**Федеральное агентство связи**

**Ордена трудового Красного Знамени федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

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

Кафедра Математическая кибернетика и информационные технологии

Лабораторная работа № 7

Выполнил: студент группы БФИ2002

Ковачев В.Е.

Проверила:

Москва, 2021

**Задачи:**

1. Программа должна принимать в командной строке два параметра:

1) Строку, которая представляет собой URL-адрес, с которого можно начать просмотр страницы.

2) Положительное целое число, которое является максимальной глубиной поиска (см. ниже) Если указаны некорректные аргументы, программа должна немедленно остановиться и выдать сообщение об используемых аргументах, например: usage: java Crawler

2. Программа должна хранить URL-адрес в виде строки вместе с его глубиной (которая для начала будет равна 0). Вам будет необходимо создать класс для представления пар [URL, depth].

3. Программа должна подключиться к указанному сайту в URL-адресе на порт 80 с использованием сокета (см. ниже) и запросить указанную вебстраницу.

4. Программа должна проанализировать возвращаемый текст, построчно для любых подстрок, имеющих формат: [Найденные URL-адреса должны быть сохранены в паре с новым значением глубины в LinkedList (URL, depth) (подробнее о LinkedLists см. ниже). Новое значение глубины должно быть больше, чем значение глубины URL-адреса, соответствующего анализируемой странице. 5. Далее программа должна закрыть соединение сокета с хостом. 6. Программа должна повторять шаги с 3 по 6 для каждого нового URL-адреса, если глубина, соответствующая URL-адресу, меньше максимальной. Обратите внимание, что при извлечении и поиске определенного URL-адреса глубина поиска увеличивается на 1. Если глубина URL-адреса достигает максимальной глубины (или больше), не извлекайте и не ищите эту веб-страницу. 7. Наконец, программа должна вывести все посещенные URLстраницы вместе с их глубиной поиска](%5bлюбой_URL-адрес_начинающийся_с_http:/%5d)

**Ход работы:**

**Для выполнения лабораторной работы был разработан класс Crawler:**

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("Depth : "+c.getDepth() + "\tLink : "+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(1000);

try {

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 && line.indexOf('"') != -1) {

StringBuilder currentLink = new StringBuilder();

int i = line.indexOf(currentPair.URL\_PREFIX);

while (line.charAt(i) != '"' && line.charAt(i) != ' ') {

if (line.charAt(i) == '<') {

currentLink.deleteCharAt(currentLink.length() - 1);

break;

}

else {

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

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();

} catch (SocketTimeoutException e) {

my\_socket.close();

}

}

viewedLink.add(currentPair);

}

showResult(viewedLink);

}

public static void main(String[] args) {

String[] arg = new String[]{"https://natribu.org/ru","4"};

try {

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

} catch (NumberFormatException | IOException e) {

System.out.println("usage: java crawler " + arg[0] + " " + arg[1]);

}

}

}

**А также был реализован второй класс URLDepthPair для работы со ссылками:**

import java.util.LinkedList;

import java.net.MalformedURLException;

import java.net.URL;

public class URLDepthPair {

public static final String URL\_PREFIX = "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 программы**

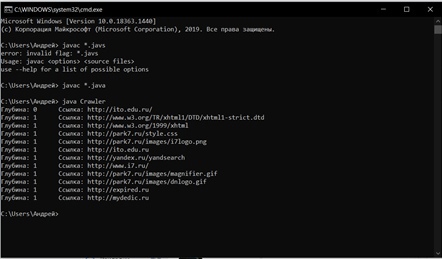


Рисунок №1- Результат работы программы

**Вывод**

В результате выполнения программы мы реализовали веб-сканер на языке java. Мы научились использовать веб-сканер, который автоматически загружает веб-страницы из Интернета, ищет новые ссылки на этих страницах и повторяет их.

**Список использованной литературы**

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

2006.

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