**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра Информатики

**Отчет по лабораторной работе №7**

по предмету «Кроссплатформенные технологии программирования»

на тему:

«Веб-сканер»

Выполнил: студент группы БВТ 1802

Ганюшкина Елизавета Сергеевна

Руководитель:

Полянцева Ксения Андреевна

Москва 2020

**Цель работы**

В лабораторной работе необходимо реализовать элементарный веб-сканер. Сканер будет автоматически загружать веб-страницы из Интернета, искать новые ссылки на этих страницах и повторять их.

**Задания**

1. Программа должна принимать в командной строке два параметра:

* Строку, которая представляет собой URL-адрес, с которого можно начать просмотр страницы.
* Положительное целое число, которое является максимальной глубиной поиска (см. ниже);

1. Программа должна хранить URL-адрес в виде строки вместе с его глубиной (которая для начала будет равна 0);
2. Программа должна подключиться к указанному сайту в URL-адресе на порт 80 с использованием сокета и запросить указанную веб-страницу;
3. Программа должна проанализировать возвращаемый текст, построчно для любых подстрок;
4. Программа должна вывести все посещенные URL-страницы вместе с их глубиной поиска;
5. Добавить код для добавления сайтов в необработанный список, если они не были просмотрены ранее;
6. Расширить возможности по поиску гиперссылок сканера, используя поиск по регулярным выражениям в собранных данных.

**Коды программ**

*Crawler.java*

import java.io.\*;

import java.net.\*;

import java.util.LinkedList;

public class Crawler {

static LinkedList <URLDepthPair> findLink = new LinkedList <URLDepthPair>();

static LinkedList <URLDepthPair> viewedLink = new LinkedList <URLDepthPair>();

public static void showResult(LinkedList<URLDepthPair> viewedLink) {

for (URLDepthPair c : viewedLink)

System.out.println("Глубина : "+c.getDepth() + "\tСсылка : "+c.getURL());}

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

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

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

out.println("Connection: close");

out.println();

out.flush();}

public static void Process(String pair, int maxDepth) throws IOException {

findLink.add(new URLDepthPair(pair, 0));

while (!findLink.isEmpty()) {

URLDepthPair currentPair = findLink.removeFirst();

if (currentPair.depth<maxDepth){

Socket my\_socket = new Socket(currentPair.getHost(), 80);

my\_socket.setSoTimeout(100000);

BufferedReader in = new BufferedReader(new InputStreamReader(my\_socket.getInputStream()));

PrintWriter out = new PrintWriter(my\_socket.getOutputStream(), true);

request(out, currentPair);

String line;

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

if (line.indexOf(currentPair.URL\_PREFIX)!=-1) {

StringBuilder currentLink = new StringBuilder();

for (int i = line.indexOf(currentPair.URL\_PREFIX) + 9; line.charAt(i) != '"'; i++) {

currentLink.append(line.charAt(i));}

URLDepthPair newPair = new URLDepthPair(currentLink.toString(), currentPair.depth + 1);

if (currentPair.check(findLink, newPair) && currentPair.check(viewedLink, newPair) && !currentPair.URL.equals(newPair.URL))

findLink.add(newPair);}}

my\_socket.close();}

viewedLink.add(currentPair);}

showResult(viewedLink);}

public static void main(String[] args) {

String[] arg = new String[]{"http://htmlbook.ru/","3"};

try {

Process(arg[0], Integer.parseInt(arg[1]));

} catch (NumberFormatException | IOException e) {

System.out.println("usage: java Crawler <URL><depth>");}}}

*URLDepthPair.java*

import java.util.LinkedList;

import java.net.MalformedURLException;

import java.net.URL;

public class URLDepthPair {

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

public String URL;

public int depth;

public URLDepthPair (String URL, int depth){

this.URL=URL;

this.depth=depth;}

public String getHost() throws MalformedURLException {

URL host = new URL(URL);

return host.getHost();}

public String getPath() throws MalformedURLException {

URL path = new URL(URL);

return path.getPath();}

public int getDepth() {

return depth;}

public String getURL() {

return URL;}

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

boolean isAlready = true;

for (URLDepthPair c : resultLink)

if (c.getURL().equals(pair.getURL()))

isAlready=false;

return isAlready;}}

**Результат работы кода**

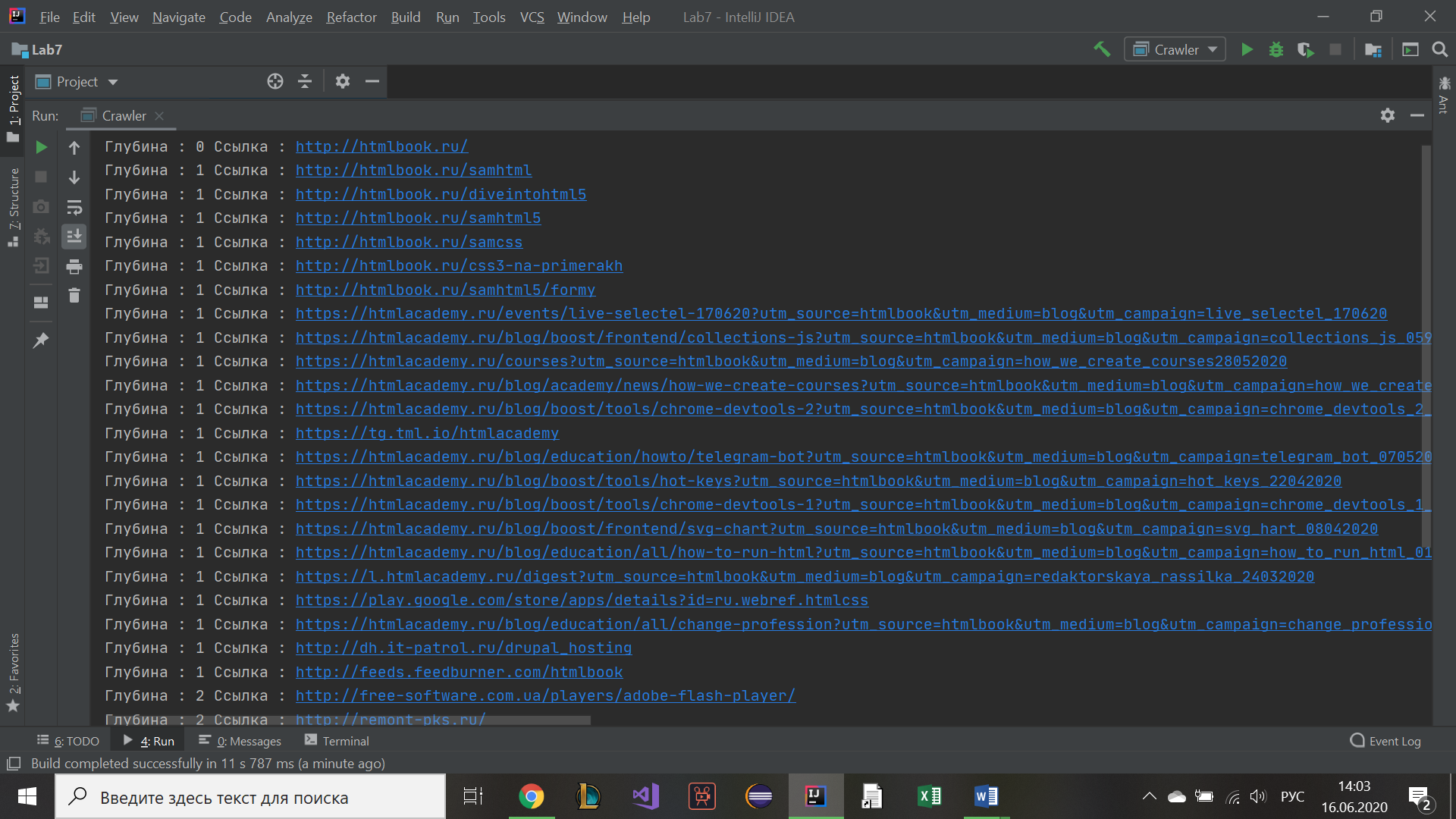


Рисунок 1 - Результат работы программы

**Вывод**

В данной лабораторной работе был разработан: веб-сканнер, который принимал в командной строке два параметра, хранит URL-адреса в виде строки вместе с его глубиной, подключается к указанному сайту с использованием сокета и запрашивает веб-страницу. Программа проанализировала возвращаемый текст, вывела все посещения и глубину поиска.