**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 «КТП»

Выполнил: студент группы БВТ1802

Ткачев Александр Владимирович

Руководитель:

Ксения Андреевна Полянцева

Москва 2020

**1 Цель работы**

Цель работы: изучить работу простейшего веб-сканера.

**2 Задание**

Написать программу, которая будет получать в аргументах командной строки URL-адрес и глубину поиска и посещать все ссылки, которые находятся на исходной web-странице в пределах указанной глубины поиска.

**3 Текст программы**

**Class ScannerApp**

package com.company;  
  
import java.io.IOException;  
  
public class ScannerApp {  
 public static void main(String args[]) throws IOException {  
 Crawler crawler = new Crawler(args[0], Integer.*parseInt*(args[1]));  
 crawler.Scan();  
 System.*out*.println("Depth: " + Integer.*parseInt*(args[1]));  
 crawler.getSites();  
 }  
}

**Class Crawler**

public class Crawler {  
  
 final static int *AnyDepth* = 0;  
 private LinkedList<URLDepthPair> m\_Processed = new LinkedList<URLDepthPair>();  
 private LinkedList<URLDepthPair> m\_NotProcessed = new LinkedList<URLDepthPair>();  
  
 private int m\_Depth;  
 private String m\_StartHost;  
   
 private String m\_Prefix = "http";  
 public Crawler(String host, int depth) {  
 m\_StartHost = host;  
 m\_Depth = depth;  
 m\_NotProcessed.add(new URLDepthPair(m\_StartHost, m\_Depth));  
 }  
  
 public void Scan() throws IOException {  
  
 while (m\_NotProcessed.size() > 0) {  
 Process(m\_NotProcessed.removeFirst());  
 }  
 }  
  
  
 public void Process(URLDepthPair pair) throws IOException{  
 URL url = new URL(pair.getURL());  
 URLConnection connection = url.openConnection();  
 String redirect = connection.getHeaderField("Location");  
 if (redirect != null) {  
 connection = new URL(redirect).openConnection();  
 }  
 m\_Processed.add(pair);  
 if (pair.getDepth() == 0) return;  
  
 BufferedReader reader = new BufferedReader(new InputStreamReader(connection.getInputStream()));  
 String input;  
 while ((input = reader.readLine()) != null) {  
 while (input.contains("a href=\"" + m\_Prefix)) {  
 input = input.substring(input.indexOf("a href=\"" + m\_Prefix) + 8);  
 String link = input.substring(0, input.indexOf('\"'));  
 if(link.contains(" "))  
 link = link.replace(" ", "%20");  
 if (m\_NotProcessed.contains(new URLDepthPair(link, *AnyDepth*)) ||  
 m\_Processed.contains(new URLDepthPair(link, *AnyDepth*))) continue;  
 m\_NotProcessed.add(new URLDepthPair(link, pair.getDepth() - 1));  
 }  
 }  
 reader.close();  
  
 }  
  
 public void getSites() {  
 for (var elem : m\_Processed)  
 System.*out*.println(elem.getURL());  
 System.*out*.println("Links visited: " + m\_Processed.size());  
 }  
  
  
}

**Class URLDepthPair**

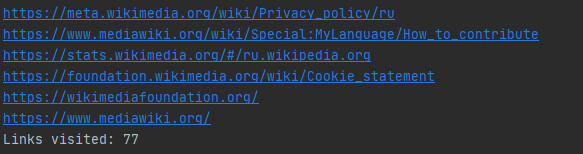
public class URLDepthPair {  
  
 private String m\_Url;  
 private int m\_Depth;  
  
 public URLDepthPair(String host, int depth) {  
 m\_Url = host;  
 m\_Depth = depth;  
 }  
  
 public String getURL() {  
 return m\_Url;  
 }  
  
 public int getDepth() {  
 return m\_Depth;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (obj instanceof URLDepthPair) {  
 URLDepthPair o = (URLDepthPair)obj;  
 return this.m\_Url.equals(o.getURL());  
 }  
 return false;  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*();  
 }  
}

**4 Работа программы**

Стартовый url - https://ru.wikipedia.org/ Глубина поиска - 1



....................................................



Стартовый url - https://ru.wikipedia.org/ Глубина поиска – 2

Часть ссылок пропущена ввиду большого количества.



.........................................................................

