5.2. Implementasi Program

Pada subbab ini akan dijelaskan mengenai pemrograman algoritma-algoritma yang telah dibahas pada bab sebelumnya. Pengimplementasi algoritma-algoritma crawler pada program dilakukan dengan menggunakan bahasa pemrograman C#. Alasan dipilihnya pemrograman menggunakan bahasa ini adalah karena belum banyak implementasi Tugas Akhir dengan menggunakan bahasa C#. Selain itu C# mendukung prinsip pemrograman berbasis objek yang sangat membantu dalam implementasi karena setiap class crawler dapat diturunkan dari crawler Breadth-First yang merupakan crawler sederhana.

Dengan kemampuan pemrograman berbasis object untuk membagi program ke dalam class-class yang berbeda, maka pada implementasi program ini program juga dipisahkan ke dalam beberapa class. Class ini dibagi menjadi beberapa yaitu class yang digunakan sebagai pendukung proses crawling dan class utama masing-masing algoritma crawling. Berikut ini penjelasan untuk masing-masing class yang digunakan.

5.2.1 Class Pendukung

Class pendukung adalah class yang digunakan untuk mendukung proses crawling seperti class untuk parsing HTML, class untuk melakukan fetch HTML, class untuk melakuka preprocessing, dan masih banyak lagi. Masing-masing class tersebut memiliki fungsi masing-masing yang spesifik yang mendukung class utama. Class pendukung dapat dibagi menjadi beberapa menurut fungsinya. Berikut ini penjelasan untuk masing-masing class pendukung menurut fungsinya masing-masing.

5.2.1.1 Class Untuk Parsing HTML

Class ini dibuat sendiri karena tidak disediakan secara built-in dalam bahasa pemrograman C#. Parsing disini ditujukan khususnya untuk mengambil semua link yang ada pada sebuah halaman web. Yang termasuk dalam halaman web tidak hanya file dengan extension html atau htm tetapi semua file yang dapat dibaca dengan mode teks. Syarat utama sebuah halaman untuk dapat diparasing adalah strukturnya harus benar. Class ini tidak mampu mem-parsing halaman yang tidak lengkap (isinya tidak lengkap).

Untuk melakukan parsing HTML, implementasinya menggunakan 4 buah class yang terpisah. Sebenarnya class-class ini dapat digabungkan tetapi untuk mempermudah pemrograman maka class ini dipisahkan menjadi class-class yang lebih kecil. Class-class tersebut antara lain :

1. Attribute

   Class ini merupakan turunan class Icloneable dan merupakan class yang dibuat sebagai parent dari class untuk parsing HTML yang lain. Class ini berfungsi untuk menyimpan data atribut pada tag html. Pada class ini terdapat 3 buah variabel untuk menyimpan atribut yaitu :

   - m_name, menyimpan nama atribut
   - m_value, menyimpan nilai atribut
   - m_delim, menyimpan delimiter atribut

2. AttributeList
   Class attributelist merupakan diturunkan dari class Attribute. Sesuai dengan namanya, class ini dibentuk untuk menyimpan daftar atribut beserta nilai dan delimiternya pada sebuah tag HTML. Pada class ini variabel baru ditambahkan yaitu :
   - m_list, bertipe arraylist, digunakan untuk menyimpan semua data atribut pada sebuah tag HTML.

   Pada class ini, method yang digunakan antara lain :
   - Add
     Method ini bertipe procedure dan berfungsi untuk menambahkan atribut baru. Parameternya bertipe Attribute
   - Clear
     Method ini bertipe procedure dan berfungsi untuk menghapus semua daftar variabel dari m_list.
   - IsEmpty
     Method ini bertipe function dengan nilai kembalian bertipe boolean dan berfungsi untuk mengetahui apakah atribut masih kosong (variabel m_list masih kosong).
   - Set (string name, string value)
     Method ini bertipe procedure dengan 2 buah parameter bertipe string yang masing-masing menyimpan nama atribut dan nilainya. Method ini berfungsi untuk membuat atau mengubah atribut. Jika Atribut sudah pernah ada pada m_list, maka akan berfungsi mengganti nilai atribut. Jika belum pernah ada atribut dengan nama sesuai pada parameter, maka akan menambahkan atribut baru pada m_list.
3. Parse
   Class parse diturunkan dari class attributelist dengan menambahkan beberapa variabel baru yaitu :
   - m_source, bertipe string, menyimpan halaman yang akan diparsing.
   - m_idx, bertipe integer, menyimpan posisi index string yang aktif.
   - m_parseDelim, bertipe string, menyimpan delimiter dari atribut yang diparsing.
   - m_parseName, bertipe string, menyimpan nama atribut yang diparsing.
   - m_parseValue, bertipe string, menyimpan nilai atribut yang diparsing.
   - m_tag, bertipe string, menyimpan nama tag yang sedang diparsing.

   Pada class ini, method yang digunakan antara lain :
   - IsWhiteSpace
     Method ini bertipe function dengan nilai kembalian bertipe boolean dan berfungsi untuk mengetahui apakah sebuah karakter termasuk whitespace (tab, spasi, dll). Parameter dari method ini ada 1 yang bertipe char. Method akan mengembalikan nilai true jika karakter yang dipasingkan termasuk whitespace, dan sebaliknya.

- EatWhiteSpace
  Method ini bertipe procedure tanpa parameter. Method ini akan melewatkan deretan whitespace yang ditemui pada m_source.
- Eof
  Method ini bertipe function dengan nilai kembalian bertipe boolean dan berfungsi untuk mengetahui apakah posisi m_idx sudah pada akhir string yang diparsing.
- ParseAttributeName
  Method ini bertipe procedure tanpa parameter. Method ini berfungsi untuk mengambil nama atribut. Nama atribut akan disimpan dalam m_parseName.
- ParseAttributeValue
  Method ini bertipe procedure tanpa parameter. Method ini berfungsi untuk mengambil nama atribut. Nama atribut akan disimpan dalam m_parseName.
- AddAttribute
  Method ini bertipe procedure tanpa parameter. Method ini akan menambahkan atribut yang baru saja diparsing ke dalam daftar atribut.
- getCurrentChar
  Method ini bertipe function dengan nilai kembalian bertipe char. Sesuai dengan namanya, method ini mengembalikan karakter yang ditunjuk saat itu.
- AdvanceCurrentChar
  Method ini bertipe function dengan nilai kembalian bertipe char. Sesuai dengan namanya, method ini mengembalikan karakter yang ditunjuk saat itu dan menggeser m_idx 1 karakter ke depan.
- Advance
  Method ini bertipe procedure tanpa parameter dan berfungsi menggeser m_idx 1 karakter ke depan.

4. ParseHTML
   Class parseHTML diturunkan dari class parse. Class inilah yang nantinya akan dipanggil pada saat akan melakukan parsing halaman web. Pada class ini beberapa method yang disediakan antara lain :
   - GetTag
     Method ini bertipe function dengan nilai kembalian bertipe AttributeList. Method ini akan mengembalikan tag beserta data atribut dan nilainya yang baru saja didapatkan.
   - BuildTag
     Method ini bertipe function dengan nilai kembalian bertipe string. Method ini membentuk kembali tag HTML beserta atributnya dan mengembalikannya dalam bentuk string.

- ParseTag
  Method ini bertipe procedure yang akan melakukan parsing terhadap tag HTML. Parsing di sini meliputi mengambil nama tag beserta atributnya.
- Parse
  Method ini bertipe function dengan nilai kembalian bertipe char. Method inilah yang dipanggil dari program pemanggil untuk memulai parsing HTML.

5.2.1.2 Class Untuk Fetch Halaman Web

Istilah fetch berarti mengambil halaman yang ada di Internet untuk disimpan ke dalam memori atau media penyimpanan lain. Untuk dapat melakukan proses fetch, komputer harus terkoneksi ke Internet baik menggunakan modem atau koneksi ke Internet yang lain. Class ini tidak hanya digunakan untuk mengambil halaman dari Internet, tetapi juga melakukan beberapa fungsi yang lain. Berikut ini akan dijelaskan source program pada class ini.

Segmen Program 5.1 Modul untuk menyimpan halaman web (Fetching)

```
1.      public string GetPage(string url){
2.        WebResponse response = null;
3.        Stream stream = null;
4.        StreamReader reader = null;
5.        try  {
6.          status = 0;
7.          HttpWebRequest request =
          (HttpWebRequest)WebRequest.Create(url);
8.          response = request.GetResponse();
9.          stream = response.GetResponseStream();
10.       if (!response.ContentType.ToLower().StartsWith("text/"))
11.         return null;
12.       string buffer = "";
13.       reader = new StreamReader(stream);
14.       string line;
15.       while( (line = reader.ReadLine())!=null ){
16.         buffer += line+"\r\n";
17.       }
18.       return buffer.ToLower();
19.     }
20.     catch(Exception) {
21.       return null;
22.     }
23.     finally {
24.     if ( reader!=null )
25.          reader.Close();
26.       if ( stream!=null )
27.         stream.Close();
28.       if ( response!=null )
29.         response.Close();
30.     }
31.  }
```

Proses fetch pada segmen program 5.1 menggunakan bantuan class WebResponse dan HttpWebRequest untuk melakukan koneksi dengan Internet. Langkah pertama sebelum melakukan fetch adalah melakukan koneksi dengan Internet. Proses melakukan koneksi Internet ada pada baris 7 sampai baris 9. Baris 7 menunjukkan koneksi dengan URL yang akan didownload dengan menggunakan bantuan class HttpWebRequest. Baris 8 untuk mendapatkan respon dari HttpWebRequest. Jika halaman yang akan didownload tidak ditemukan maka perintah pada baris 8 ini akan mengembalikan pesan error yang akan ditangkap pada baris 20 – 21. Baris 9 digunakan untuk mendapatkan stream yang digunakan untuk membaca isi halaman web. Baris 15 sampai baris 17 adalah perulangan yang membaca seluruh isi halaman web. Hasil pembacaan halaman ditampung dalam variabel Buffer yang bertipe string, dan kemudian method akan mengembalikan Buffer. Baris 23 sampai dengan 30 adalah bagian untuk menutup koneksi ke Internet setelah proses fetch selesai dilakukan.

Segmen Program 5.2 Modul untuk membatasi tipe outlink

```
1.   public Boolean followExtRule(string link){
2.     Boolean result = true;
3.     if ((link.IndexOf(".jpg")>=0)||(link.IndexOf(".jpeg")>=0)||
4.        (link.IndexOf(".gif")>=0) || (link.IndexOf(".exe")>=0)||
5.        (link.IndexOf(".mp3")>=0) || (link.IndexOf(".mpg")>=0)||
6.        (link.IndexOf(".pdf")>=0) || (link.IndexOf(".zip")>=0)||
7.        (link.IndexOf(".css")>=0) || (link.IndexOf(".rar")>=0)||
8.        (link.IndexOf(".gz")>=0) || (link.IndexOf(".tar")>=0) ||
9.        (link.IndexOf(".ps")>=0) || (link.IndexOf(".doc")>=0) ||
10.       (link.IndexOf("javascript:")>=0) ||
11.       (link.IndexOf("#")>=0)||(link.IndexOf("mailto:")>=0)||
12.       (link.IndexOf(".bmp")>=0) || (link.IndexOf(".ico")>=0))
          {
13.         result = false;
14.    }
15.    return result;
16.  }
```

Modul untuk membatasi tipe link ini bertujuan agar link yang bertipe nonteks tidak perlu dimasukkan ke dalam queue sehingga dapat menghemat space dalam queue yang dapat digunakan untuk link lain yang lebih berguna. Modul ini hanya berupa perintah percabangan sederhana dimana jika pada link yang merupakan parameter modul ditemukan substring yang menandakan tipe file tertentu, maka link tersebut akan ditolak dan tidak dimasukkan ke dalam queue. Tipe file yang tidak diijinkan untuk dimasukkan dalam queue adalah file gambar (jpg, jpeg, gif, ico, dan bmp), file dokumen (.doc) , file hasil kompresi (.zip, .rar, .gz, .tar), file pdf (.pdf, .ps), link menuju halaman sendiri, link yang menunjuk pada alamat email (mailto:).

Segmen Program 5.3 Modul untuk mendapatkan outlink halaman (Parsing)

```
1. public ArrayList parseLink(string page){
2.   ArrayList link = new ArrayList();
3.   try {
4.     if(page==null)
5.       return null;
6.     ParseHTML parse = new ParseHTML();
7.     parse.Source = page;
8.     String Text="";
9.     while( !parse.Eof() ){
10.      char ch = parse.Parse();
11.        if(ch==0){
12.          AttributeList tag = parse.GetTag();
13.          if( tag["href"]!=null ){
14.            string tlink = (string)tag["href"].Value;
15.            if (followExtRule(tag["href"].Value))
16.              link.Add(tag["href"].Value);
17.          }
18.        }
19.        else {
20.          Text += ch;
21.        }
22.      }
23.    }
24.    catch (Exception) {
25.      return null;
26.    }
27.    return link;
28.  }
```

Proses parsing ditujukan untuk mengambil semua link keluar (outlink) dari halaman web. Method ini mengembalikan nilai berupa tipe data array yang berisi semua link yang ditemukan pada halaman web. Method ini menerima parameter berupa string yang merupakan isi halaman web. Proses parsing dilakukan pada baris 9 sampai baris 22. Teks yang diterima sebagai inputan diparsing dengan class ParseHTML seperti yang telah dijelaskan pada bagian sebelumnya. Semua tag HTML diambil, tetapi hanya atribut href saja yang disimpan. Nilai setiap atribut href inilah yang diambil sebagai daftar outlink.

Method-method yang dijelaskan di atas adalah method utama yang paling penting dari class Fetcher. Sebenarnya selain method yang dijelaskan di atas, ada beberapa method lain yang juga digunakan pada proses crawling antara lain :

- parseHtmlWithText
  Method ini menerima parameter string isi halaman yang akan diparsing. Tujuan dari method ini adalah mengubah bentuk HTML sehingga setiap teks yang tidak masuk dalam tag HTML diapit diantara tag <text>…</text> dan atribut yang diambil hanya atribut href saja. Tujuan melakukan proses ini adalah untuk mempermudah pada saat akan mangambil text yang ada di sekitar link (diperlukan pada algoritma Shark-Search).

- parseText
  Method parseText digunakan untuk mengambil semua teks pada halaman web. Semua tag beserta atribut dan comment akan dibuang. Tujuan dari proses ini adalah untuk mengambil teks html saja untuk melakukan perhitungan cosine similarity sehingga hasil yang diperoleh lebih baik.
- parseLinkWithText
  Method ini khusus digunakan dalam algoritma Shark-Search yaitu untuk sekaligus mengambil setiap link, teks disekitar link, dan teks pada link. Nilai kembalian method ini berupa HashTable dengan alamat outlink sebagai key.

5.2.1.3 Class Untuk Preprocessing

Preprocessing yang dimaksud disini ada 2 buah proses yaitu proses membuang stopword dan common word dan proses stemming. Untuk proses membuang stop word dan common word, class yang digunakan adalah Stopper, sedangkan untuk proses stemming, class yang digunakan adalah Stemmer. Berikut ini penjelasan kedua class tersebut.

1. Stopper
   Class Stopper digunakan untuk membuang semua stopword dan commond word yang ditemukan pada teks. Dalam melakukan prosesnya, class ini mengandalkan daftar stopword dan common word yang disimpan dalam sebuah teks file. Beberapa method yang disediakan pada class ini antara lain :
   - readStopWord
     Method ini digunakan untuk membaca daftar stopword dan common word dari teks file dan dimasukkan ke dalam sebuah ArrayList. Parameter dari method ini berupa nama file yang akan dibaca.
   - StopString
     Method StopString digunakan untuk menghilangkan semua stopword dan common word yang ditemukan dalam sebuah string. Parameter string yang diterima dipisah-pisahkan per kata berdasarkan whitespace kemudian setiap kata yang termasuk dalam stopword atau common word dibuang. Nilai kembalian dari method ini adalah string yang telah diproses.
   - isStopWord
     Method isStopWord digunakan untuk mencocokkan apakah sebuah kata termasuk stopword atau bukan. Kata yang akan dicek dicocokkan dengan daftar kata pada ArrayList. Jika ditemukan pada daftar, maka nilai true akan dikembalikan dan false jika sebaliknya.
2. Stemmer
   Class Stemmer digunakan untuk melakukan proses stemming seperti yang telah dijelaskan pada bab 2. Algoritma yang diimplementasikan adalah Porter Stemmer. Pada class ini method yang dipanggil untuk melakukan proses stemming antara lain :

- stemString
  Method ini digunakan untuk melakukan proses stemming pada string panjang yang terdiri dari beberapa kata. Setiap kata pada string dipisahkan kemudian diproses terpisah. Hasilnya digabungkan lagi menjadi satu string panjang.
- stemTerm
  Method ini digunakan untuk melakukan proses stemming pada sebuah term atau kata.

5.2.1.4 Class Robot Exclusion Protokol

Untuk menghargai pemilik web server, maka perlu diimplementasikan robot exclusion protokol yang membatasi crawler dengan halaman-halaman tertentu yang tidak diperbolehkan untuk dibaca. Class ini menyimpan bagian dari mana saja dari sebuah server yang tidak boleh dikunjungi oleh crawler. Pada implementasinya, setiap kali crawler akan membaca halaman web, terlebih dahulu melakukan pengecekan apakah diperbolehkan untuk didownload. Beberapa method yang diimplementasikan pada class ini adalah sebagai berikut.

- Get
  Method ini digunakan untuk mengembalikan daftar link yang terlarang bagi crawler pada sebuah server. Parameter dari method ini adalah nama server, dan mengembalikan nilai berupa ArrayList.
- Add
  Method ini digunakan untuk menambahkan exclusion yang baru pada sebuah server yang sebelumnya belum ada pada daftar.
- isDisallowed
  Method ini memiliki nilai kembalian bertipe boolean. Parameter method ini adalah URL yang akan diperiksa (string) dan daftar URL yang tidak dapat dikunjungi pada sebuah server (ArrayList). Jika sebuah URL diperbolehkan untuk dikunjungi maka method ini mengembalikan nilai false, dan true jika sebaliknya.
- getList
  Method getList berfungsi untuk mendapatkan daftar URL yang tidak diperbolehkan untuk dikunjungi pada sebuah server. Method ini menerima parameter berupa string yang merupakan isi dari file robots.txt. Method ini akan mengembalikan daftar URL yang terlarang (ArrayList).

5.2.1.5 Class Penyimpan Daftar Halaman

Pada proses crawling, beberapa thread berjalan bersamaan sehingga membutuhkan sebuah struktur data yang dapat diakses secara bergantian oleh semua thread yang berjalan. Maka dari itu diperlukan class khusus yang menyimpan data-data pada saat proses crawling. Selain itu juga perlu dipastikan bahwa class ini harus aman dari proses Thread.

Class yang dibuat untuk menangani permasalahan ini adalah class List. Class ini terutama digunakan untuk menyimpan daftar URL yang telah dikunjungi

crawler, daftar URL yang tidak dapat dikunjungi, dan beberapa data penting yang lain. Pada class ini beberapa data yang penting adalah sebagai berikut :

- Buffer
  Data ini bertipe ArrayList. Setiap elemen pada arraylist bertipe class ListElement, yaitu class yang menyimpan alamat URL, score, dan parent. Data ini menyimpan alamat URL yang dapat dikunjungi oleh crawler. Setiap elemen pada data ini telah diurutkan secara descending berdasarkan score.
- Visited
  Data ini bertipe ArrayList. Setiap elemen bertipe string, yaitu alamat URL yang telah dikunjungi dan berhasil didownload.
- BadURL
  Data ini bertipe ArrayList. Setiap elemen bertipe string, yaitu alamat URL yang telah dikunjungi dan tetapi tidak berhasil didownload. Variabel ini berguna untuk membantu agar crawler tidak perlu mengunjungi halaman yang sama.
- Downloading
  Data ini bertipe ArrayList. Setiap elemen bertipe string, yaitu alamat URL yang sedang didownload pada setiap thread.
- urlInBuffer
  Data ini bertipe ArrayList. Setiap elemen bertipe string, yaitu alamat URL yang sedang didownload pada setiap thread.
- maxBuffer
  Data ini bertipe integer yang menyimpan jumlah maximum elemen yang diperbolehkan pada Buffer.
- maxVisited
  Data ini bertipe integer yang menyimpan jumlah maximum halaman yang harus dikunjungi crawler .
- current
  Data ini bertipe string yang menyimpan alamat URL yang terakhir diambil dari buffer.

Segmen Program 5.4 Modul untuk memasukkan secara terurut pada List

```
1.  public void insertOrder(string url, double score)  {
2.   int first =  0;
3.   int last =  Buffer.Count-1;
4.   int mid =  0;
5.   ListElement fe = new ListElement(url,score);
6.   if (Buffer.Count == 0) {
7.     insertFirst(fe);
8.   }
9.   else {
10.       if (score > getScoreAt(first) ) {
11.          Buffer.Insert(0, fe);
12.          urlInBuffer.Insert(0, fe.url);
13.       }
14.       else if (score <= getScoreAt(last)) {
15.          Buffer.Add(fe);
16.          urlInBuffer.Add(fe.url);
```

```
17.       }
18.       else {
19.          while ((last-first) > 1) {
20.             mid = (last + first)/2;
21.             if (getScoreAt(mid) >= score) {
22.                first = mid;
23.             }
24.             else {
25.                last = mid;
26.             }
27.          }
28.          Buffer.Insert(last, fe);
29.          urlInBuffer.Insert(last,fe.url);
30.       }
31.    }
32.    if (Buffer.Count >= maxBuffer) {
33.       Buffer.RemoveAt(maxBuffer-1);
34.    }
35. }
```

Method insertOrder pada segmen program 5.4 digunakan untuk memasukkan alamat URL beserta score-nya ke dalam list (Buffer) secara terurut descending agar link dengan score tertinggi diletakkan pada urutan yang pertama sehingga akan diambil pertama kali. Inti dari method ini adalah mencari posisi index untuk meletakkan alamat URL yang baru. Baris 10 sampai 13 digunakan untuk memastikan jika skor alamat yang baru lebih besar daripada skor URL yang pertama pada list, maka proses langsung meletakkan alamat URL pada index pertama dan keluar. Sedangkan Baris 14 sampai 17 untuk kasus skor yang baru paling kecil.baris 19 sampai 27 adalah perulangan untuk mencari posisi jika kondisi pertama dan kedua tidak terpenuhi. Jika jumlah elemen pada buffer telah lebih dari batas yang diperbolehkan maka elemen dengan skor terburuk (terletak di akhir dari list akan dibuang (baris 32 dan 33).

Segmen Program 5.5 Modul untuk mengambil elemen pertama

```
1. public ListElement getFirstElement()
2. {
3.   Monitor.Enter(this);
4.   ListElement le = (ListElement)Buffer[0];
5.   Buffer.RemoveAt(0);
6.   urlInBuffer.RemoveAt(0);
7.   Monitor.Exit(this);
8.   return le;
9. }
```

Potongan program di atas adalah untuk mengambil elemen pertama dari Buffer. Selain membaca elemen yang pertama, sekaligus menghapus elemen tersebut dari Buffer. Perintah pada baris 3 digunakan untuk mengunci object selama ada thread yang mengakses. Hal ini ditujukan agar pada suatu waktu hanya 1 thread saja yang diperbolehkan mengakses object yang sama.

5.2.1.6 Class Pendukung Lain

Class-class yang telah disebutkan di atas adalah class pembantu yang sifatnya spesifik. Terdapat satu class pendukung lain yang juga penting. Class Helper adalah class dengan sifat yang umum. Beberapa fungsi umum dimasukkan ke dalam class ini. Berikut ini dijelaskan beberapa method yang penting dari class Helper.

Segmen Program 5.6 Modul untuk menghitung Cosine Similarity

```
1. public static double getSim(String text1, String text2){
2.   text1 = text1.ToLower();
3.   text2 = text2.ToLower();
4.   Regex rgx = new Regex("\\s+");
5.   String[] text1Tokens = rgx.Split(text1);
6.   String[] text2Tokens = rgx.Split(text2);
7.   Hashtable ht1 = new Hashtable();
8.   Hashtable ht2 = new Hashtable();
9.   for (int i = 0; i < text1Tokens.Length; i++) {
10.    if (ht1.ContainsKey(text1Tokens[i])) {
11.      ht1[text1Tokens[i]] =
   (UInt16)((UInt16)ht1[text1Tokens[i]] + 1);
12.    }
13.    else
14.      ht1.Add(text1Tokens[i], (UInt16)1);
15.  }
16.  for (int i = 0; i < text2Tokens.Length; i++) {
17.    if (ht2.ContainsKey(text2Tokens[i])) {
18.      ht2[text2Tokens[i]] =
   (UInt16)((UInt16)ht2[text2Tokens[i]] + 1);
19.    }
20.    else
21.      ht2.Add(text2Tokens[i], (UInt16)1);
22.  }
23.  double mag1 = 0;
24.  IEnumerator en = ht1.Keys.GetEnumerator();
25.  while (en.MoveNext()){
26.    mag1 = mag1 +  Math.Pow(((UInt16)ht1[en.Current]),2);
27.  }
28.  mag1 = Math.Sqrt(mag1);
29.  double mag2 = 0;
30.  en = ht2.Keys.GetEnumerator();
31.  while (en.MoveNext()){
32.    mag2 = mag2 +  Math.Pow(((UInt16)ht2[en.Current]),2);
33.  }
34.  mag2 = Math.Sqrt(mag2);
35.  double mult = 0;
36.  for (IEnumerator e=ht1.Keys.GetEnumerator();e.MoveNext();) {
37.    String term = (String) e.Current;
38.    if (ht2.ContainsKey(term)) {
39.      mult=mult+((UInt16)ht2[term])*((UInt16)ht1[term]);
40.    }
41.  }
42.  double cosineSim = mult / (mag1 * mag2);
43.  return cosineSim;
44. }
```

Method getSim pada segmen program 5.6 digunakan untuk menghitung nilai kemiripan (similarity) antara 2 buah string. Perhitungan kemiripan ini menggunakan Cosine Similarity yang menghitung kemunculan term dan frekuensinya pada kedua buah string dan membandingkannya. Pertama-tama setiap kata pada kedua buah string dipisahkan, kemudian masing-masing dimasukkan ke dalam struktur data HashTable (baris 5 sampai baris 24). Perhitungan nilai cosine similarity dilakukan pada baris 25 sampai baris 44.

Alamat absolute adalah alamat URL lengkap yang terdiri dari nama domain dan diakhiri dengan nama path. Seringkali pada sebuah halaman web, link hanya menggunakan alamat relatif sehingga crawler tidak dapat langsung menggunakan alamat relatif tersebut. Untuk memberi gambaran mengenai alamat relatif dan absolute, misalnya sebuah halaman web dengan alamat URL : http://www.TugasAkhir.com/index.html memiliki sebuah tag dengan atribut href=”about.html”. Alamat URL yang ditunjuk oleh tag href tersebut adalah alamat relatif. Alamat absolute dari alamat tersebut sebenarnya adalah http://www.TugasAkhir.com/about.html. Agar crawler dapat menerima alamat URL relatif ini, maka alamat relatif tersebut harus diterjemahkan menjadi alamat absolute terlebih dahulu.

Segmen Program 5.7 Modul untuk mendapatkan alamat absolute url

```
1. public static string getCanonicalCombination(string parent,
   string url){
2.  string result =url;
3.  try {
4.   if (!url.StartsWith("http://")) {
5.     if (parent == null) {
6.       result = getCanonical(url);
7.     }
8.     else {
9.       Uri uri1 = new Uri(parent);
10.        Uri uri2 = new Uri(uri1,url);
11.        result = getCanonical(uri2.ToString());
12.      }
13.    }
14.    else {
15.      result = getCanonical(url);
16.    }
17.  }
18.  catch(Exception)
19.  {
20.    return null;
21.  }
22.  return result;
23. }
```

Untuk mendapatkan outlink dengan alamat absolute dari sebuah alamat relatif, diperlukan alamat absolute halaman yang bersangkutan (parent). Untuk melakukannya, menggunakan bantuan class Uri. Baris 4 sampai baris 13 dijalankan jika alamat URL dalam bentuk alamat relatif. Baris 9 sampai baris 11 digunakan untuk membentuk alamat absolute. Setelah membentuk alamat absolute

(kombinasi antara alamat parent dengan alamat relatif), selanjutnya alamat tersebut diperiksa lagi dengan method getCanonical yang melakukan pengecekan apakah alamat yang dibentuk menunjuk pada sebuah file atau tidak.

Segmen Program 5.8 Modul untuk pengecekan alamat web

```
1. public static string getCanonical(string url){
2.   url = url.ToLower();
3.   Boolean fileflag = false;
4.   if (url.StartsWith("http://")) {
5.     try {
6.       Uri uri1 = new Uri(url);
7.       string domain = uri1.Host;
8.       string path = uri1.PathAndQuery;
9.       Match m = Regex.Match(path,"(.*?)#(.*)");
10.        if (m.Success) {
11.          path = m.Groups[1].ToString();
12.          fileflag = true;
13.        }
14.        if (!fileflag &&
15.          !Regex.IsMatch(path,"(.+?)\\.(.+)") &&
16.          !path.EndsWith("/") &&
17.          path.Length > 0 &&
18.          !Regex.IsMatch(path,".*%3F.*")){
19.          path += "/";
20.        }
21.        string result = "http://" + domain + path;
22.        return result;
23.      }
24.      catch (Exception e){
25.        System.Console.WriteLine(e.Message);
26.        return null;
27.      }
28.    }
29.    else {
30.      return null;
31.    }
32.  }
```

Method pada segmen program 5.8 terutama berguna untuk melakukan pengecekan apakah sebuah alamat URL menunjuk pada sebuah file atau tidak. Pada baris 7 adalah untuk mendapatkan domain dari alamat URL. Kemudian baris 8 mendapatkan path beserta query (kalau ada). Baris 9 sampai dengan baris 13 adalah untuk melakukan pengecekan link ke halaman itu sendiri sekaligus menghilangkan bagian di belakang tanda “#”. Baris 14 sampai baris 19 adalah jika alamat URL tidak menunjuk pada file dan tidak diakhiri dengan “/” maka akan ditambahkan tanda “/” pada akhir string. Jika sudah menunjuk pada file, langsung dikembalikan. Setelah melakukan pengecekan, alamat URL yang telah dipisahkan menjadi domain dan path digabungkan kembali. Jika terjadi error dalam proses, maka method akan mengembalikan nilai null.