**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая кибернетика и информационные технологии»

**Отчет по лабораторной работе №8**

по дисциплине «Информационные технологии и программирование»

на тему:

Улучшенный веб-сканер

Выполнил: студент группы БФИ2101

Иванов Глеб

Проверил преподаватель:

Мосева Марина Сергеевна

Москва, 2022

**Оглавление**

[Поставленная задача 3](#_Toc122875905)

[Скриншоты выполненной работы 4](#_Toc122875906)

[Код 5](#_Toc122875907)

[Вывод 11](#_Toc122875908)

[Список источников 12](#_Toc122875909)

**Поставленная задача**

1. Реализовать многопоточность в нашем методе поиска сайтов.

**Скриншоты выполненной работы**

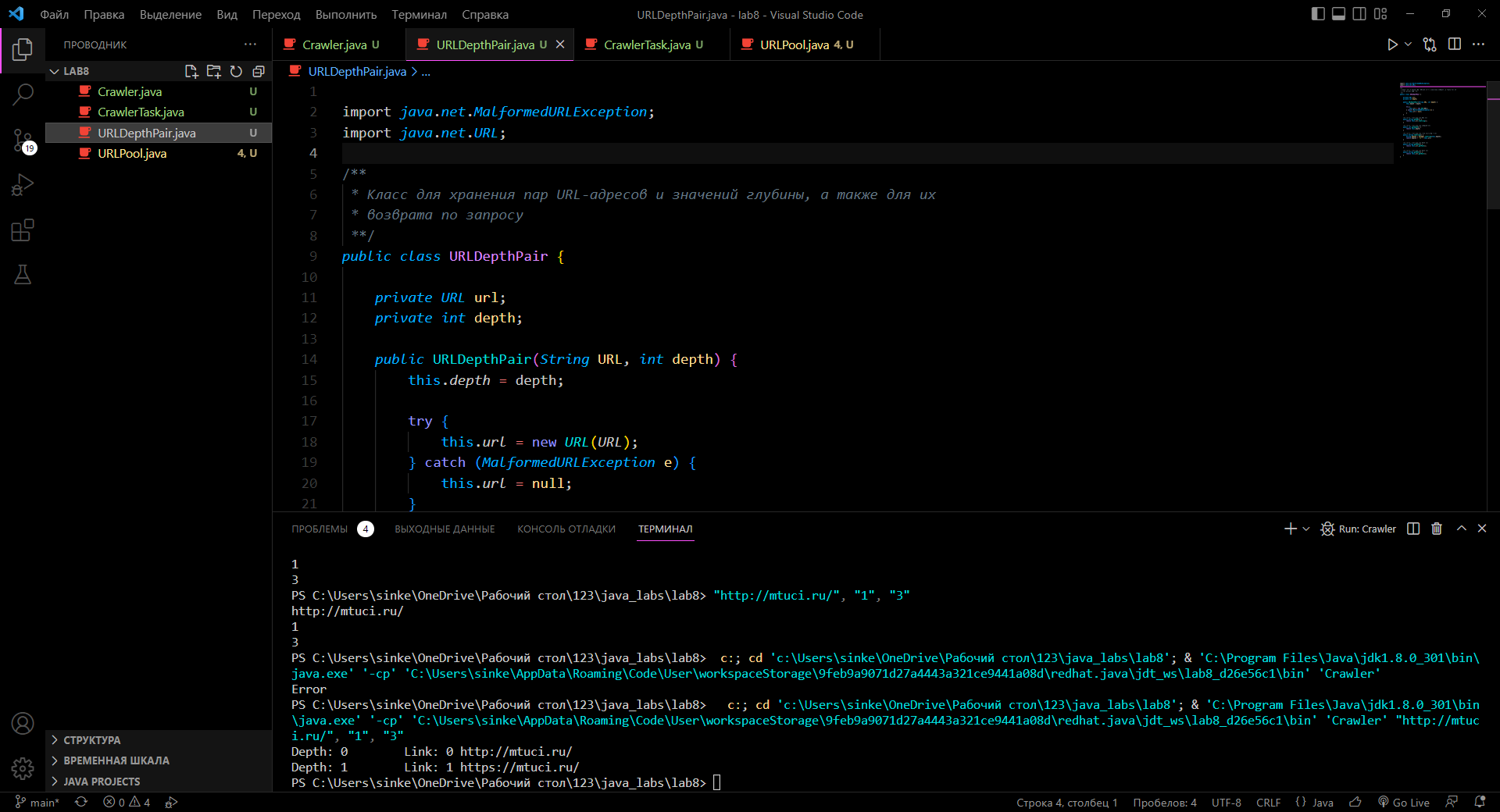
****

Рисунок 1 – Работа нашей программы

**Код**

import *java*.*util*.*LinkedList*;

/\*\* *Класс, который перемещается по веб-страницам и ищет URL-адреса \**\*/

*public* *class* Crawler {

*public* *static* *void* main(*String*[] args) {

        // *String[] args = new String[]{"http://mtuci.ru/", "1", "3"};*

        if (args.*length* == 3 && *checkNum*(*args*[1]) && *checkNum*(*args*[2])) {

*String* surl = args[0];

*int* treadNum = Integer.parseInt(args[2]);

*URLPool* pool = new *URLPool*(Integer.*parseInt*(*args*[1]));

            pool.addPair(new *URLDepthPair*(*surl*,0));

            for (*int* i = 0; i < treadNum; i++) {

*CrawlerTask* c = new *CrawlerTask*(*pool*);

*Thread* t = new *Thread*(*c*);

                t.start();

            }

            while (pool.getWait() != treadNum) {

                try {

                    Thread.sleep(500);

                }

                catch (*InterruptedException* e) {

                    System.*out*.println("Ignoring Interrupted Exception");

                }

            }

            try {

*showResult*(pool.*getResult*());

            }

            catch (*NullPointerException* e) {

                System.*out*.println("Not Link");

            }

            System.exit(0);

        } else

            System.*out*.println("Error");

    }

*public* *static* *void* showResult(*LinkedList*<*URLDepthPair*> resultLink) {

        for (*URLDepthPair* c : resultLink)

            System.*out*.println("Depth: " + c.getDepth() + "\tLink: " + c.toString());

    }

/\*\* *Вспомогательная функция для проверки, является ли строка числом \**\*/

*public* *static* *boolean* checkNum(*String* num) {

*boolean* ok = true;

        for (*int* i = 0; i < num.length() && ok; i++)

            ok = Character.isDigit(num.charAt(i));

        return ok;

    }

}

Код файла Crawel.java

import *java*.*io*.*BufferedReader*;

import *java*.*io*.*IOException*;

import *java*.*io*.*InputStreamReader*;

import *java*.*io*.*PrintWriter*;

import *java*.*net*.*Socket*;

*public* *class* CrawlerTask *implements* Runnable {

*URLPool* upool;

*public* *static* *final* *String* URL\_PREFIX = "<a href=\"http";

    @*Override*

*public* *void* run() {

        do {

            try {

*URLDepthPair* pair = upool.getPair();

*int* currDepth = pair.getDepth();

                try {

*Socket* s = new *Socket*(pair.*getHost*(),80);

                    s.setSoTimeout(1000);

*PrintWriter* out = new *PrintWriter*(s.*getOutputStream*(), *true*);

*BufferedReader* in = new *BufferedReader*(new *InputStreamReader*(s.*getInputStream*()));

*request*(*out*, *pair*);

*String* line;

                    while ((line = in.readLine()) != null) {

                        if (line.contains(URL\_PREFIX))

*buildNewUrl*(*line*, *currDepth*, *upool*);

                    }

                    s.close();

                } catch (*IOException* e) {

                }

            } catch (*NullPointerException* e) {

            }

        } while (true);

    }

*public* CrawlerTask(*URLPool* pool) {

        this.*upool* = pool;

    }

*public* *static* *void* request(*PrintWriter* out, *URLDepthPair* pair) {

        out.println("GET " + pair.getPath() + " HTTP/1.1");

        out.println("Host: " + pair.getHost());

        out.println("Connection: close");

        out.println();

        out.flush();

    }

*public* *static* *void* buildNewUrl(*String* str, *int* depth, *URLPool* pool) {

        try {

*String* currentLink = str.substring(str.indexOf(URL\_PREFIX) +

                    URL\_PREFIX.indexOf("\"") + 1,

                    str.indexOf("\"", str.indexOf(URL\_PREFIX) +

                            URL\_PREFIX.indexOf("\"") + 2));

            pool.addPair(new *URLDepthPair*(*currentLink*, *depth* +1));

        } catch (*StringIndexOutOfBoundsException* e) {

        }

    }

}

Код файла CrawelTask.java

import *java*.*net*.*MalformedURLException*;

import *java*.*net*.*URL*;

/\*\*

*\* Класс для хранения пар URL-адресов и значений глубины, а также для их*

*\* возврата по запросу*

*\**\*/

*public* *class* URLDepthPair {

*private* *URL* url;

*private* *int* depth;

*public* URLDepthPair(*String* URL, *int* depth) {

        this.*depth* = depth;

        try {

            this.*url* = new *URL*(*URL*);

        } catch (*MalformedURLException* e) {

            this.*url* = null;

        }

    }

/\*\* *Метод, возвращающий URL \**\*/

*public* *String* getURL() {

        return this.*url*.toString();

    }

/\*\* *Метод, возвращающий глубину \**\*/

*public* *int* getDepth() {

        return this.*depth*;

    }

/\*\* *Метод, возвращающий содержимое пары \**\*/

*public* *String* toString() {

*String* depths = Integer.toString(this.*depth*);

        return depths + '\t' + this.*url*;

    }

/\*\* *Метод, возвращающий Path \**\*/

*public* *String* getPath() {

        return this.*url*.getPath();

    }

/\*\* *Метод, возвращающий Host \**\*/

*public* *String* getHost() {

        return this.*url*.getHost();

    }

}

Код файла URLDepthPair.java

import *java*.*util*.*LinkedList*;

*public* *class* URLPool {

*LinkedList*<*URLDepthPair*> urlFind = new *LinkedList*();

*LinkedList*<*URLDepthPair*> urlRes = new *LinkedList*();

*int* maxDepth;

*int* waitN;

*public* URLPool(*int* maxDepth) {

        this.*maxDepth* = maxDepth;

        waitN = 0;

    }

/\*\*

*\* Метод для использования wait(), чтобы потоки сканера могли ожидать появления*

*\* новых URL-адресов.*

*\**\*/

*public* *synchronized* *URLDepthPair* getPair() {

        while (urlFind.size() == 0) {

            waitN++;

            try {

*wait*();

            } catch (*InterruptedException* e) {

                System.*out*.println("Ignoring Interrupted Exception");

            }

            waitN--;

        }

*URLDepthPair* nextPair = urlFind.getFirst();

        urlFind.removeFirst();

        return nextPair;

    }

/\*\*

*\* Метод для использования notify(), чтобы потоки сканера могли ожидать*

*\* появления новых URL-адресов.*

*\**\*/

*public* *synchronized* *void* addPair(*URLDepthPair* pair) {

        if (*check*(*urlRes*, *pair*)) {

            urlRes.add(pair);

            if (pair.getDepth() < maxDepth) {

                urlFind.add(pair);

*notify*();

            }

        }

    }

*public* *synchronized* *int* getWait() {

        return waitN;

    }

/\*\* *Метод для проверки на совпадение пар \**\*/

*private* *static* *boolean* check(*LinkedList*<*URLDepthPair*> resultLink, *URLDepthPair* pair) {

*boolean* chekin = true;

        for (*URLDepthPair* c : resultLink)

            if (c.toString().equals(pair.toString()))

                chekin = false;

        return chekin;

    }

*public* *LinkedList*<*URLDepthPair*> getResult() {

        return urlRes;

    }

}

Код файла URLPool.java

**Вывод**

В ходе данной лабораторной работы я смог усовершенствовать алгоритм поиска сайтов на нашей веб-странице.

**Список источников**

1. Камаев В.А., Костерин В.В. Технологии программирования. М.: Высшая школа, 2006.

2. Жоголев Е.А.Технология программирования. – М.: Научный мир, 2004.