**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

Отчет по лабораторной работе №7

по дисциплине «Объектно-ориентированное программирование»

Выполнил: студент группы БВТ2002

ФИО: Мосева Алеся Сергеевна

Проверил:Полянцева Ксения Андреевна

Москва, 2020

**Цель работы**: необходимо будет реализовать элементарный веб-сканер.

**Код программы:**

import java.net.MalformedURLException;

import java.net.URL;

import java.util.LinkedList;

//класс, который хранит url и глубину, на которой находится этот сайт

public class URLDepthPair {

//поддерживаемый протокол

public final static String URL\_PREFIX = "http://";

//url адрес

public String url;

//глубина, на которой находится url

public int depth;

public URLDepthPair (String url, int depth) {

this.url = url;

this.depth = depth;

}

public int getDepth() {

return depth;

}

public String getUrl() {

return url;

}

public void setDepth(int depth) {

this.depth = depth;

}

public void setUrl(String url) {

this.url = url;

}

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();

}

@Override

public String toString() {

return "URLDepthPair{" +

"url='" + url + '\'' +

", depth=" + depth +

'}';

}

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

boolean isAlready = true;

for (URLDepthPair c : resultLink)

if (c.getUrl().equals(pair.getUrl()))

isAlready=false;

return isAlready;

}

}

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 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 showResult(LinkedList<URLDepthPair> list){

for (URLDepthPair c:viewedLink){

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

}

}

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

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

while (!findLink.isEmpty()) {

URLDepthPair current = findLink.removeFirst();

if (current.getDepth()<maxDepth) {

Socket my\_socket = new Socket(current.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, current);

String line;

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

if (line.indexOf(current.URL\_PREFIX) != -1 && line.indexOf('"') != -1) {

StringBuilder currentLink = new StringBuilder();

int i = line.indexOf(current.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(), current.depth + 1);

if (current.check(findLink, newPair) && current.check(viewedLink, newPair) && !current.url.equals(newPair.url))

findLink.add(newPair);

}

}

my\_socket.close();

} catch(SocketTimeoutException e){

my\_socket.close();

}

}

viewedLink.add(current);

}

showResult(viewedLink);

}

public static void main(String[] args) {

// write your code here

String[] arg = new String[]{"http://government.ru/","1"};

try {

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

} catch (NumberFormatException | IOException e) {

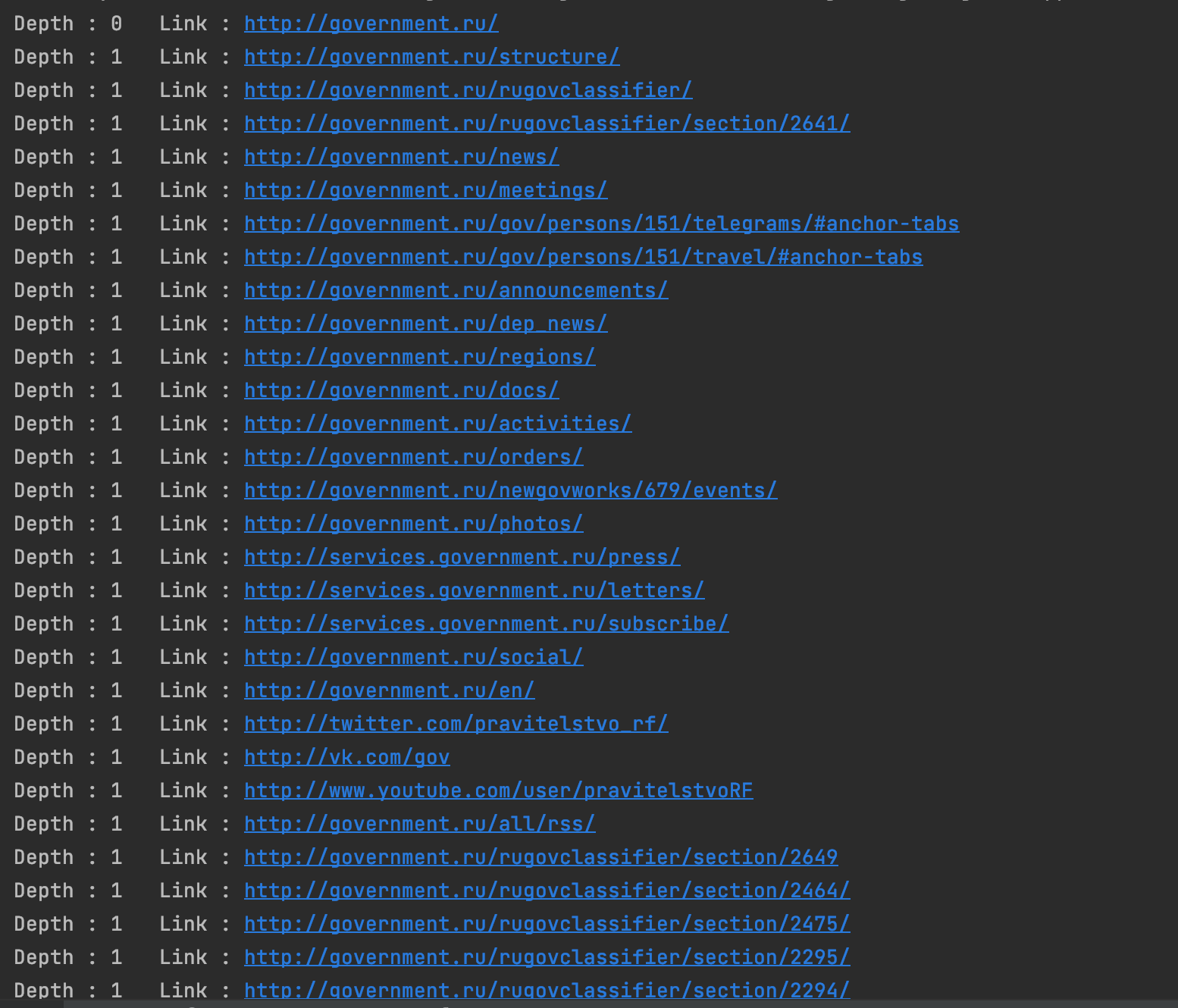
System.out.println("usage: java crawler " + arg[0] + " " + arg[1]);

}

}

}

**Скриншоты выполнения:**

****

**Заключение:**

Таким, образом мы реализовали элементарный веб-сканнер