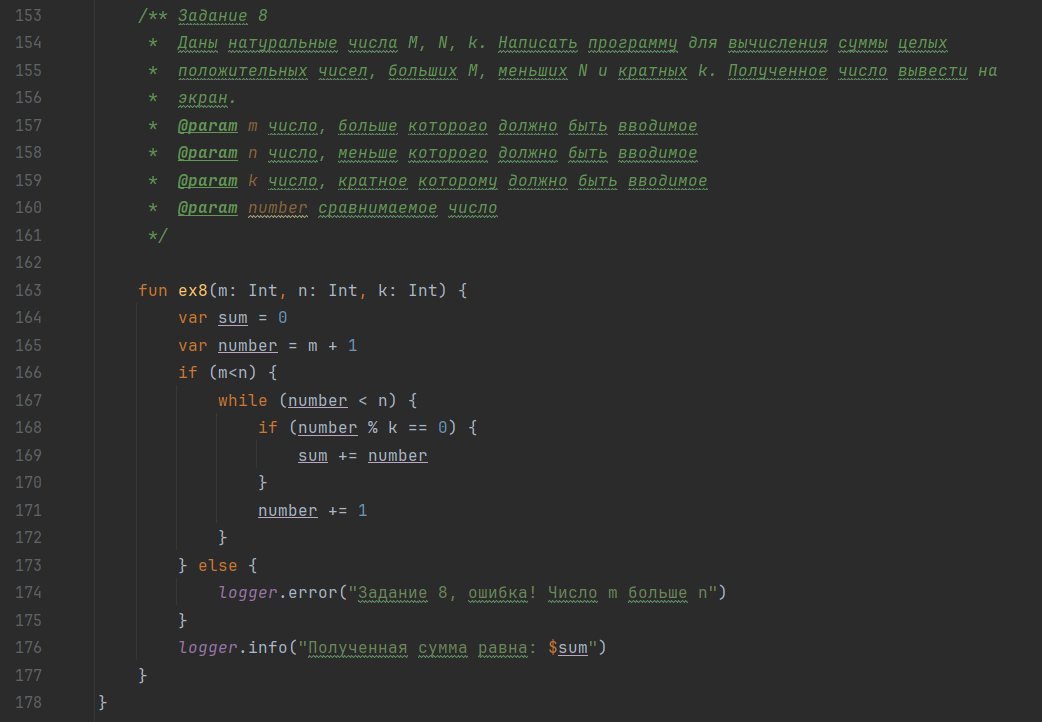


Рисунок 15 – Код задания 8

На рисунке 16 представлен результат задания 8.

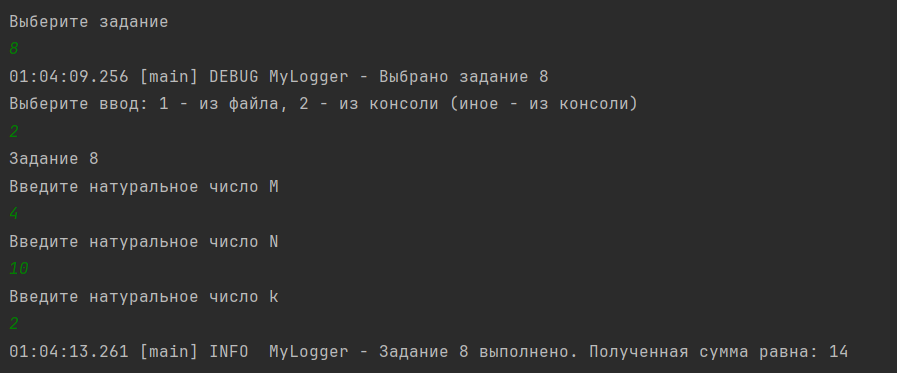


Рисунок 16 – Результат задания 8

**Вывод:** изучили основные понятия, синтаксис языка Java и общую структуру программ, получить практические навыки программирования на языке Java.

Листинг программы:

package com.org.lab3

import org.apache.logging.log4j.LogManager

import java.nio.file.Files

import java.nio.file.Paths

import org.apache.logging.log4j.Logger

val logger: Logger = LogManager.getLogger("MyLogger")

/\*\*

\* @author Naboychenko "insomnia" Nick

\* @version 1.0

\*/

fun main() {

val exmpl = Lab3()

var n: Int = -1

while (n != 0) {

println("---------------------------------------")

println("Выберите задание")

n = readln().toInt()

/\*\*

\* @see Lab3

\*/

try {

processInput(n, exmpl)

} catch (e: Exception) {

logger.fatal("Something went wrong\n${e.javaClass.name}\n${e.message}")

}

}

}

private fun processInput(n: Int, exmpl: Lab3) {

when (n) {

1 -> {

logger.debug("Выбрано задание 1")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case1(a, exmpl)

}

2 -> {

logger.debug("Выбрано задание 2")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case2(a, exmpl)

}

3 -> {

logger.debug("Выбрано задание 3")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case3(a, exmpl)

}

4 -> {

logger.debug("Выбрано задание 4")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case4(a, exmpl)

}

5 -> {

logger.debug("Выбрано задание 5")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case5(a, exmpl)

}

6 -> {

logger.debug("Выбрано задание 6")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case6(a, exmpl)

}

7 -> {

logger.info("Выбрано задание 7")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case7(a, exmpl)

}

8 -> {

logger.debug("Выбрано задание 8")

println("Выберите ввод: 1 - из файла, 2 - из консоли (иное - из консоли)")

val a = readln().toInt()

case8(a, exmpl)

}

else -> println("CrItIcAl ErRoR")

}

}

private fun case8(a: Int, exmpl: Lab3) {

println("Задание 8")

if (a == 1) {

val path = "src/main/resources/aboba8.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex8(encoded[0].toInt(), encoded[1].toInt(), encoded[2].toInt())

} else {

println("Введите натуральное число M")

val m: Int = readln().toInt()

println("Введите натуральное число N")

val n: Int = readln().toInt()

println("Введите натуральное число k")

val k: Int = readln().toInt()

exmpl.ex8(m, n, k)

}

}

private fun case7(a: Int, exmpl: Lab3) {

println("Задание 7")

if (a == 1) {

val path = "src/main/resources/aboba7.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex7(encoded[0].toInt(), encoded[1].toFloat())

} else {

println("Введите натуральное число n")

val n: Int = readln().toInt()

println("Введите действительное число x")

val x: Float = readln().toFloat()

exmpl.ex7(n, x)

}

}

private fun case6(a: Int, exmpl: Lab3) {

println("Задание 6")

if (a == 1) {

val path = "src/main/resources/aboba6.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex6(encoded[0].toInt())

} else {

println("Введите натуральное число n")

val n: Int = readln().toInt()

exmpl.ex6(n)

}

}

private fun case5(a: Int, exmpl: Lab3) {

println("Задание 5")

if (a == 1) {

val path = "src/main/resources/aboba5.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex5(encoded[0].toInt())

} else {

println("Введите номер класса")

val k: Int = readln().toInt()

exmpl.ex5(k)

}

}

private fun case4(a: Int, exmpl: Lab3) {

println("Задание 4")

if (a == 1) {

val path = "src/main/resources/aboba4.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex4(

encoded[0].toInt(), encoded[1].toInt(), encoded[2].toInt(),

encoded[3].toInt(), encoded[4].toInt(), encoded[5].toInt(),

encoded[6].toInt(), encoded[7].toInt()

)

} else {

println("Введите X координату точки A")

val x: Int = readln().toInt()

println("Введите Y координату точки A")

val y: Int = readln().toInt()

println("Введите x1 координату вершины треугольника")

val x1: Int = readln().toInt()

println("Введите y1 координату вершины треугольника")

val y1: Int = readln().toInt()

println("Введите x2 координату вершины треугольника")

val x2: Int = readln().toInt()

println("Введите y2 координату вершины треугольника")

val y2: Int = readln().toInt()

println("Введите x3 координату вершины треугольника")

val x3: Int = readln().toInt()

println("Введите y3 координату вершины треугольника")

val y3: Int = readln().toInt()

exmpl.ex4(x, y, x1, y1, x2, y2, x3, y3)

}

}

private fun case3(a: Int, exmpl: Lab3) {

println("Задание 3")

if (a == 1) {

val path = "src/main/resources/aboba3.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex3(encoded[0].toDouble(), encoded[1].toDouble(), encoded[2].toDouble())

} else {

println("Введите число a")

val a: Double = readln().toDouble()

println("Введите число b")

val b: Double = readln().toDouble()

println("Введите число c")

val c: Double = readln().toDouble()

exmpl.ex3(a, b, c)

}

}

private fun case2(a: Int, exmpl: Lab3) {

println("Задание 2")

if (a == 1) {

val path = "src/main/resources/aboba2.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex2(encoded[0].toInt(), encoded[1].toInt(), encoded[2].toDouble())

} else {

println("Введите большое основание a")

val a: Int = readln().toInt()

println("Введите основание b")

val b: Int = readln().toInt()

println("Введите угол при большем основании")

val corner: Double = readln().toDouble()

exmpl.ex2(a, b, corner)

}

}

private fun case1(a: Int, exmpl: Lab3) {

println("Задание 1")

if (a == 1) {

val path = "src/main/resources/aboba1.txt"

val encoded = Files.readAllLines(Paths.get(path))

exmpl.ex1(encoded[0].toDouble(), encoded[1].toDouble())

} else {

println("Введите X")

val x: Double = readln().toDouble()

println("Введите Y")

val y: Double = readln().toDouble()

exmpl.ex1(x, y)

}

}

package com.org.lab3

import kotlin.math.\*

class Lab3 {

/\*\*

\* @author Naboychenko "insomnia" Nick

\* @version 1.0

\*/

/\*\* Задание 1

\* Вычислить значения выражения по формуле:

\* (2x^-x) - cos(x) + sin(2xy)

\* @param x - первое число

\* @param y - второе число

\*/

fun ex1(x: Double, y: Double) {

val function = (2 \* x.pow(-x) - cos(x) + sin(2 \* x \* y))

logger.info("Задание 1 выполнено. Значение выражения равно $function")

println()

}

/\*\* Задание 2

\* Найти площадь равнобедренной трапеции с основаниями а и b и углом а при большем

\* основании а.

\* @param a - большее основание

\* @param b - меньшее основание

\* @param corner - угол при большем основании

\*/

fun ex2(a: Int, b: Int, corner: Double) {

if (a > 0 && b > 0 && corner >= 0) {

val result = (a \* a - b \* b) \* tan(corner \* PI / 180) / 4

logger.info("Задание 1 выполнено. Площадь трапеции равна $result")

} else {

logger.error("Задание 2, ошибка! Неверные значения")

}

println()

}

/\*\* Задание 3

\* Подсчитать количество отрицательных чисел среди чисел а, b, с

\* @param a - первое число

\* @param b - второе число

\* @param c - третье число

\*/

fun ex3(a: Double, b: Double, c: Double) {

var result = 0

if (a < 0) {

result++

}

if (b < 0) {

result++

}

if (c < 0) {

result++

}

logger.info("Задание 3 выполнено. Количество отрицательных чисел равно $result")

println()

}

/\*\* Задание 4

\* Дана точка A(x, у). Определить, принадлежит ли она треугольнику с вершинами в

\* точках (x1, у1), (х2, у2), (х3, у3).

\* @param x - координата x точки А

\* @param y - координата y точки A

\* @param x1 - координата x первой вершины треугольника

\* @param y1 - координата y первой вершины треугольника

\* @param x2 - координата x второй вершины треугольника

\* @param y2 - координата y второй вершины треугольника

\* @param x3 - координата x третьей вершины треугольника

\* @param y3 - координата y третьей вершины треугольника

\*/

fun ex4(x: Int, y: Int, x1: Int, y1: Int, x2: Int, y2: Int, x3: Int, y3: Int) {

if ((x - x1) \* (y1 - y2) - (y - y1) \* (x1 - x2) >= 0 &&

(x - x2) \* (y2 - y3) - (y - y2) \* (x2 - x3) >= 0 &&

(x - x3) \* (y3 - y1) - (y - y3) \* (x3 - x1) >= 0

) {

logger.info("Задание 4 выпонлено. Точка принадлежит треугольнику!")

} else {

logger.info("Задание 4 выполнено. Точка не принадлежит треугольнику")

}

println()

}

/\*\* Задание 5

\* Написать программу, которая по вводимому числу от 1 до 11 (номеру класса) выдает

\* соответствующее сообщение «Привет, k-классник». Например, если k = 1, «Привет,

\* первоклассник»; при k = 4: «Привет, четвероклассник». (switch)

\* @param k - вводимое число

\*/

fun ex5(k: Int) {

if (k in 1..11) {

when (k) {

1 -> logger.info("Задание 5 выполнено. Привет, первоклассник")

2 -> logger.info("Задание 5 выполнено. Привет, второклассник")

3 -> logger.info("Задание 5 выполнено. Привет, третьеклассник")

4 -> logger.info("Задание 5 выполнено. Привет, четвероклассник")

5 -> logger.info("Задание 5 выполнено. Привет, пятиклассник")

6 -> logger.info("Задание 5 выполнено. Привет, шестиклассник")

7 -> logger.info("Задание 5 выполнено. Привет, семиклассник")

8 -> logger.info("Задание 5 выполнено. Привет, восьмиклассник")

9 -> logger.info("Задание 5 выполнено. Привет, девятиклассник")

10 -> logger.info("Задание 5 выполнено. Привет, десятиклассник")

11 -> logger.info("Задание 5 выполнено. Привет, одиннадцатиклассник")

}

} else {

logger.error("Задание 5, ошибка! Введено неверное число")

}

println()

}

/\*\* Задание 6

\* Найти все делители натурального числа n

\* @param n - вводимое натуральное число

\*/

fun ex6(n: Int) {

if (n >= 0) {

logger.info("Задание 6 выполнено. Делителями числа $n являются:")

for (i in 1..n) {

if (n % i == 0) {

print("$i ")

}

}

} else {

logger.error("Задание 6, ошибка! Вы ввели не натуральное число")

}

println()

}

/\*\* Задание 7

\* Дано натуральное число n и действительное x. Найти сумму n членов ряда:

\* @param x - действительное число

\* @param n - натуральное число

\*/

fun ex7(n: Int, x: Float) {

var s = 0.0

if (n >= 0) {

for (i in 1..n) {

s += cos(2 \* i \* x) / i

}

}

logger.info("Сумма $n членов ряда равна: $s")

}

/\*\* Задание 8

\* Даны натуральные числа M, N, k. Написать программу для вычисления суммы целых

\* положительных чисел, больших M, меньших N и кратных k. Полученное число вывести на

\* экран.

\* @param m число, больше которого должно быть вводимое

\* @param n число, меньше которого должно быть вводимое

\* @param k число, кратное которому должно быть вводимое

\* @param number сравнимаемое число

\*/

fun ex8(m: Int, n: Int, k: Int) {

var sum = 0

var number = m + 1

if (m<n) {

while (number < n) {

if (number % k == 0) {

sum += number

}

number += 1

}

} else {

logger.error("Задание 8, ошибка! Число m больше n")

}

logger.info("Задание 8 выполнено. Полученная сумма равна: $sum")

}

}

**LOG4J PROPERTIES**

status = error

name = PropertiesConfig

#Make sure to change log file path as per your need

property.filename = C:\\logs\\debug.log

filters = threshold

filter.threshold.type = ThresholdFilter

filter.threshold.level = debug

appenders = rolling

appender.rolling.type = RollingFile

appender.rolling.name = RollingFile

appender.rolling.fileName = ${filename}

appender.rolling.filePattern = C:\\logs\\Previous\\debug-backup-%d{MM-dd-yy-HH-mm-ss}-%i.log.gz

appender.rolling.layout.type = PatternLayout

appender.rolling.layout.pattern = %d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n

appender.rolling.policies.type = Policies

appender.rolling.policies.time.type = TimeBasedTriggeringPolicy

appender.rolling.policies.time.interval = 1

appender.rolling.policies.time.modulate = true

appender.rolling.policies.size.type = SizeBasedTriggeringPolicy

appender.rolling.policies.size.size=10MB

appender.rolling.strategy.type = DefaultRolloverStrategy

appender.rolling.strategy.max = 20

loggers = rolling

#Make sure to change the package structure as per your application

logger.rolling.name = MyLogger

logger.rolling.level = debug

logger.rolling.additivity = true

logger.rolling.appenderRef.rolling.ref = RollingFile