**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе №7**

по дисциплине «Технологии разработки программного обеспечения»

Выполнил: студент группы БВТ1801

Клюшкин Дмитрий Алексеевич

Руководитель:

Мосева Марина Сергеевна

Москва 2019

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

**Выполнение:**

**Socket:**

import java.io.\*;  
  
public class Socket {  
 java.net.Socket socket;  
 Socket (String host, int port) throws IOException {  
 socket = new java.net.Socket(host,port);  
 socket.setSoTimeout(5000);  
 }  
 void setSoTimeout(int timeout)throws IOException {  
 socket.setSoTimeout(timeout);  
 }  
 InputStream getInputStream()throws IOException {  
 return socket.getInputStream();  
 }  
 OutputStream getOutputStream()throws IOException{  
 return socket.getOutputStream();  
 }  
 void close()throws IOException{  
 socket.close();  
 }  
}

**URLDepthPair:**

import java.net.\*;  
  
public class URLDepthPair {  
 String url;  
 int depth;  
  
 URLDepthPair(String url, int depth){  
 this.url=url;  
 this.depth=depth;  
 }  
  
 public boolean isURL(){  
 return url.matches("\\b(http)://[-a-zA-Z0-9+&@#/%?=~\_|!:,.;]\*[-a-zA-Z0-9+&@#/%=~\_|]");  
 }  
  
 public String getUrl() {  
 try {  
 URL url = new URL(this.url);  
 return url.getHost();  
 }  
 catch (MalformedURLException e) {  
 System.*err*.println("MalformedURLException: " + e.getMessage());  
 return null;  
 }  
 }  
  
 public int getDepth() {  
 return depth;  
 }  
 public String toString(){  
 return url+" ["+depth+"]";  
 }  
}

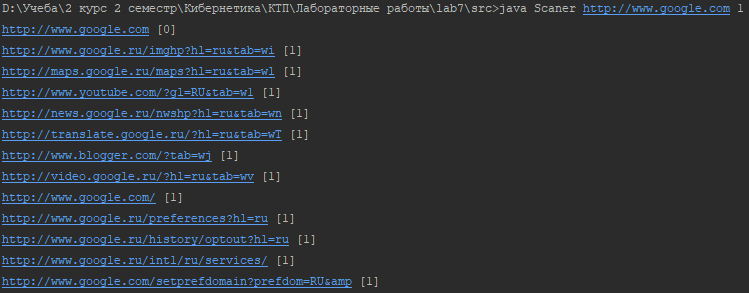
**Crawler:**

import java.io.\*;  
import java.util.LinkedList;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Crawler {  
 Socket socket;  
 BufferedReader bufferedReader;  
 PrintWriter printWriter;  
 LinkedList<URLDepthPair> siteList;  
 int depth;  
 Pattern regHTTP;  
  
 public LinkedList <URLDepthPair> work(URLDepthPair urlDepthPair,int port) throws IOException{  
 try {  
 //init  
 socket = new Socket(urlDepthPair.getUrl(), port);  
 bufferedReader = new BufferedReader(new InputStreamReader(socket.getInputStream()));  
 printWriter = new PrintWriter(socket.getOutputStream(), true);  
 depth = urlDepthPair.depth;  
 siteList = new LinkedList<URLDepthPair>();  
 regHTTP = Pattern.*compile*("(http:\\/\\/[\\w\\-\\.!~?&=+\\\*'(),\\/\\#\\:]+)((?!\\<\\/\\w\\>))\*?");  
 }  
 catch (Exception exc){  
 System.*out*.println(exc);  
 return new LinkedList<URLDepthPair>();  
 }  
 //end  
 //connection  
 printWriter.println("GET / HTTP/1.1");  
 printWriter.println("Host: "+urlDepthPair.getUrl()+":"+port);  
 printWriter.println("Connection: Close");  
 printWriter.println();  
 //end  
 try{  
 String line;  
 while ((line=bufferedReader.readLine())!=null) {  
 while(line.contains("<a")){  
 while (line.indexOf(">", line.indexOf("<a"))==-1) line+=bufferedReader.readLine();  
  
 String http = line.substring(line.indexOf("<a"),line.indexOf(">", line.indexOf("<a")));  
 if (http.contains("http://")){  
 Matcher matcher = regHTTP.matcher(http);  
 matcher.find();  
 String url = matcher.group();  
 siteList.add(new URLDepthPair(url,depth+1));  
 }  
 line=line.replace(http,"");  
 }  
 }  
 }  
 catch (IOException except){  
 System.*out*.println(except);  
 }  
 socket.close();  
 return siteList;  
 }  
}

**Scaner:**

import java.io.\*;  
import java.util. \*;  
public class Scaner {  
 static LinkedList<URLDepthPair> *site*;  
  
 public static void main(String[] args) throws IOException {  
 *site* = new LinkedList <URLDepthPair>();  
 Crawler crawler = new Crawler();  
  
 URLDepthPair startUrl = new URLDepthPair(args[0],0);  
  
 if (!startUrl.isURL() || args[1].matches("\\D+")) {  
 System.*out*.println("java Crawler <" + args[0] + "><" + args[1] + "> ");  
 return;  
 }  
  
 *site*.add(startUrl);  
 int i=0;  
 while(i<*site*.size()){  
 LinkedList<URLDepthPair> siteList = crawler.work(*site*.get(i),80);  
 if (siteList.size()>0)  
 *site*.addAll(i+1,siteList);  
 if (*site*.get(i).depth>=Integer.*parseInt*(args[1])-1){  
 if (siteList.size()==0)  
 i++;  
 else  
 i+=siteList.size()+1;  
 }  
 else  
 i++;  
 }  
 for(URLDepthPair url: *site*){  
 System.*out*.println(url);  
 }  
 }  
}

**Скриншот работы программы:**



Вывод: реализовали элементарный веб-сканер.