IMPLEMENTASI METODE RABIN KARP UNTUK MENDETEKSI TINGKAT KESAMAAN DUA DOKUMEN

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana
Teknik Pada Jurusan Teknik Informatika

oleh :

ZAINAL MUJAHIDIN
10751000186

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM
PEKANBARU
RIAU
2013

IMPLEMENTASI METODE *RABIN KARP* UNTUK MENDETEKSI TINGKAT KESAMAAN DUA DOKUMEN

ZAINAL MUJAHIDIN
10751000186

Tanggal Sidang: 02 JuLi 2013
Periode Wisuda: November 2013

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Plagiat atau biasa disebut penjiplakan adalah sebuah masalah yang cukup signifikan pada berbagai kalangan yang selalu berinteraksi dengan komputer. Hal plagiat yang biasanya dilakukan terhadap konten digital adalah melakukan *copy-paste*, *quote*, dan revisi terhadap dokumen asli. Untuk mengantisipasinya, dibutuhkan suatu cara yang dapat menganalisis teknik-teknik plagiat yang dilakukan. Ada beberapa pendekatan yang bisa diambil, salah satunya dengan mempergunakan algoritma *Rabin Karp*. Algoritma *Rabin Karp* adalah salah satu algoritma pencocokan string yang dapat digunakan untuk mengukur kemiripan teks. Tugas ahir ini bertujuan untuk merancang dan membangun sebuah aplikasi dengan menggunakan algoritma *Rabin Karp* untuk mencari persentase kesamaan pada dua dokumen teks yang diuji. Dari hasil pengujian tersebut persentase yang didapatkan mulai dari 0% sampai dengan 100%. Semakin kecil tingkat persentase kesamaan pada dokumen teks (*similarity*) yang dihasilkan dari pengujian, maka dokumen tersebut tidak termasuk plagiarisme, tetapi jika hasil *similarity* dari pengujian yang dilakukan pada dua dokumen semakin besar, maka dapat disimpulkan bahwa dokumen tersebut termasuk hasil dari tindakan plagiat.

Kata kunci : algoritma *Rabin Karp*, kesamaan dokumen teks, plagiarisme, *similarity*

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Alhamdulillaahi Robbil'alamin, penulis bersyukur ke-hadirat Allah SWT, karena atas segala limpahan rahmat dan karunia-Nya yang diberikan sehingga penulis dapat menyelesaikan penelitian dan penulisan laporan tugas akhir ini. *Allahumma sholli'ala Muhammad wa'ala ali sayyidina Muhammad*, yang tidak lupa penulis haturkan juga untuk Rosul Allah, Muhammad SAW. Laporan tugas akhir ini merupakan salah satu persyaratan akademis untuk meraih gelar sarjana di Jurusan Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim Riau (UIN SUSKA Riau). Selama menyelesaikan tugas akhir ini, penulis telah banyak mendapatkan bantuan, bimbingan, dan petunjuk dari banyak pihak baik secara langsung maupun tidak langsung. Untuk itu dalam kesempatan ini penulis ingin mengucapkan terimakasih yang sebesar-besarnya kepada:

1. Prof. Dr. H. M. Nazir, selaku Rektor Universitas Islam Negeri Sultan Syarif Kasim Riau.
2. Dra. Yenita Morena, M.Si, selaku Dekan Fakultas Sains dan Teknologi.
3. DR. Okfalisa, ST, M.Sc, selaku Ketua Jurusan Teknik Informatika.
4. Surya Agustian ST, M.Kom Selaku dosen pembimbing tugas akhir. Dengan sabar telah membarikan banyak waktu, ilmu, semangat, dan motivasinya.
5. Jasril, ST, MSc, selaku dosen penguji 1, yang banyak meluangkan waktu dan menberikan ilmunya yang sangat membantu dalam penyempurnaan Laporan Tugas Akhir ini.
6. Iwan Iskandar, ST, MT, selaku dosen penguji 2, dengan cukup sabar memberikan ilmunya dan masukan- masukan yang sangat membantu untuk menyelasaikan Tugas Ahir ini.
7. Nazrudin Syafaat H, ST,MT, terimakasih pak atas saran dan masukannya.

8. Resky, ST, MSc, sebagai koordinator tugas akhir, yang telah memberikan banyak waktu mempersiapkan semua kebutuhan penulis dalam penyelesaian Tugas Akhir ini.
9. Kedua orang tua, adik, kakak serta saudara penulis yang selalu memerikan do'a seratadukungan salama menyelesaikan tugas ahir ini9. Para dosen Teknik Informatika yang telah memberikan ilmu serta bimbingan dan nasehat-nasehat disaat kuliah maupun di luar kampus.
10. Teman-teman PIS yang selalu membirikan semangat dan memberikan babtuanny dalam menyelesaikan Tugas Ahir ini.
11. Teman-teman kelas TIF 2007, Agutin kurniasari, Nuriyadi, Joko Nuryanto, Jadno, Hendra, Ardian, very dan imam serta teman-teman lain yang tidak dapat penulis sebutkan satu-persatu, terima kasih atas saran dan bantuannya serta semangat yang diberikan selama ini.
12. kakak dan adik tinkat TIF yang telah memberikan saran dan masukan dalam menyelesaikan Tugas Ahir.
13. Sari Widayati, Andi Ardiansyah, Tri Handoko dan seroja yang telah memberikan waktu serta bantuanya dalam membuat Tugas Ahir ini.

Akhirnya, penulis menyadari dalam penulisan laporan ini masih terdapat kekurangan. Oleh karena itu, saran dan kritik sangat penulis harapkan untuk kemajuan penulis secara pribadi. Terimakasih.

Pekanbaru, July 2013

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi informasi dan komunikasi yang semakin pesat, mengakibatkan pencarian informasi semakin mudah. Begitu juga dengan penyimpanan data dan prosesnya yang mengalami perkembangan secara pesat sehingga memudahkan orang untuk menyimpan, mencari, dan mengolah data dengan cepat. Dengan adanya kemudahan yang ditawarkan oleh media seperti *internet* untuk mendapatkan informasi merupakan salah satu dampak positif dari kemajuan teknologi. Namun perkembangan tersebut tidak terlepas dari dampak negatif yang hampir tidak dapat dihindari lagi, salah satunya adalah plagiarisme. Plagiarisme adalah tindakan penyalahgunaan, pencurian atau perampasan, penerbitan, pernyataan, atau menyatakan sebagai milik sendiri sebuah pikiran, ide, tulisan, atau ciptaan yang sebenarnya milik orang lain.

Praktik plagiat mudah terjadi diberbagai kalangan yang selalu berinteraksi dengan komputer mengingat adanya fasilitas untuk menyalin dan mengubah teks (*copy* dan *paste*) dan fasilitas koneksi yang memungkinkan untuk mengakses hasil karya orang lain secara bebas melalui internet. Plagiarisme dapat mematikan kreatifitas seseorang karena tindakan ini tidak membutuhkan tenaga dan tidak harus berfikir keras. Oleh karena itu, tindakan plagiarisme secara perlahan harus dicegah. Dengan memanfaatkan metode pencocokan string pada dokumen, dapat dikembangkan untuk merancang aplikasi pendeteksi plagiarisme. Algoritma pencocokan string itu sendiri ada bermacam-macam, antara lain Boyer-Moore, Brute Force, Knuth-Morris-Pratt, Rabin-Karb, Smith-Waterman dan lain-lain.

Pada penelitian sebelumnya (Nugroho, 2011) dilakukan penelitian secara skematis bagaimana cara kerja algoritma *Rabin Karp* mendeteksi *plagiarisme* pada suatu dokumen, dalam penelitian tersebut Nugroho menyatakan bahwa algoritma ini cocok untuk pencarian banyak *multi pattern*.

Pada skripsi ini Penulis mencoba untuk mengimplementasikan metode *Rabin Karp* untuk mendeteksi tingkat kesamaan pada dua dokumen.

Berdasarkan latar belakang diatas, maka penulis tertarik untuk mengangkat kedalam penelitian yang berjudul "**IMPLEMENTASI METODE RABIN KARP UNTUK MENDETEKSI TINGKAT KESAMAAN DUA DOKUMEN**".

1.2. Rumusan Masalah

Berdasarkan latar belakang diatas dapat diuraikan rumusan masalahnya, yaitu : Bagaimana mengimplementasikan metode *Rabin Karp* agar dapat mendeteksi tingkat kesamaan yang terdapat pada dua dokumen?

1.3. Batasan Masalah

Dalam pembuatan skripsi ini, untuk mengatasi permasalahan diatas akan diberi beberapa batasan masalah, yaitu :

1. Menguji data berupa teks, tidak menguji data berupa gambar maupun suara.
2. Tidak memperhatikan kesalahan ejaan/penulisan pada dokumen.
3. Tidak memperhatikan sinonim/persamaan kata.
4. Data yang diuji menggunakan bahasa Indonesia.

1.4. Tujuan Penelitian

Tujuan yang ingin dicapai penulis dari skripsi ini antara lain :

1. Membuat aplikasi pendeteksi plagiat menggunakan metode *Rabin Karp*.
2. Mendeteksi tingkat kesamaan dua dokumen teks dengan metode *Rabin Karp*.

1.5. Sistematika Penulisan

Bab I Pendahuluan

Berisikan mengenai latar belakang permasalahan, rumusan masalah, batasan masalah, tujuan dari pembahasan, metodologi penelitian dan sistematika penulisan.

Bab II Landasan Teori

Pada bab ini membahas teori-teori pendukung yang berkaitan dengan skripsi yang akan dibuat. Teori yang akan diangkat yaitu mengenai deteksi plagiarism dengan metode *Rabin Karp*.

Bab III Metodologi Penelitian

Membahas tentang tahapan penelitian, tahapan pengumpulan data, analisa kebutuhan sistem, perancangan perangkat lunak, implementasi, pengujian sistem dan analisa akhir.

Bab IV Analisa dan Penelitian

Pada bab ini membahas tentang analisis sistem pendeteksi tingkat plagiatisme dengan menerapkan metode *Rabin Karp*.

Bab V Implimentasi dan Pengujian

Bab ini berisi penjelasan mengenai implementasi sistem pendeteksi tingkat plagiatisme dengan menerapkan metode serta kesimpulan dari pengujian yang telah diakukan terhadap sistem.

Bab VI Penutup

Bab ini berisi kesimpulan dari skripsi yang dibuat dan menjelaskan saran-saran penulis kepada pembaca agar penerapan metode *Rabin Karb* dapat dikembangkan lebih lanjut.

BAB II

LANDASAN TEORI

2.1. Pengertian Plagiarisme

Plagiarisme berasal dari kata Latin, “plagiarius”, yang berarti “pencuri”.Plagiarisme didefinisikan sebagai tindakan atau praktik mengambil danmengumpulkan atau menyampaikan pemikiran, tulisan atau hasil karya oranglain selayaknya hasil karya diri sendiri tanpa persetujuan dari pemilik hasil karyatersebut. Dengan kata lain, mempraktikkan plagiarisme berarti mencuri hasilkarya atau kepemilikan intelektual orang lain. Plagiarisme adalah tindakan penyalahgunaan, pencurian/perampasan, penerbitan, pernyataan, atau menyatakan sebagai milik sendiri sebuah pikiran, ide, tulisan, atau ciptaan yang sebenarnya milik orang lain. (Eko Nugroho, 2011). Sedangkan menurut Kamus Besar Bahasa Indonesia (KBBI), Plagiat merupakan pengambilan karangan (pendapat, dan sebagainya) orang lain dan menjadikannya seolah-olah karangan (pendapat dan sebagainya) sendiri, misalnya menerbitkan karya tulis orang lain atas nama dirinya sendiri.

Plagiat dapat dianggap sebagai tindak pidana karena mencuri hak cipta orang lain. Pelaku plagiat disebut sebagai plagiator.Sistem pendeteksi plagiarisme dapat dikembangkan untuk :

1. Data teks seperti *essay*, artikel, jurnal, penelitian dan sebagainya.
2. Dokumen teks yang lebih terstruktur seperti bahasa pemrograman.

Berikut ini adalah hal-hal yang tergolong kedalam tindakan plagiarisme, antara lain :

1. Mengakui tulisan orang lain sebagai tulisan sendiri.
2. Mengakui gagasan orang lain sebagai pemikiran sendiri.
3. Mengakui temuan orang lain sebagai kepunyaan sendiri.
4. Mengakui karya kelompok sebagai kepunyaan atau hasil sendiri.

5. Menyajikan tulisan yang sama dalam kesempatan yang berbeda tanpa menyebutkan asal-usulnya.
6. Meringkas dan memparafrasekan (mengutip tak langsung) tanpa menyebutkan sumbernya.
7. Meringkas dan memparafrasekan dengan menyebut sumbernya, tetapi rangkaian kalimat dan pilihan katanya masih terlalu sama dengan sumbernya.

2.1.1. Tipe-tipe Plagiarisme

Plagiarisme tidak selalu disengaja atau mencuri sesuatu dari orang lain. Praktik ini dapat bersifat tidak disengaja, kebetulan, dan dapat mencakup pencurian sendiri (*self stealing*).Berikut ini beberapa tipe plagiarisme.

a. Kebetulan (*accidental*)

   Praktik plagiarisme ini dapat terjadi karena kurangnya pengetahuan akan plagiarisme dan pemahaman mengenai penulisan referensi.

b. Tidak disengaja (*unintentional*)

   Ketersediaan informasi dalam jumlah yang sangat besar mempengaruhi pemikiran sehingga ide yang sama dapat dihasilkan secara tertulis maupun lisan sebagai milik pribadi.

c. Disengaja (*intentional*)

   Tindakan menyalin sebagian atau keseluruhan hasil karya orang lain secara sengaja tanpa mengikutsertakan nama pemilik hasil karya.

d. Diri sendiri (*self plagiarism*)

   Penggunaan hasil karya yang dibuat diri sendiri dalam bentuk lain tanpa menunjuk hasil karya asli.

2.1.2. Metode Pendeteksi Plagiarisme

Metode Pendeteksi Plagiarisme dibagi menjadi tiga bagian yaitu metode perbandingan teks lengkap, metode dokumen *fingerprinting*, dan metode kesamaan kata kunci. Metode pendeteksi plagiarisme dapat digambarkan sebagai berikut :

a. Perbandingan Teks Lengkap

Metode ini diterapkan dengan membandingkan semua isi dokumen.Dapat diterapkan untuk dokumen yang besar.Pendekatan ini membutuhkan waktu yang lama tetapi cukup efektif, karena kumpulan dokumen yang diperbandingkan adalah dokumen yang disimpan pada penyimpanan lokal.Metode perbandingan teks lengkap tidak dapat diterapkan untuk kumpulan dokumen yang tidak terdapat pada dokumen lokal. Algoritma yang digunakan pada metode ini adalah algoritma *Brute-Force*, algoritma *edit distance*, algoritma *Boyer Moore* dan algoritma *lavenshtein distanc*e.

b. Dokumen *Fingerprinting*

Dokumen *fingerprinting*merupakan metode yang digunakan untuk mendeteksi keakuratan salinan antar dokumen, baik semua teks yang terdapat di dalam dokumen atau hanya sebagian teks saja.Prinsip kerja dari metode dokumen *fingerprinting*ini adalah dengan menggunakan teknik *hashing*.Teknik *hashing* adalah sebuah fungsi yang mengkonversi setiap string menjadi bilangan. Misalnya Rabin-Karp, Winnowing dan Manber.

c. Kesamaan Kata Kunci

Prinsip dari metode ini adalah mengekstrak kata kunci dari dokumen dan kemudian dibandingkan dengan kata kunci pada dokumen yang lain. Pendekatan yang digunakan pada metode ini adalah teknik dot.

2.2. Pengertian *Information Retrieval* (IR)

Information Retrieval merupakan bagian dari *computer science* yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. Berdasarkan referensi dijelaskan bahwa *Information Retrieval* merupakan suatu pencarian informasi yang didasarkan pada suatu *query* yang diharapkan dapat memenuhi keinginan *user* dari kumpulan dokumen yang ada. Beberapa pengertian *Information Retrieval* dari berbagai sumber, antara lain :

ISO 2382/1 mendefinisikan *Information Retrieval* (IR) sebagai tindakan, metode dan prosedur untuk menemukan kembali data yang tersimpan, kemudian menyediakan informasi mengenai subyek yang dibutuhkan.

Information Retrieval adalah studi tentang sistem pengindeksan, pencarian, dan mengingat data, khususnya teks atau bentuk tidak terstruktur lainnya.(Manning, 2009).

Information Retrieval adalah "bidang di persimpangan ilmu informasi dan ilmu komputer. Berkutat dengan pengindeksan dan pengambilan informasi dari sumber informasi heterogen dan sebagian besar-tekstual. Istilah ini diciptakan oleh Mooers pada tahun 1951, yang menganjurkan bahwa diterapkan ke aspek intelektual deskripsi informasi dan sistem untuk pencarian". (Hersh, 2003).

Information Retrieval (IR) dapat didefinisikan sebagai penerapan teknologi komputer untuk perolehan, pengorganisasian, penyimpanan, pencarian dan pendistribusian informasi.IR adalah sebuah aktivitas, dan seperti aktivitas lainnya IR juga memiliki tujuan. Contoh sederhana dari IR adalah ketika orang mencari informasi menggunakan *search engine* dengan memberikan *query* untuk mendapatkan informasi yang diinginkan. *Query* yang diberikan belum tentu memberikan hasil pencarian yang baik. Walaupun begitu, *query* hanya sebagai kata kunci untuk sebuah *search engine* dalam melakukan pencarian untuk mendapatkan hasil yang diinginkan oleh orang tersebut.(Jackson dan Moulinier, 2002).

Beberpa contoh aplikasi dari *Information Retrival* adalah *search engine* atau mesin pencarian yang terdapat pada jaringan internet. Pengguna dapat mencari halamanhalaman web yang dibutuhkannya melalui *search engine.* Contoh lain penerapan dari sistem temu kembali informasi adalah *text mining, dokumen clastering.*

Proses yang terjadi di dalam *Information Retrieval System* terdiri dari 2 bagian utama, yaitu *Indexing subsystem*, dan *Searching subsystem* (*matching system*).

2.2.1 Proses *Indexing*

Indexing subsystem adalah proses *subsystem* yang merepresentasikan koleksi dokumen kedalam bentuk tertentu untuk memudahkan dan mempercepat proses pencarian dan penemuan kembali dokumen yang relevan.

Pembangunan index dari koleksi dokumen merupakan tugas pokok pada tahapan *preprocessing* di dalam IR. Kualitas index mempengaruhi efektifitas dan efisiensi sistem IR. Index dokumen adalah himpunan term yang menunjukkan isi atau topik yang dikandung oleh dokumen. Index akan membedakan suatu dokumen dari dokumen lain yang berada di dalam koleksi. Ukuran index yang kecil dapat memberikan hasil buruk dan mungkin beberapa item yang relevan terabaikan.Index yang besar memungkinkan ditemukan banyak dokumen yang relevan tetapi sekaligus dapat menaikkan jumlah dokumen yang tidak relevan dan menurunkan kecepatan pencarian (*searching*).

Tahap-tahap yang terjadi pada proses *indexing* adalah sebagai berikut :

1. *Word Token*

   Yaitu mengubah dokumen menjadi kumpulan *term* dengan cara menghapus semua karakter dalam tanda baca yang terdapat pada dokumen dan mengubah kumpulan *term* menjadi *lowercase*.
2. *Stopword Removal*

   Proses penghapusan kata-kata yang sering ditampilkan dalam dokumen seperti: *and, or, not* dan sebagainya.
3. *Stemming*

   Proses mengubah suatu kata bentukan menjadi kata dasar.
4. *Term Weighting*

   Proses pembobotan setiap *term* di dalam dokumen.

2.2.2 Proses *Searching*

Ada beberapa proses yang terjadi saat melakukan *search*, antara lain adalah sebagai berikut :

1. *Parse query* yaitu memecah query menjadi bentuk token.
2. Proses *Stopword filtration*.

Token-token *query* yang telah dihasilkan pada proses parse *query* kemudian di filter melalui proses pembuangan *token* yang termasuk *Stopword.*

3. Proses *Stemming*

   Stopword tokens dari proses *stopword* sebelumnya kemudian di filter kembali melalui proses *Stemming* sehingga menghasilkan *stemmed term query.*

4. Transformasi *Query*

   Stemmed term query yang dihasilkan kemudian ditransformasikan apabila memerlukan. Artinya, apabila *query* yang diinputkan membutuhkan terjemahan ke dalam bentuk *query* bahasa lain maka sebelum mencari dokumen pada koleksi dokumen, *query* tersebut diterjemahkan duhulu melalui proses penerjemahan *query.* Sistem akan membandingkan *query* tersebut dengan koleksi dokumen sehingga mengembalikan dokumen-dokumen yang relevan dalam suatu bahasa yang berbeda dengan bahasa *query.*

5. Pemodelan dalam model ruang vektor

   Tiap *term* atau kata yang ditemukan pada dokumen dan *query* diberi bobot dan disimpan sebagai salah satu elemen vektor dan dihitung nilai kemiripan antara *query* dan dokumen.

6. Perangkingan dokumen atau konten berdasarkan nilai kemiripan antara *query* dan dokumen.

2.2.3 Model *Information Retrieval* (IR)

Model yang terdapat dalam *Information Retrieval* terbagi dalam 3 model besar, yaitu:

1. *Set-theoretic models*, model merepresentasikan dokumen sebagai himpunan kata atau frase. Contoh model ini ialah *standard Boolean model* dan *extended Boolean model*.
2. *Algebratic model*, model merepresentasikan dokumen dan *query* sebagai vektor atau matriks *similarity* antara vektor dokumen dan vektor *query*

yang direpresentasikan sebagai sebuah nilai skalar. Contoh model ini ialah *vector space model* dan *Latent Semantic Indexing* (LSI).

3. *Probabilistic model*, model memperlakukan proses pengembalian dokumen sebagai sebuah *probabilistic inference*. Contoh model ini ialah penerapan teorema bayes dalam model probabilistik.

2.3 *Text Mining*

Text Mining adalah suatu proses yang bertujuan untuk menemukan informasi atau tren terbaru yang sebelumnya tidak terungkap, dengan memproses dan menganalisa data dalam jumlah besar. (Ahmad Hatta A, 2011).

Text Mining memiliki definisi menambang data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen.

2.3.1 Tahapan *Text Mining*

Teks *mining* merupakan suatu proses yang melibatkan beberapa area teknologi. Text *mining* berusaha untuk membawa keluar dari teks dalam bentuk yang cocok untuk dikonsumsi oleh komputer secara langsung, dengan tidak perlu perantara manusia. Namun secara umum proses-proses pada teks *mining*mengadopsi proses *data mining*. Bahkan beberapa teknik dalam proses teks *mining* juga menggunakan teknik-teknik *data mining*.

Ada empat tahap proses pokok dalam teks *mining*, yaitu pemrosesan awal terhadap teks (*text preprocessing*), transformasi teks *texttransformation*), pemilihan fitur (*feature selection*), dan penemuan pola (*pattern discovery*).

2.3.1.1 *Text Preprocessing*

Tahap ini melakukan analisis semantik (kebenaran arti) dan sintaktik (kebenaran susunan) terhadap teks. Tujuan dari pemrosesan awal adalah untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan lebih lanjut.

Operasi yang dapat dilakukan pada tahap ini meliputi proses untuk menghilangkan bagian-bagianyang tidak diperlukan atau pembersihan teks yang dilakukan untuk mengubah data-data berkulitas yaitu data yang telah memenuhi persyaratan untuk dieksekusi padasebuah algoritma. Bentuk pembersihan teks ini seperti menghilangkan spasi, tandabaca, mengubah huruf kapital menjadi huruf kecil dan menghilangkan karakter-karakteryang tidak relevan lainnya.

1. *Case Folding*

   Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil.Hanya huruf “a” sampai dengan huruf “z” yang diterima.Karakter selain huruf dihilangkan dan dianggap delimiter.

2. *Tokenizing*

   Tokenizing adalah proses penghilangan tanda baca pada kalimat yang ada dalam dokumen sehingga menghasilkan kata-kata yang berdiri sendiri-sendiri.

3. *Filtering*

   Filtering adalah tahap mengambil kata-kata penting dari hasil *tokenizing*. Bisa menggunakan algoritma *stop list* (membuang kata yang kurang penting) atau *word list* (menyimpan kata penting).

4. *Stemming*

   Stemming adalah proses mengubah kata menjadi kata dasarnya dengan menghilangkan imbuhan-imbuhan pada kata dalam dokumen atau mengubah kata kerja menjadi kata benda. *Stem* (akar kata) adalah bagian dari kata yang tersisa setelah dihilangkan imbuhannya (awalan dan akhiran). Contoh :*connect* adalah *stem* dari *connected*, *connecting*, *connection*, dan *connections*.

5. *Tagging*

   Tagging adalah tahap mencari bentuk awal atau *root* dari tiap kata lampau atau kata hasil *stemming*. Contoh dari tahap ini adalah sebagai berikut :*was* → *be, used* → *use, stori* → *story.*

6. *TahapAnalyzing*

   Tahap *Analyzing* adalah merupakan tahap penentuan seberapa jauh keterhubungan antar kata-kata antar dokumen yang ada.

2.3.1.2 *Text Transformation*

Transformasi teks atau pembentukan atribut mengacu pada proses untuk mendapatkan representasi dokumen yang diharapkan. Pendekatan representasi dokumen yang lazim digunakan adalah model *bag of words* dan model ruang vector (*vector space model*). Transformasi teks sekaligus juga melakukan pengubahan kata-kata ke bentuk dasarnya dan pengurangan dimensi kata di dalam dokumen.Tindakan ini diwujudkan dengan menerapkan *stemming* dan menghapus *stopwords*.

2.3.1.3 *Feature Selection*

Pemilihan fitur (kata) merupakan tahap lanjut dari pengurangan dimensi pada proses transformasi teks. Walaupun tahap sebelumnya sudah melakukan penghapusan kata-kata yang tidak deskriptif (*stopwords*), namun tidak semua kata-kata di dalam dokumen memiliki arti penting.Oleh karena itu, untuk mengurangi dimensi, pemilihan hanya dilakukan terhadap katakata yang relevan yang benar-benar merepresentasikan isi dari suatu dokumen.Ide dasar dari pemilihan fitur adalah menghapus kata-kata yang kemunculannya di suatu dokumen terlalu sedikit atau terlalu banyak.Algoritma yang digunakan pada teks *mining*, biasanya tidak hanya melakukan perhitungan pada dokumen saja, tetapi juga pada *feature*. Empat macam *feature* yang sering digunakan:

1. *Character*, merupakan komponan individual, bisa huruf, angka, karakter spesial dan spasi, merupakan *block* pembangun pada level paling tinggi pembentuk semantik *feature*, seperti kata, *term* dan *concept*. Pada umumnya, representasi *character-based* ini jarang digunakan pada beberapa teknik pemrosesan teks.

2. *Words*.

   Terms merupakan *single word* dan *multiword phrase*yang terpilih secara langsung dari *corpus*.Representasi *term-based* dari dokumen tersusun dari *subset term* dalam dokumen.

3. *Concept*, merupakan *feature* yang di-*generate* dari sebuah dokumen secara manual, *rule-based*, atau metodologi lain. (Triawati, 2009)

2.3.1.4 *Pattern Discovery*

Pattern discovery merupakan tahap penting untuk menemukan pola atau pengetahuan (*knowledge*) dari keseluruhan teks.Tindakan yang lazim dilakukan pada tahap ini adalah operasi teks *mining*, dan biasanya menggunakan teknik-teknik *data mining*.Dalam penemuan pola ini, proses teks *mining*dikombinasikan dengan proses-proses *data mining*. Masukan awal dari proses teks *mining* adalah suatu data teks dan menghasilkan keluaran berupa pola sebagai hasil interpretasi atau evaluasi. Apabila hasil keluaran dari penemuan pola belum sesuai untuk aplikasi, dilanjutkan evaluasi dengan melakukan iterasi ke satu atau beberapa tahap sebelumnya. Sebaliknya, hasil interpretasi merupakan tahap akhir dari proses teks *mining* dan akan disajikan ke pengguna dalam bentuk visual. (Even-Zohar,2002).

2.4 ALGORITHMA RABIN KARP

Algoritma Rabin Karp adalah algoritma pencarian kata yang mencari sebuah pola berupa substring dalam sebuah teks menggunakan hashing. Algoritma ini sangat efektif untuk pencocokan kata dengan pola banyak.Salah satu aplikasi praktis dari algoritma Rabin Karp adalah dalam pendeteksian plagiarisme. Dalamilmu komputer, algoritma Rabin-Karp adalah algoritma pencarian string yangdibuat oleh Michael O. Rabin dan Richard M. Karp pada tahun 1987 yang menggunakan *hashing* untuk menemukan salah satu dari satu set string pola dalam teks.

Algoritma *Rabin Karp* adalah algoritma pencocokan string yang akan menggunakan fungsi *hash* sebagai pembanding antara string yang dicari (m) dengan *substring* pada teks (n). Apabila *hash value* keduanya sama maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila hasil keduanya tidak sama, maka *substring* akan bergeser ke kanan. Pergeseran dilakukan sebanyak (n-m) kali. Perhitungan nilai *hash* yang efisien pada saat pergeseran akan mempengaruhi performa dari algoritma ini. (David Indra Lesmana, 2012).

Langkah-langkah dalam algortima Rabin Karp :

1. Menghilangkan tanda baca dan mengubah ke teks sumber dan kata yang ingin dicari menjadi kata-kata tanpa huruf.
2. Membagi teks kedalam gram-gram yang ditentukan nilai *k-gram*nya
3. Mencari nilai *hash* dengan fungsi *hash* dari tiap kata yang terbentuk
4. Mencari nilai *hash* yang sama antara dua teks

2.4.1 *Preprosesing*

Menghilangkan karakter yang tidak digunakan pada dokumen teks, seperti menghilangkan tanda baca, meghilangkan spasi dan mengubah huruf basar menjadi huruf kecil.

2.4.2 *Parsing k-gram*

Parsing k-gram adalahmembentuk pola kata pada teks dengan memecah kata menjadi potongan-potongan dimana setiap potongan mengandung karakter sebanyak *k.K-gram*merupakansebuah metode yang diaplikasikan untuk pembangkitan kata ataukarakter.Metode *k-grams* ini digunakan untuk mengambil potongan-potongankarakter huruf sejumlah k dari sebuah kata yang secarakontinuitas dibaca dari teks sumber hingga akhir dari dokumen.

2.4.3 *Hashing*

Setelah melakukan *Parsing K-gram* maka langkah selanjutnya adalah *hashing* terhadap seluruh pecahan string hasil dari proses *parsing k-gram*. *Hashing* itu sendiri adalah suatu cara untuk mentransformasi *string* menjadi suatu nilai yang unik (*hash value*)dengan panjang tertentu (*fixed-length*) yang berfungsi sebagai penanda *string* tersebut. Pada sistem ini proses *hashing* memanfaatkan tabel ascii dengan rumus *Hash* (Diana dkk,2011):

$$H_{(C1 \ldots CK)} = C_1.b^{(K-1)} + c_2.b^{(K-2)} + \ldots + C_{(k-1)}.b^{(k)} + ck \quad \ldots\ldots\ldots \quad (2.1)$$

Keterangan :

H= Nilai *hash*

c = Nilai *ascii*karakter (desimal)

k = Banyak karakter (indeks karakter)

b = Basis Bilangan (nilai dari basis bilangan harus bilangan prima).

Alasan kenapa Basis bilangan (*b*) harus dipilih bilangan prima yang cukup besar, adalah untuk mengurangi kemungkinan adanya dua *hash value* yang sama.

2.4.4 *Similarity*

Similarity adalah persentasetingkat kemiripan antar dokumen. Setelah dilakukan pengujian terhadap dokumen teks dengan menerapkan metode tertentu, maka akan diperoleh nilai persentase kemiripannya atau nilai *similarity*nya. Untuk mendapatkan tingkat presentase kesamaan sebuah dokumen dengan dokumen lain dapat menggunakan Persamaan Jaccard Coefficient (Diana, dkk, 2011), yang ditunjukan pada Persamaan (2.2) .

$$Similarity\ (d_1,d_2) = \frac{\sum H\ d1 \cap \sum H(d2)}{\sum H\ d1 \cup \sum H(d2)} \times 100 \quad \% \qquad (2.2)$$

Keterangan :

d_1 = dokumen asli

d_2 = dokumen uji

H = nilai *hash*

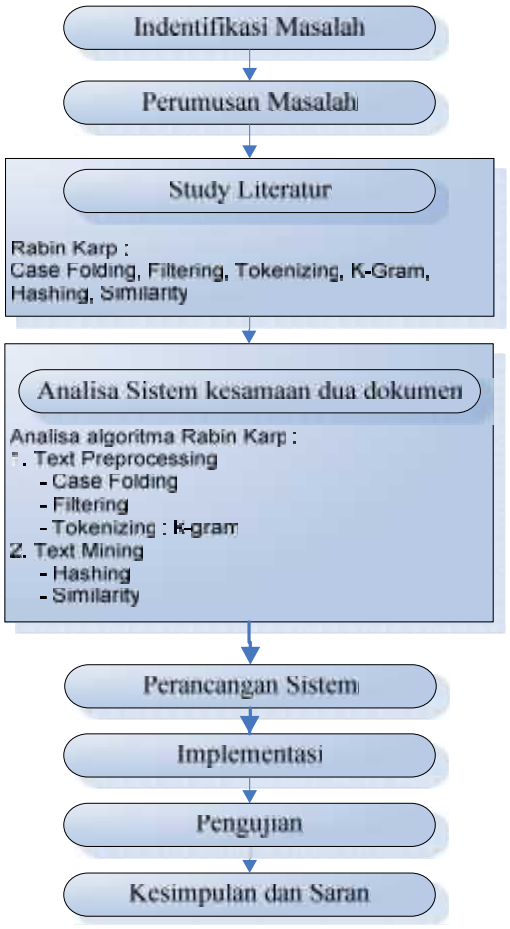
Algoritma *Rabin-Karp* mempercepat pengujian kemiripan untuk pola dari *substring* dalam teks dengan menggunakan fungsi *hash*. Fungsi *hash* adalah fungsi yang mengubah setiap kata dalam suatu nilai numerik, yang disebut nilai *hash*. Misalnya, *hash*(“kata”) = 5. *Rabin-Karp* menggunakan fakta bahwa jika dua kata merupakan kata yang sama, maka nilai *hash* dari kedua kata tersebut akan sama juga. Karenanya, untuk memeriksa kecocokan kata, hanya diperlukan perhitungan nilai *hash* dari *substring* yang akan dicari dengan kata-kata yang mempunyai nilai *hash* yang sama. Akan tetapi, ada permasalahan yang ditimbulkan dengan cara ini. Karena jenis kata yang berbeda sangatlah banyak, untuk menjaga nilai *hash* agar tetap kecil, beberapa kata yang berbeda perlu diberi nilai *hash* yang sama, yang berarti jika nilai *hash* sama, kata tersebut belum tentu sama. Untuk itu, diperlukan fungsi *hash* yang dapat dihandalkan agar untuk data masukan yang ada, hal ini tidak akan sering terjadi, sehingga algoritma ini mempunyai waktu pencarian rata-rata yang baik.

BAB III

METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Metodologi penelitian menjelaskan bagaimana langkah-langkah atau tahapan-tahapan yang akan dilakukan dalam penelitian untuk dapat menjawab perumusan masalah penelitian sesuai konsep sistem temu balik informasi.



Gambar 3.1. Tahapan Penelitian

3.2 Identifikasi Masalah

Pada tahapan ini dilakukan identifikasi permasalahan bahwa pentingnya bagi *user* untuk mengetahui tingkat plagiat atau kemiripan atau kesamaan yang terjadi pada dokumen teks sesuai dengan *query inputan* pengguna dari sekumpulan informasi (dokumen).

3.3 Perumusan Masalah

Dari tahapan identifikasi masalah, didapatkan rumusan masalah tentang bagaimana proses membangun aplikasi pendeteksi tingkat kesamaan dua dokumen dengan menggunakan metode *Rabin Karp.*

3.4 Studi Literatur

Studi literatur dilakukan dengan mempelajari konsep-konsep yang berkaitan dengan penelitian ini, seperti sistem pendeteksi tingkat kesamaan dua dokumen, tahapan *prepocessing* seperti *case folding, filtering, tokenizing, k-gram, hashing,* serta menghitung nilai *similarity* melalui literatur-literatur seperti buku, *paper*, dan sumber ilmiah lain seperti situs internet ataupun artikel dokumen teks yang berhubungan.

3.5 Analisa Sistem Kesamaan Dua Dokumen

Pada tahapan ini akan dilakukan analisa pembangunan sistem pendeteksi tingkat kesamaan dokumen terhadap kebutuhan pengguna dan kebutuhan perangkat lunak, dijelaskan secara rinci tentang proses dari sistem pendeteksi tingkat kesamaan dokumen yang akan dibangun sehingga mempermudah pemahaman terhadap sistem.

3.5.1 Analisa Algoritma *Rabin Karp*

Pada tahap ini dilakukan pengidentifikasian terhadap syarat-syarat algoritma dan langkah-langkah algoritma *Rabin karp.* Adapun langkah-langkah algoritma *Rabin Karp* adalah:

1. *Text Preprosesing*.

   Tahapan *text preprosesing* meliputi:

   a. Case folding

   b. Filtering

   c. Tokenizing: k-gram

2. *Text Mining*

   Tahapan *text mining* meliputi:

   a. *Hashing*

   b. *Similarity*

   .

3.6 Perancangan Sistem

Tahap perancangan sistem untuk mendeteksi tingkat kesamaan dua dokumen dengan metode *Rabin Karp* merupakan tahapan dalam membuat rincian sistem temu balik informasi berdasarkan analisa agar dapat dimengerti oleh pengguna (*user*). Rancangan utama yang akan dilakukan, yaitu:

1. Transformasi koleksi dokumen ke dalam *database* yang akan digunakan dalam sistem meliputi, *tokenization*, pembuangan *stopwords*, *filtering* dan juga *K-Gram*.
2. Perancangan antar muka sistem (*interface*) yang baik sehingga mudah digunakan (*user friendly*).

3.7 Implementasi Sistem

Pada implementasi sistem akan dilakukan pembuatan modul-modul yang telah dirancang dalam tahap perancangan kedalam bahasa pemrograman. Implementasi sistem akan dilakukan dengan spesifikasi sebagai berikut:

Perangkat Keras

Processor : *Intel pentium processor T4200 2.00 GHz*

RAM : 1 GB

Perangkat Lunak

Operating System : *Windows 7*

Bahasa Pemrograman : *PHP*

Tools Perancang : *XAMPP 1.5.3*

3.8 Pengujian Sistem

Pengujian merupakan tahapan dimana sistem akan dijalankan. Tahap pengujian diperlukan untuk menjadi ukuran bahwa sistem dapat dijalankan sesuai dengan tujuan. Pengujian sistem dilakukan dengan dua cara yaitu, pengujian secara *Whitebox* dan pengujian *Blackbox*. Pengujian *Whitebox* yaitu pengujian sampai ke-*coding* dimana ditemukan *bug* atau kesalahan pada algoritma yang tidak tampak pada pengujian *Blackbox*. Sedangkan pengujian *Blackbox* yaitu pengujian dimana kita sebagai pengguna biasa, dapat melihat secara umum aplikasi yang digunakan, tanpa mengetahui *coding* dan algoritma didalamnya.

3.9 Kesimpulan dan Saran

Tahapan ini akan membahas tentang kesimpulan yang dihasilkan dari penelitian skripsi, kesimpulan diambil dari proses analisa kepada implementasi dan pengujian.

Pada tahapan saran, penelitian ini diharapkan dapat memberikan sebuah catatan rekomendasi untuk menyempurnakan dan mengembangkan penelitian sistem pendeteksi tingkat kesamaan dokumen.

BAB IV

ANALISA DAN PERANCANGAN

4.1 Analisa Pendeteksi Kesamaan Dokumen

Pada bab ini akan dibahas secara garis besar tentang perancangan sistem deteksikesamaan dokumen dengan menggunakan algoritma *RabinKarb*. Pada umumnya algoritma pendeteksi kesamaan dokumen atau plagiarisme memiliki tahapan yang sama dalam pemrosesan dokumen teks. Tahapan-tahapan tersebut terdiri dari *input*, proses, *output*. Berawal dari tahap *input* dokumen teks yangakan diuji kemudian dokumen diproses melalui tahapan *prepocessing* (*Case Folding*, *Tokenizing*, *Filtering*) dan *hashing* kemudian hitungan *similarity* dokumen. Setelah diproses maka selanjutnya aplikasi akan menghasilkan informasi dokumen.

4.2 Analisa Metode *Rabin Karp*

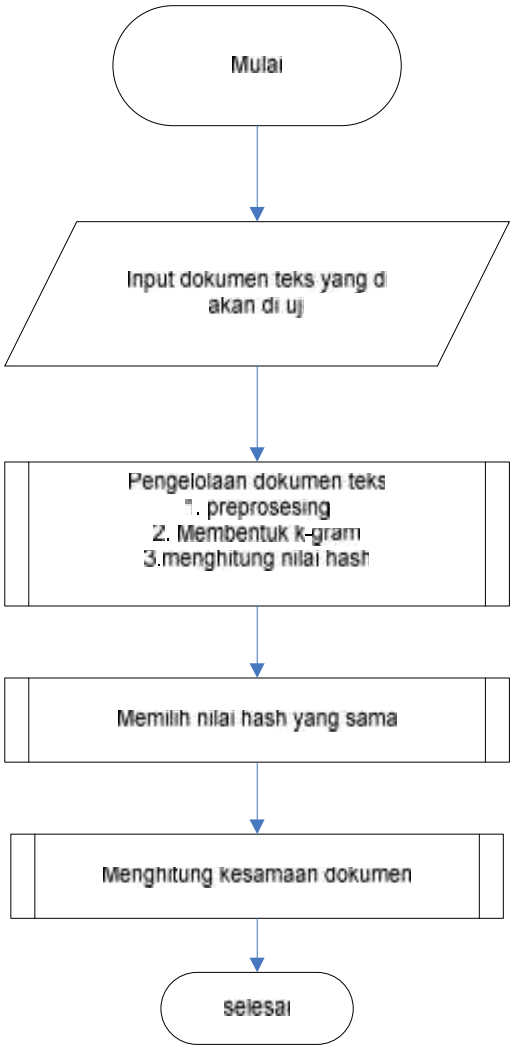
Pada tahap ini dilakukan analisa terhadap algoritma *rabin karp*. Dengan menentukan nilai *k-gram* dan basis bilangan primanya.Hasil dari *hashing* asli dan *hashing* uji kemudian dibandingkan untuk dicari *hashing* yang sama. Jika *hashing* yang sama ditemukan, maka dari hasil *hashing* yang sama tersebut dihitung tingkat persentase kesamaannya (*similarity*).

Adapun langkah-langkah algoritma Rabin karp yang dilakukan dalam perancangan sistem ini adalah sebagai berikut :

1. Menghilangkan tanda baca.
2. Membagi teks kedalam bentuk *k-gram*, dimana nilai *k* merupakan nilai parameter yang dipilih oelh pengguna.
3. Menghitung nilai *hash* dari setiap *k-gram.*
4. Memilih nilai *hash* yang sama.

5. Menghitung nilai tingkat kesamaan dua dokumen.

Untuk lebih jelasnya dapat dilihat pada *flowchart* berikut:

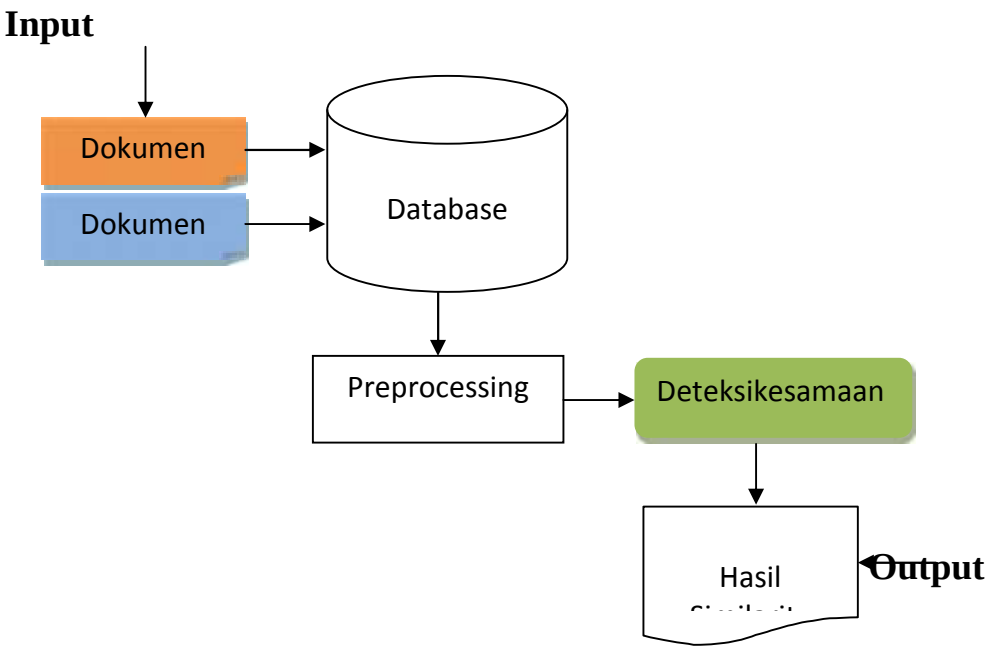


Gambar 4.1 *Flowchart* Metode *Rabin karp*

4.3 Perancangan Sistem Global

Tujuan dari perancangan sistem secara umum adalah untuk memberikangambaran secara umum kepada user tentang sistem yang baru.Analisis sistem dan desain sistem secara umum bergantung satu sama lain.Studi menunjukkan bahwa apa yang dikumpulkan, dianalisis dan dimodelkanselama fase analisis menyediakan dasar bagi desain sistem secara umumuntuk dibuat. Fase analisis sistem merupakan investigasi dan berorientasi ketemuan.

Secara garis besar sistem pendeteksi kesamaan pada dua dokumen dengan menggunakan metode *Rabin Karp*akandirancang dan kemudian akan dibuat sistemnya seperti yang tergambar pada skema alur kerja sistem berikut ini:



Gambar 4.2 Skema Alur KerjaSistem

Pada sistem yang akan dibuat ini, kedua dokumen yaitu dokumen asli dan dokumen uji yang diinputkan ke sistem akan diuji. Dokumentersebut harus berupa dokumen teks dengan ekstensi.txt.Kemudian dilakukan *preprocessing* pada dokumen tersebut dengan menggunakan metode-metode penunjang sebagai tahapan yang harus dilakukan, seperti yang sudah diuraikan pada bab sebelumnya. Setelah dilakukan tahapan *preprocessing*, kemudian dilakukan pengujian dengan metode *Rabin Karp*. Hasil akhir keluaran dari sistem ini adalah persentase kemiripan (*similarity*) dari metode *Rabin Karp*. Dari situ dapat diketahui berapa persen tingkat kesamaan antara kedua dokumen, yaitu dokumen asli dan dokumen uji.

4.4 Tahapan Proses Sistem Deteksi Kesamaan Dengan Metode Rabin Karp

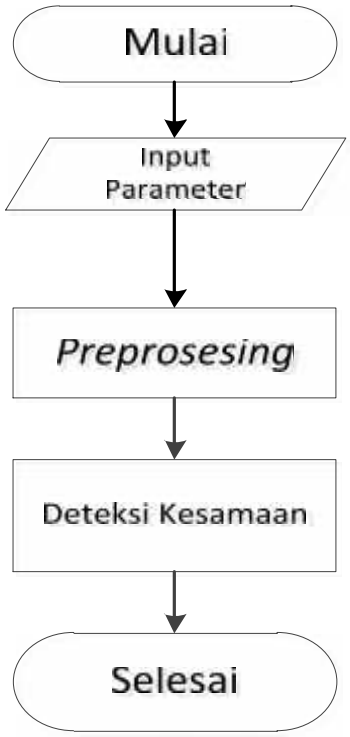
Inputan pada aplikasi ini berupa dokumen teks yang mempunyai ekstensi .txt. User akan menginputkan dua dokumen, yaitu dokumen asli dan dokumen yang ingin diuji. Setelah itu, sistem akan memproses kedua dokumen tersebut dan mengevaluasi berapakah *similarity* antara kedua dokumentersebut.

Pertama kali proses yang dilakukan oleh sistem adalah membaca file teks yang diinputkan oleh user. Dari dokumen yang telah diinputkan oleh user tadi, sistem akan melakukan pengecekan terhadap dokumen tersebut sehingga akan didapatkan informasi berupa ukuran file, type file dan jumlah file.

Pada perancangan sistem yang akan dibangun ini, penulis menjelaskan tahapan-tahapan perancangan sistem, baik dari tahapan *preprocessing* sampai hasil *similarity*.

Pertama kali proses yang dilakukan oleh sistem adalah membaca *file* teks yang diinputkan oleh *user*. Setelah sistem mendapatkan informasi dari dokumen yang telah diinputkan, sistem akan masuk ke tahap *preprocessing*. Pada tahap ini akan dilakukan beberapa proses, yaitu *Case folding* (mengubah semua huruf menjadi huruf kecil), *filtering* (penghilangan kata-kata yang kurang penting), kemudian proses *tokenizing*

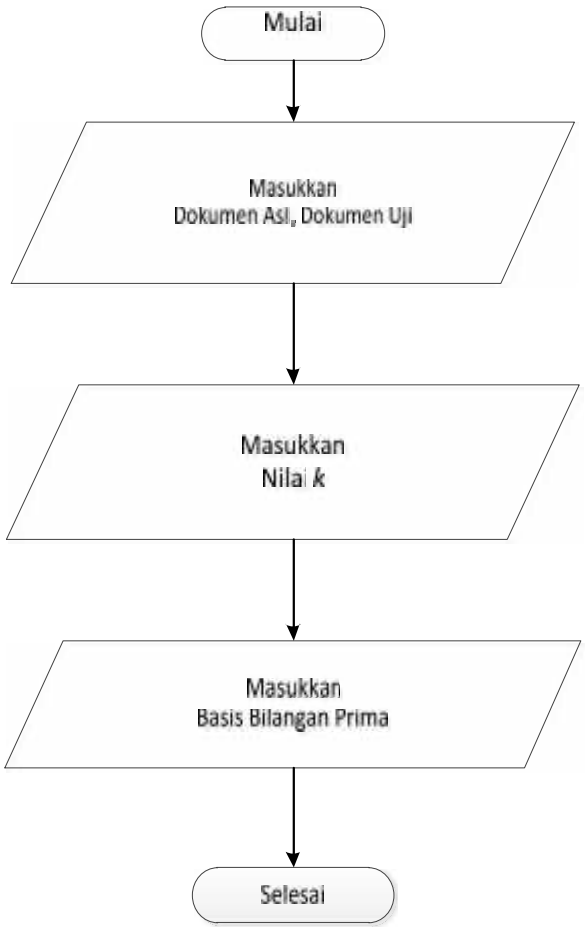
(pembentekan rangkaian *k-gram*).Kemudian dilakukan proses *hashing* dan untuk selanjutnya akan diperoleh hasil *similarity*nya. Seperti gambar 4.3 berikut



Gambar 4.3 Tahapan Sistem Deteksi Kesamaan Dua Dokumen Dengan Menggunakan Metode *Rabin Karp*

4.4.1 Tahap Input Parameter

Berikut adalah rancangan arsitektur sistem deteksi tingkat kesamaan dua dokumen dengan menggunakan metode *Rabin Karp*, seperti yang terlihat pada gambar 4.4 berikut.



Gambar 4.4 Proses Input Parameter

Pada *flowchart* program tersebut, digambarkan bahwa proses sistem yang akan dibuat dimulai dari memasukkan kedua dokumen yaitu dokumen asli dan dokumen uji. Kemudian akan dilakukan pengujian melalui tahapan-tahapan *preprocessing*, yaitu menghilangkan tanda baca, merubah huruf kehuruf kecil, menghilangkan tanda baca, menghilangkan kata kata yang kurang penting, kemudian pembentukan rangkaian *gran* panjang *k*, lalu menghitung nilai *hashing*. Dari pengujian tersebut akan didapatkan hasil akhirnya berupa persentase kemiripan (*similarity*).

Berikut ini adalah contoh perhitungan kesamaan dokumen yang akan di proses dengan menggunakan metode *Rabin Karp.*

Dokumen asli yang di input

> Salah satu dampak negatif dari perkembangan ilmu, teknologi dan informasi adalah maraknya tindakan “Plagiarisme”.

Dokumen uji yang di input

> "Plagiarisme" merupakan dampak negatif dari perkembangan teknologi informasi.

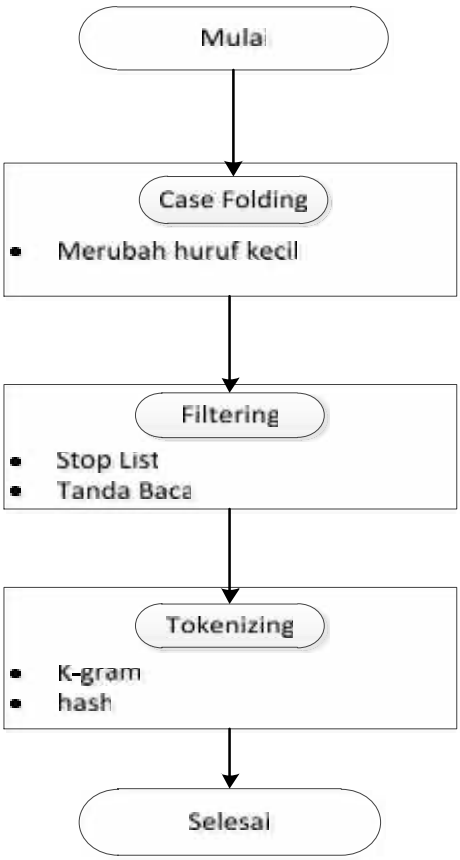
Dengan *k-gram* k= 5

Basis bilanga b= 7

4.4.2 Tahapan *Preprocessing*

Tahap *Preprocessing* harus dilalui untuk menentukan *keyword* pada kedua dokumen yang akan dilakukan pengujian, yaitu dokumen asli dan dokumen uji. Pada tahap *preprocessing* terdapat beberapa proses yang dilakukan oleh sistem terhadap dokumen yang diinputkan. Proses-proses tersebut antara lain *case folding*, *filtering*, *tokenizing*.

Berikut ini adalah *flowchart* dari *preprocessing* seperti yang terlihat pada gambar 4.5 dibawah ini.



Gambar 4.5 Proses *Preprocessing* Dokumen

4.4.2.1 Sub Proses *Case Folding*

Proses *case folding* (mengubah huruf kapital menjadi huruf kecil) merupakan tahap pertama yang akan dilakukan dari rangkaian tahapan yang terdapat pada *preprocessing*. Berikut adalah contoh proses *case folding* :

Dokumen asli setelah melalui proses *case folding* (proses 1.1)

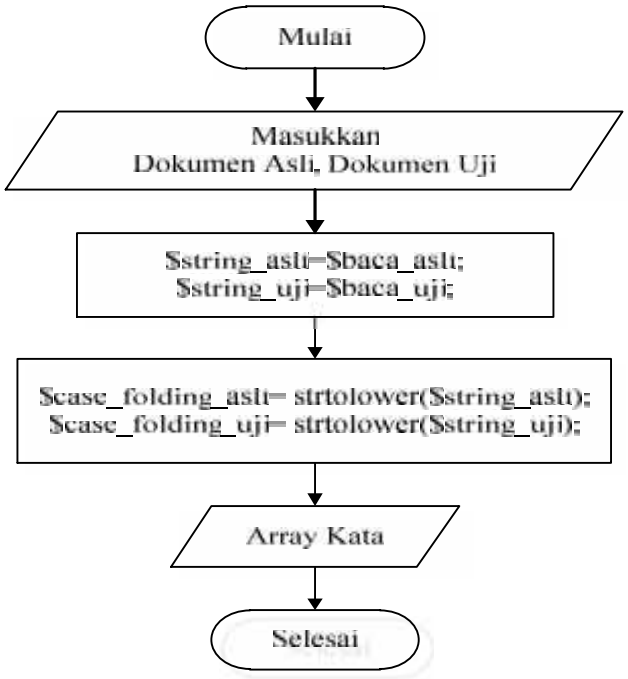
salah satu dampak negatif dari perkembangan ilmu, teknologi dan informasi adalah maraknya tindakan “plagiarisme”.

dokumen uji setelah melalui proses *case folding* (proses 1.1)

"plagiarisme" merupakan dampak negatif dari perkembangan teknologi informasi.

Pada contoh proses *case folding* diatas, kalimat yang terdapat huruf kapitalnya adalah pada kata “Salah” dan pada kata “Plagiarisme”. Dimana pada kedua kata tersebut setelah dilakukan proses *case folding*akan diubah menjadi huruf kecil semua. Pada kata “Salah” akan berubah menjadi “salah” dan pada kata “Plagiarisme” akan berubah menjadi “plagiarisme”. Sehingga secara keseluruhan kata-kata pada dokumen yang terdapat huruf kapitalnya akan berubah menjadi huruf kecil semua.

Berikut ini adalah proses *case folding* yang digambarkan dalam bentuk *flowchart*, seperti yang terlihat pada gambar 4.6 dibawah ini.



Gambar 4.6 Sup Proses *Preprosessing* untuk *Case Folding*

4.4.2.1 Sub Proses *Filtering*

Proses *filtering* dilakukan untuk mengambil kata-kata penting dari hasil *tokenizing*. Bisa menggunakan algoritma *stop list* (membuang kata yang kurang penting) atau *word list* (menyimpan kata penting). Setelah proses *case folding*, dokumen selanjutny masuk ketahap *filtering*, yaitu proses membuang kata yang tidak penting dan membang tanda baca. Berikut ini contoh proses *filtering:*

Dokumen asli setelah melalui proses *filtering* (proses 1.2)

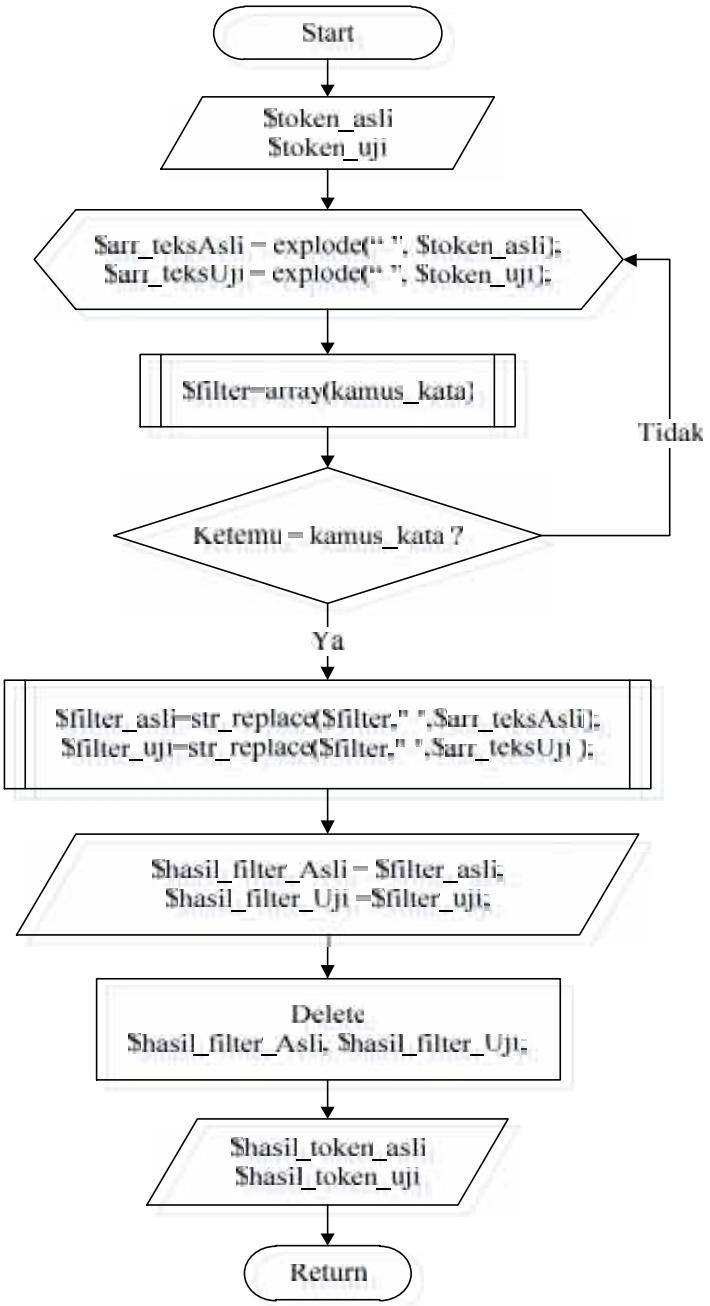
salah satu dampak negatif perkembangan ilmu teknologi informasi maraknya tindakan plagiarisme

Dokumen uji setelah melalui proses *filtering* (proses1.2)

Plagiarisme merupakan dampak negatif perkembangan teknologi informasi

Pada contoh *filtering* diatas yaitu proses menghialngkan tanda baca dan menghilangkan *stop word*. Pada contoh dokumen asli tanda baca yang di hilangkan adalah *titik* (.), *koma* (,) dan *tanda petik* (“ ”), sedangkan kata yang di anggap tidak penting dan dihilangkan adalah kata “dari”, kata “dan” dan kata “adalah. Sedangkan pada dokumen uji tanda baca yang di hilanghan adalan *titik* (.) dan *tanda petik* (“ ”), sedangkan dokumen uji kata yang dihilangkan adalah kata “dari”.

Proses *filtering* tersebut seperti yang digambarkan dalam sebuah *flowchart* pada gambar 4.7 berikut.



Gambar 4.7 Sub Proses *Filtering* Yang Menghilangkan *Stopword*

4.4.2.2 Sub Proses *Tokenizing*

Setelah dilakukan proses *filtering*, selanjutnya adalah proses *tokenizing*, proses ini merupakan proses pembentukan pola kata, dimana pola katanya dalam bentuk *gram* dengan panjang *k*. Pada proses *tokenizing* di bagi menja dua sub proses yaitu proses *parsing k-gram* dan proseses *hashing.*

Berikutini adalah proses *tokenizing* yang digambarkan dalam bentuk *flowchart*, seperti yang terlihat pada gambar 4.8 dibawah ini.

Gambar 4.8 Sub Proses *Tokenizing*

4.4.2.2.1 Proses *Parsingk-gram*

Langkah selanjutnya adalah *parsing k-gram*, dimana pada proses ini kata dipecah menjadi potongan-potongan dimana setiap potongan mengandung karakter sebanyak *k*. Berikut ini adalah contoh proses *parsing k-gram k*=5:

Hasil *k-gram* dokumen asli yang telah melalui proses tokenisasi (proses 2.1)

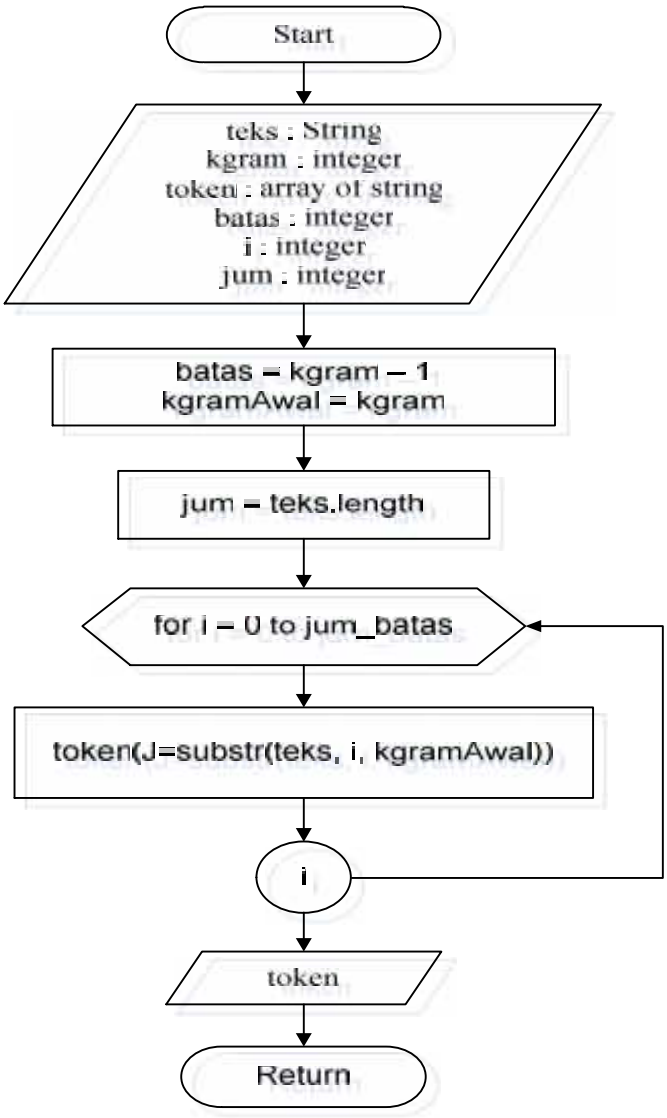
{ salah }{ alah }{ lah s }{ ah sa }{ h sat }{ satu }{ satu }{ atu d }{ tu da }{ u dam }{ damp }{ dampa }{ ampak }{ mpak }{ pak n }{ ak ne }{ k neg }{ nega }{ negat }{ egati }{ gatif }{ atif }{ tif p }{ if pe }{ f per }{ perk }{ perke }{ erkem }{ rkemb }{ kemba }{ emban }{ mbang }{ banga }{ angan }{ ngan }{ gan i }{ an il }{ n ilm }{ ilmu }{ ilmu }{ lmu t }{ mu te }{ u tek }{ tekn }{ tekno }{ eknol }{ knolo }{ nolog }{ ologi }{ logi }{ ogi i }{ gi in }{ i inf }{ info }{ infor }{ nform }{ forma }{ ormas }{ rmasi }{ masi }{ asi m }{ si ma }{ i mar }{ mara }{ marak }{ arakn }{ rakny }{ aknya }{ knya }{ nya t }{ ya ti }{ a tin }{ tind }{ tinda }{ indak }{ ndaka }{ dakan }{ akan }{ kan p }{ an pl }{ n pla }{ plag}{plagi}{lagia}{ agiar }{ giari }{ iaris }{ arism }{ risme }{ isme }

Hasil *k-gram* dokumen uji yang telah melalui proses tokenisasi (proses 2.1)

{ plag}{plagi}{lagia}{agiar}{giari}{iaris}{arism}{risme}{isme }{ sme m }{ me me }{ e mer }{ meru }{ merup }{ erupa }{ rupak }{ upaka }{ pakan }{ akan }{ kan d }{ an da }{ n dam }{damp }{ dampa }{ ampak }{ mpak }{pak n}{akne}{kneg}{nega}{negat}{egati}{gatif}{atif}{tifp}{ifpe}{fper}{perk}{perke}{erkem}{rkemb}{kemba}{emban}{mbang}{banga}{angan}{ngan}{gant}{ante}{ntek}{tekn}{tekno}{eknol}{knolo}{nolog}{ologi}{logi }{ogi i}{gi in}{iinf}{ info}{infor}{nform}{forma}{ormas}{rmasi}{masi }

Pada contoh *parsing k-gram* diatas adalah proses pembentukan pola kata dalam bentuk *gram* dengan panjang karakter k=5, sehinga isi kalimat pada dokumen asli dan dokumen uji dirubah dalam bentuk *kgram.*

Berikut ini adalah*flowchart* dari proses *parsing k-gram*.Proses *parsing k-gram* digambarkan pada sebuah*flowchart* berikut ini :



Gambar 4.9 Sub Proses*Tokenizing* yang menghasilkan *k-gram*

4.4.2.2.2 Proses *Hashing*

Kemudian dilakukan proses *hashing*, dimana pada proses ini *hash* berfungsi untuk mengkonvert setiap string menjadi bilangan. Dengan cara mengalikan nilai *ASCII* hasil huruf hasil *k-gram* dengan basis bilangan tertentu, dimana basis bilang merupakan bilangan prima. Dengan menggunakan persamaan (2.1), maka dapat

ditung hasil *hash*nya. Metode *Rabin-Karp* didasarkan pada fakta jika dua buah string sama maka harga *hash value*nya pasti sama. Sebagai contoh kita ambil kata dari hasil *k-gram* yang pertama pada dokumen asli, yaitu kata {salah}.

Contoh proses *hashing* untuk menghitung nilai *hash* dari kata {salah}, dengan nilai k= 5 dan b=7

Nilai *ASCII* dari kata {salah}

ASCII(s)= 115

ASCII(a)= 97

ASCII(l)= 108

ASCII(h)= 104

$H_{(C1 \ldots. CK)} = C_1.b^{(K-1)} + c_2.b^{(K-2)} + \ldots.. + C_{(k-1)}.b^{(k)} + ck$

$H = (115.7^4)+(97.7^3)+(108.7^2)=(\ 97.7^1)+(\ 104.7^0)$

H = (115.2401)+(97.343)+(108.49)+(97.7)+(104.1)

H= 276115+33271+5292+679+104

H= 315461

Jadi nilai *hash* pada dokumen asli *k-gram* yang pertama adalah 315461, proses perhitungan nilai *hash* di ulang kembali hingga *k-gram* keseluruhan dihitung. Berikut ini adalah nilai hasil *hash* dari dokomen asli dan *hash* dokumen uji:

Hasil *hashing* dokumen asli:

```
{315461} {275454} {298014} {271039} {267110} {121959}
{315921} {278742} {321012} {297581} {116760} {279593}
{276558} {305659} {307760} {272037} {274083} {120329}
{304595} {283500} {287095} {278576} {319865} {289544}
{262187} {121102} {309991} {287662} {316225} {297674}
{285479} {300949} {274777} {276463} {304994} {286293}
{273038} {281096} {119019} {295341} {302768} {304321}
{298391} {122428} {319283} {285477} {300943} {308355}
{309820} {303195} {307314} {285731} {269098} {119062}
{295724} {305442} {289421} {311748} {316764} {301382}
{277820} {314558} {269215} {119867} {301352} {277611}
{313119} {275932} {301277} {310706} {326277} {250402}
{122635} {320718} {295521} {304009} {279403} {275153}
{295904} {273087} {281427} {121322} {311535} {298458}
{274164} {288974} {291812} {278058} {316228} {297630}
```

Jumlah *hash* pada dokumen asli adalah= $\sum H_{asli}$= 90

Hasil *hashing* dokumen uji:

```
{6982} {13275} {12706} {11988} {12498} {12580} {12334}
{13532} {12926} {13372} {12272} {10443} {6903} {13045}
{12745}  {13799}  {13792}  {13055}  {11981}  {12472}
{11512} {11074} {6604} {12133} {12206} {13079} {12860}
{11465} {10927} {6877} {12971} {12288} {12423} {12272}
{13357} {11984} {10551} {6974} {13247} {12634} {13457}
{12766}  {12407}  {12781}  {11953}  {12155}  {12926}
{12164} {11564} {11228} {7064} {13527} {12501} {13071}
{13315} {13320} {13019} {12918} {11891} {10746} {6834}
{12840} {13114} {12709} {13456} {13500} {12830}
```

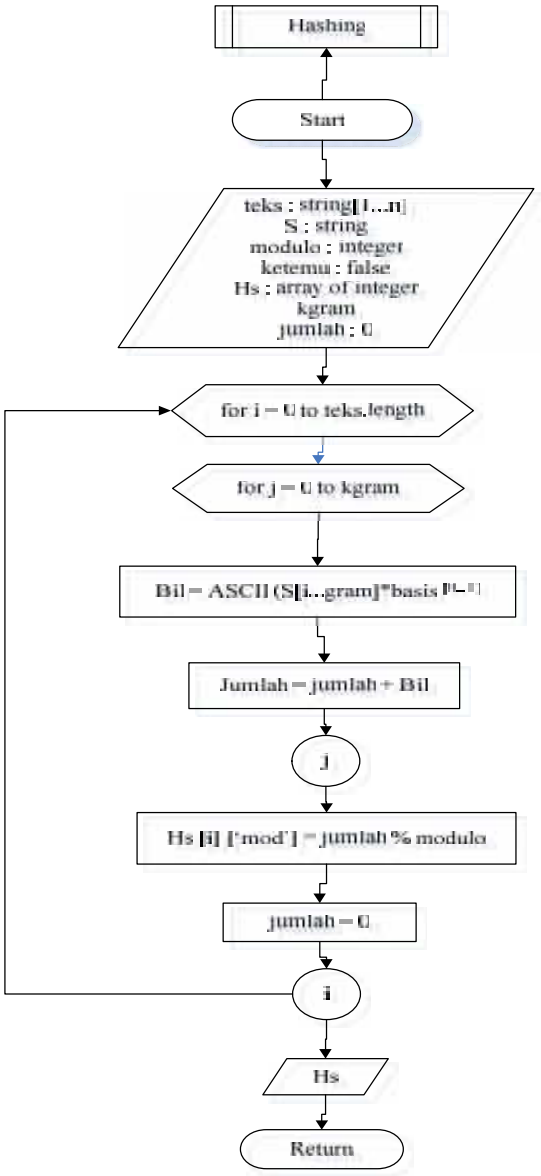
Jumlah *hash* pada dokumen uji adalah= $\sum H_{uji}$= 62

Hash yang sama:

```
{116760} {279593} {276558} {305659} {307760} {272037}
{274083} {120329} {304595} {283500} {287095} {278576}
{319865} {289544} {262187} {121102} {309991} {287662}
{316225} {297674} {285479} {300949} {274777} {276463}
{304994} {122428} {319283} {285477} {300943} {308355}
{309820} {303195} {307314} {285731} {269098} {119062}
{295724} {305442} {289421} {311748} {316764} {301382}
{275153} {121322} {311535} {298458} {274164} {288974}
{291812} {278058} {316228} {297630}
```

Jumlah *hash* yang sama dari dokumen asli dan dokumen uji adalah $= \sum H_{asli} \cap H_{uji} = 52$

Gambaran proses *hashing* seperti yang telihat pada *flowchart* berikut ini.



Gambar 4.10 Sub Proses*Tokenizing* menghasilkan *Hash*

Setlah jumlah *hash* diketahui, yaitu *hash* pada dokumen asli sebanyak 90, *hash* dokumen uji sebanyak 67 dan *hash* yang sama sebanyak 52, maka proses selanjutnya adalah menghitung *similarity.*

4.4.3 Deteksi Kesamaan

Setelah diketahui nilai *hashny*, jumlah *hash* pada dokumen asli 90, jumlah *hash* pada dokumen uji 67 dan *hash* yang sanma pada kedua dokumen 52. Proses selanjutnya adalah menghitung *similarity* yaitu tingkat kesamaan dua dokumen dengan menggunakan persamaan (2.2), yaitu barapa persen tingkat kasamaannya. *Similarity* didapat dari hasil *hash* yang sama di bagi dengan jumlah *hash* kedua dokumen dikali dengan seratus persen. Berikut ini adalah proses menghitung *similarity* dua dokumen di atas:

Similarity (asli, uji) $= \frac{\sum H\ asli \cap \sum H(uji)}{\sum H\ asli \cup \sum H(uji)} \times\ 100\ \%$

$= \frac{\sum Hasli \cap Huji}{\sum H\ asli + \sum H\ uji - \sum Hasli \cap uji} \times\ 100\ \%$

$= \frac{52}{90 + 67 - 52} \times\ 100\ \%$

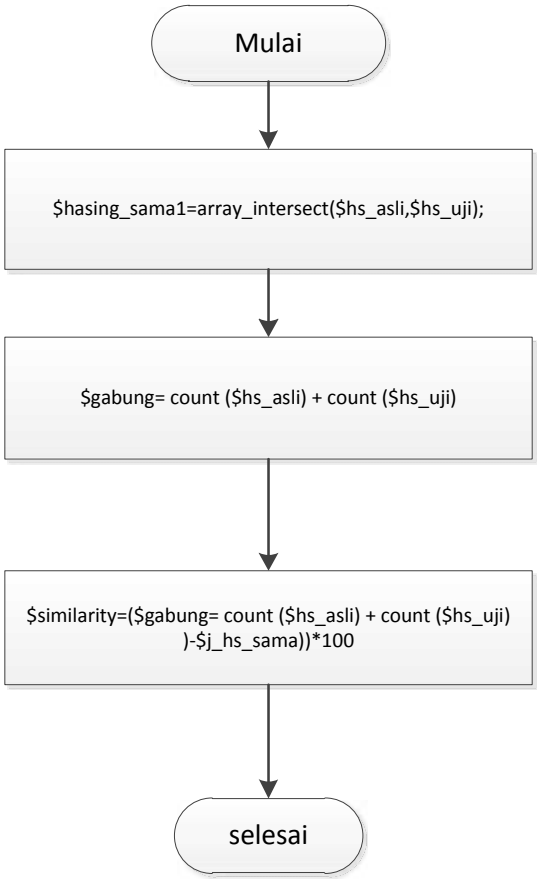
$= \frac{52}{157 - 52} \times\ 100\%$

$= \frac{52}{105} \times\ 100\ \%$

$= 49{,}52\ \%$

Hasil perhitungan *similaritry* dari kedua dokumen diatas adalah 49,52%, dokumen yang diuji bias dikatakn plagiat, karena tingkat kesamaannya sangat besar.

Gambaran proses deteksi kesamaan dokumen seperti yang telihat pada *flowchart* berikut ini.



Gambar 4.11 proses Menghitung *Similarity*

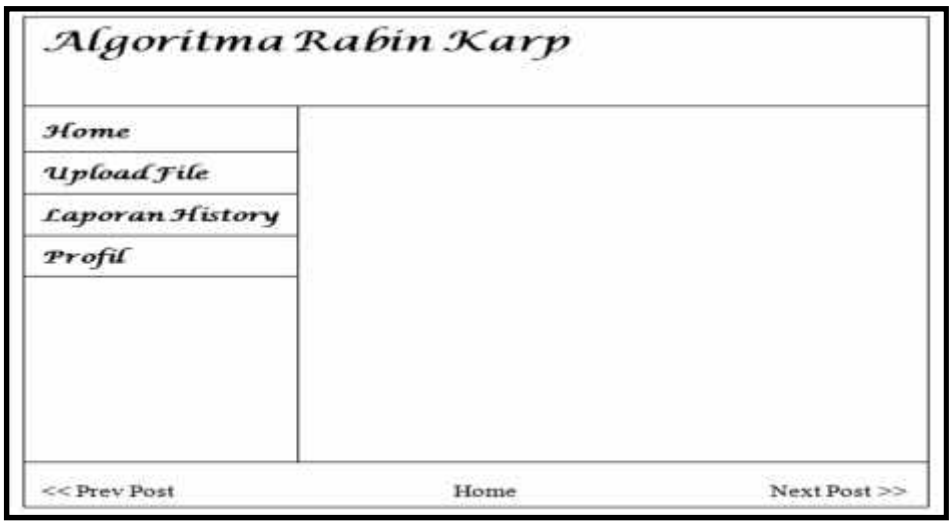
4.5 Perancangan Desain Input dan Output Program

Pada perancangan desain sistem ini, digambarkan desain tampilan sistem yang akan dibuat dan diimplementasikan pada sistem.

4.5.2 Desain Utama

Rancangan untuk desain tampilan halaman utama seperti yang terlihat pada gambar 4.11 berikut. Halaman utama dirancang dengan beberapa menu yang dapat

diakses, terdiri dari halaman*home* atau halaman utama itu sendiri, halaman *upload file* yaitu halaman yang dirancang untuk meng*upload* file yang akan diuji, halaman laporan *history* merupakan halaman yang dirancang untuk menampilkan *history* dari file-file yang sudah pernah di*upload* dan diuji, dan halaman profil berisi *profile* dari penulis.



Gambar 4.12 Desain Menu Utama

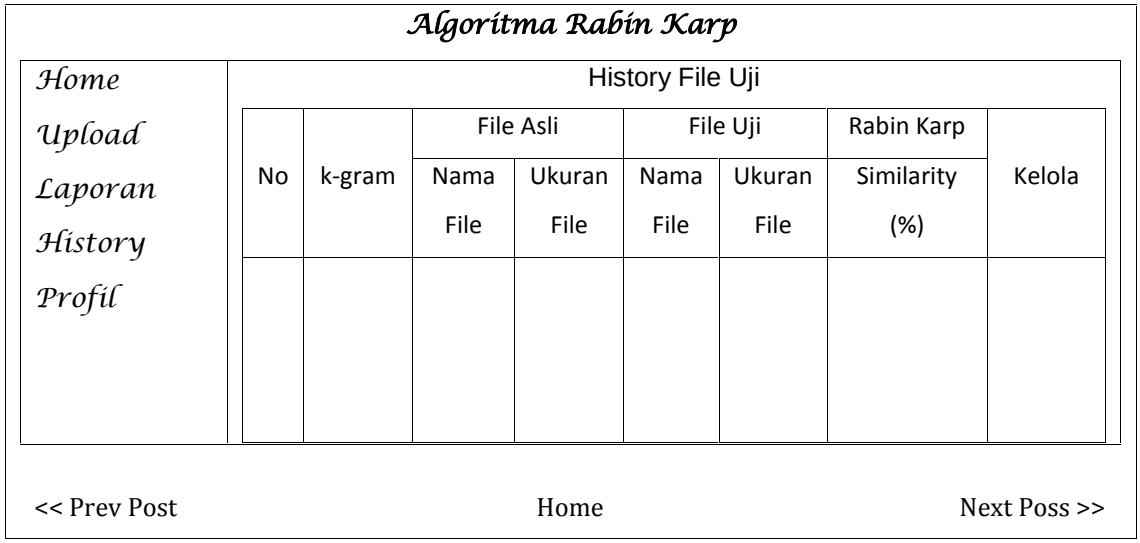
4.5.3 Desain Input

Rancangan desain *input* pada sistem yang akan dibuat, terdapat pada halaman *upload file*. Dimana pada halaman ini dirancang untuk dapat mengambil dan meng*upload* file yang akan diuji, seperti yang terlihat pada gambar 4.12 berikut.

Gambar 4.13 Desain *Input*

4.5.4 Desain Output

Pada rancangan desain *output*nya akan ditampilkan hasil dari file yang sudah diuji. File ditampilkan dengan dilengkapi keterangan dari masing-masing file seperti yang terlihat pada gambar 4.13 berikut.



Gambar 4.14 Desain Laporan History

Dari laporan histori seperti yang dirancang pada desain diatas, pada kolom kelola akan terdapat link lihat kembali dan link hapus. Pada link lihat kembali fungsinya untuk melihat kembali dokumen teks yang sudah dilakukan pengujian.Adapun desain dari halaman tersebut seperti yang terlihat pada gambar 4.14. Pada halaman tersebut ada beberapa link yang menampilkan keterangan hasil pencarian seperti hasil *k-gram*, *hashing*, *hashing*sama dan *similarity*.

Laporan Histori Uji File
Algoritma Rabin Karp

StringK-GramHashingHashing SamaSimilarity

KETERANGAN	FILE ASLI	FILE UJI
Info File	Ukuran File : Type File : Jumlah Kata :	Ukuran File : Type File : Jumlah Kata :
Isi String		
Case Folding		
Tokenizing		
Filtering		

Gambar 4.15 Desain *Output* Hasil Uji Dokumen Teks

BAB V

IMPLEMENTASI DAN PENGUJIAN

Implementasi merupakan penerapan hasil perancangan sebuah aplikasi yangsiap untuk dioperasikan. Pengujian pada aplikasi dilakukan untuk mengetahuikelebihan serta kekurangan dari aplikasi sehingga mampu memberi gambaranuntuk proses pengembangan selanjutnya.

5.1 Lingkungan Implementasi

Tahap implementasi sistem merupakan tahap menerjemahkanperancangan berdasarkan hasil analisis ke bahasa yang dapat dimengerti olehmesin, serta menerapkan perangkat lunak pada keadaan yang sesungguhnya.Lingkungan implementasi meliputi lingkungan perangkat lunak(*software*) dan lingkungan perangkat keras(*hardware*).

5.2 Implementasi Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistemdeteksi kesamaan dua dokumen ini adalah:

1. Sistem Operasi Windows XP SP2
2. XAMPP 1.6.3
3. PHP Version 5.2.3
4. Apache 2.0
5. MySQL 5.0.45
6. PHP Designer 7.0
7. Mozilla Firefox 3.6

5.2.1 Implementasi Perangkat Keras

Dalam perancangan dan pengembangan sistem deteksi kesamaan dua dokumen ini menggunakan komputer (PC) dengan spesifikasi sebagai berikut :

Perangkat Keras	Spesifikasi
Processor	Intel pentium processor T4200 2.00 GHz
Memory	1 GB
Harddisk	320 GB
VGA	VGA 256 ATI Radeon X1050
Monitor	17”
Keyboard, Mouse	-

5.3 Implementasi *User Interface*

Berangkat dari perancangan yang sudah dilakukan pada bab sebelumnya, kemudian diimplementasikan kedalam sebuah sistem. Pada tampilan halaman utama pada *User Interface*terdapat beberapa menu seperti *Home*, *Upload* Bahan, Laporan *History* dan Profil. Pada halaman *Home* berisi tentang kilasan pengertian algoritma *Rabin Karp*.Adapun tampilannya seperti yang terlihat pada gambar 5.1 berikut.

Gambar 5.1 Tampilan Halaman *Home*

Berikut ini adalah tampilan halaman *upload* bahan dokumen yangakan diuji berupa file dokumen dengan ekstensi.txt.atau .pdf.Kedua file yang akan diuji, baik file asli maupun file uji di*upload* dari form ini. Sebelumnya jangan lupa untuk menentukan basis bilangan dan *k-gram*nya.

Gambar 5.2 Tampilan Halaman *Upload File*

Pada halaman laporan *history*, ditampilkan dokumen-dokumen yang sudah dilakukan pengujian. Pada halaman ini akan terlihat *similarity*nya atau tingkat persentase kesamaan antara kedua dokumen yang dibandingkan. Jika hasil *similarity*nya 0,00persen, maka itu berarti kedua file dokumen tersebut jelas berbeda, dan tidak ada kesamaan dokumen. Dengan kata lain dokumen tersebut bukan termasuk plagirisme. Tetapi ketika hasil *similarity* dari kedua dokumen yang dibandingkan lebih dari 0,00 persen maka sudah dipastikan bahwa dokumen tersebut ada kesamaannya. Jika hasil *similarity* mendekati 100 persen, itu tandanya dokumen tersebut banyak ditemukan kesamaannya, dan itu berarti dokumen tersebut termasuk hasil plagiarisme.Berikut merupakan tampilan dari halaman laporan *history*.

Gambar 5.3 Tampilan Halaman *Laporan History*

5.4 Uji Coba Sistem

Tahap pengujian sistem ini merupakan tahap yang dilakukan setelah tahap implementasi selesai.Tujuan tahap pengujian ini adalah untuk menjamin apakah aplikasi yang sudah dibangun telah sesuai dengan analisa dan perancangan yang dilakukan sehingga tujuan yang diinginkan tercapai.

5.4.1 Uji Dokumen Dengan Nilai K-gram yang Sama

Pada pengujian ini, dilakukan pengujian terhadap dua dokumen dengan *k-gram* yang bersifat tetap atau sama dan basis bilangan prima yang berbeda-beda. Dengan file asli dan file uji berekstensi.*txt*. Berikut adalah hasil pengujian dari dua dokumen teks dengan ekstensi *.txt*.

Pengujian 1.1

Dokumen **asli_dokumen.txt**

```
"Plagiarisme" merupakan dampak negatif dari perkembangan teknologi informasi.
```

Dokumen **uji_dokumen.txt**

```
"Plagiarisme" sangat merebak seiring dengan pesatnya perkembangan teknologi informasi.
```

Pengujiaan 1.2

Dokumen **IR-1.txt**

```
Information Retrieval adalah studi tentang sistem pengindeksan, pencarian, dan mengingat data, khususnya teks atau bentuk tidak terstruktur lainnya. (Manning, 2009).
```

Dokumen **IR-2.txt**

```
Information Retrieval adalah “bidang di persimpangan ilmu informasi dan ilmu komputer.  Berkutat dengan pengindeksan dan pengambilan informasi dari sumber informasi heterogen dan sebagian besar-tekstual.Istilah ini diciptakan oleh Mooers pada tahun 1951, yang menganjurkan bahwa diterapkan ke aspek intelektual deskripsi informasi dan sistem untuk pencarian”.  (Hersh, 2003)
```

Pengujian 1.3

Dokumen **monitoring-1.txt**

```
Monitoring merupakan program yang terintegrasi, bagian penting dipraktek manajemen yang baik dan arena itu merupakan bagian integral di manajemen sehari-hari.
```

Dokumen **monitoring-2.txt**

```
Monitoring sebagai suatu proses mengukur, mencatat, mengumpulkan, memproses dan mengkomunikasikan informasi untuk membantu pengambilan keputusan manajemen program/proyek.
```

Untuk hasil *k-gram* dan hasil *hashing* pengujian 1 lihat lampiran A

Tabel 5.1 Pengujian Dokumen Dengan K-gram yang Sama

No.	Dokumen	*K-gram*	Basis Bilangan	*Hash* Sama	*Hash* Beda		*Similarity* (%)
1.	asli_dokumen.txt uji_dokumen.txt	5	3	39	67	75	37.86 %
2.	asli_dokumen.txt uji_dokumen.txt	5	5	39	67	75	37.86 %
3.	asli_dokumen.txt uji_dokumen.txt	5	7	39	67	75	37.86 %
4.	asli_dokumen.txt uji_dokumen.txt	5	11	39	67	75	37.86 %
5.	IR-1.txt IR-2.txt	7	5	36	136	308	8.82 %
6.	IR-1.txt IR-2.txt	7	7	36	136	308	8.82 %
7.	IR-1.txt IR-2.txt	7	11	36	136	308	8.82 %
8.	monitoring-1.txt monitoring-2.txt	6	5	23	134	152	8.75 %
9.	monitoring-1.txt monitoring-2.txt	6	7	23	134	152	8.75 %

10.	monitoring-1.txt monitoring-2.txt	6	11	23	134	152	8.75 %

Dari tabel hasil pengujian diatas, dapat simpulkan bahwa :

1. Pengujian yang dilakukan dengan nilai *k-gram* yang sama, maka nilai *similarity* yang dihasilkan juga akan sama.
2. Itu berarti bahwa meskipun nilai basis bilangan prima berubah-ubah sementara nilai *k-gram*nya masih tetap atau sama, maka hal itu tidak berpengaruh pada hasil nilai *similarity*-nya.
3. Besar kecilnya nilai basis bilangan prima hanya berpegaruh pada saat proses konversi dari nilai *k-gram* ke nilai *hashing*.
4. Hasil yang paling bagus pada pengujian ini adalah dengan menentukan nilai *k-gram* dan basis bilangan yang tidak terlalu kecil dan tidak terlalu besar. Pada pengujian ini yang paling bagus adalah nilai *k-gram* : 6 dan nilai basis bilangan prima : 7 dan *similarity* yang dihasilkan adalah 8.75 %.
5. Hasil pengujian yang buruk pada tabel diatas adalah dengan menentukan nilai *k-gram* yang terlalu kecil dan terlalu besar. Pada tabel pengujian diatas hasil yang paling buruk dengan nilai *k-gram* : 5 dan 7.

5.4.2 Uji Dokumen Dengan Nilai K-gram yang Berbeda

Pada pengujian kali ini, dilakukan pengujian dengan nilai *k-gram* yang berbeda-beda, tetapi nilai basis bilangan primanya sama atau tetap. Maka hasil pengujian pada beberapa dokumen teks adalah sebagai berikut :

Pengujian 2.1

Dokumen GIS-1.txt

```
GIS singkatan dari Geographic Information System atau Sistem Informasi Geografi. GIS merupakan suatu alat yang dapat digunakan untuk mengelola (input, manajemen, proses, dan output) data spasial atau data yang berefernsi geografis.
```

Dokumen GIS-2.txt

```
Alat yang dapat digunakan untuk mengelola (input, manajemen, proses, dan output) data spasial atau data yang berefernsi geografis disebut GIS. Geographic Information System merupakan kepanjangan dari GIS atau Sistem Informasi Geografi.
```

Untuk hasil *k-gram* dan hasil *hashing* pengujian 2 lihat lampiran B

Tabel 5.2 Pengujian Dokumen Dengan Basis Bilangan yang Sama

No.	Dokumen	*K-gram*	Basis Bilangan	*Hash* Sama	*Hash* Beda		*Similarity* (%)
					D 1	D 2	
1.	GIS-1.txt GIS-2.txt	5	3	165	197	201	70.82 %
2.	GIS-1.txt GIS-2.txt	8	3	150	194	198	61.98 %
3.	GIS-1.txt GIS-2.txt	11	3	138	191	195	55.65 %
4.	GIS-1.txt GIS-2.txt	5	5	163	197	201	69.36 %
5.	GIS-1.txt GIS-2.txt	8	5	150	194	198	61.98 %
6.	GIS-1.txt	11	5	138	191	195	55.65 %

	GIS-2.txt						
7.	GIS-1.txt GIS-2.txt	5	11	163	197	201	69.36 %
9.	GIS-1.txt GIS-2.txt	8	11	150	194	198	61.98 %
10.	GIS-1.txt GIS-2.txt	11	11	138	191	195	55.65 %

Dari tabel hasil pengujian pada beberapa dokumen diatas, dapat simpulkan bahwa :

1. Nilai *k-gram* yang berbeda-beda berpengaruh pada hasil *similarity* yang berbeda-beda pula, meskipun nilai basis bilangan primanya tetap.
2. Semakinkecil nilai *k-gram* maka semakin besar persentase kemiripan (*similarity*) yang dihasilkan.
3. Besar kecilnya nilai *k-gram* berpengaruh pada saat proses pemecahan string ke bentuk *k-gram*.
4. Hasil yang paling bagus pada pengujian ini adalah dengan menentukan nilai *k-gram* dan basis bilangan yang tidak terlalu kecil dan tidak terlalu besar. Pada pengujian ini yang paling bagus adalah nilai *k-gram* : 8 dan nilai basis bilangan prima : 5 dan *similarity* yang dihasilkan adalah 61.98 %.
5. Hasil pengujian yang buruk pada tabel diatas adalah dengan menentukan nilai *k-gram* yang terlalu kecil dan terlalu besar. Pada tabel pengujian diatas hasil yang paling buruk dengan nilai *k-gram* : 5 dan 11.

5.4.3 Uji Dokumen Dengan Mengubah Tata Bahasa

Pada pengujian ini, dokumen yang ada diubah terlebih dahulu susunan kalimatnya sehingga secara tata bahasa berubah namun memiliki maksud yang sama.

Berikut adalah isi dokumen asli yang belum diubah diuji dengan dokumen yang sudah diubah tata bahasanya.

Pengujian 3.1

Dokumen **dokumen_asli.txt**

```
Algoritma Rabin Karp adalah algoritma pencarian kata yang mencari sebuah pola berupa substring dalam sebuah teks menggunakan hashing. Algoritma ini sangat efektif untuk pencocokan kata dengan pola banyak.Salah satu aplikasi praktis dari algoritma Rabin Karp adalah dalam pendeteksian plagiarisme.
```

Dokumen **uji_dok_ubah.txt**

```
Algoritma pencarian kata untuk mencari sebuah pola berupa substring dalam sebuah teks menggunakan hashing disebut dengan Algoritma Rabin Karp. Pencocokan kata dengan pola banyak sangat efektif menggunakan algoritma ini. Pendeteksian plagiarisme merupakan salah satu aplikasi praktis dari algoritma rabin karp.
```

Pengujian 3.2

Dokumen **token.txt**

```
Tokenizing adalah proses penghilangan tanda baca pada kalimat sehingga menghasilkan kata-kata yang berdiri sendiri-sendiri.
```

Dokumen **token_ubah.txt**

```
Proses penghilangan tanda baca pada kalimat yang ada dalam dokumen sehingga menghasilkan kata-kata yang berdiri sendiri disebut sebagai proses tokenizing.
```

Pengujian 3.3

Dokumen **komputer.txt**

```
Komputer adalah alat yang dipakai untuk mengolah data menurut prosedur yang telah dirumuskan. Kata computer semula dipergunakan untuk menggambarkan orang yang perkerjaannya melakukan perhitungan aritmatika, dengan atau tanpa alat bantu, tetapi arti kata ini kemudian dipindahkan kepada mesin itu sendiri.
```

Dokumen **komputer_ubah**.txt

```
Alat untuk mengolah data menurut prosedur yang telah dirumuskan disebut sebagai komputer. orang yang pekerjaannya melakukan perhitungan aritmatika, dengan atau tanpa alat bantu, semula digambarkan dengan kata computer. tetapi arti kata ini kemudian dipindahkan kepada mesin itu sendiri.
```

Pengujian 3.4

Dokumen **dok_gis**.txt

```
GIS singkatan dari Geographic Information System atau Sistem Informasi Geografi. GIS merupakan suatu alat yang dapat digunakan untuk mengelola (input, manajemen, proses, dan output) data spasial atau data yang berefernsi geografis.
```

Dokumen **dok_gis_ubah**.txt

```
Geographic Information System atau Sistem Informasi Geografi merupakan kepanjangan dari GIS. Suatu alat yang dapat digunakan untuk mengelola (input, manajemen, proses, dan output) data spasial atau data yang berefernsi geografis merupakan bagian dari GIS.
```

Untuk hasil *k-gram* dan *hashing* pengujian 3 lihat lanpiran C

Tabel 5.3 Pengujian Dokumen yang Diubah Tata Bahasanya

No.	Dokumen	*K-gram*	Basis Bilangan	*Hash* Sama	*Hash* Beda		*Similarity* (%)
					D 1	D 2	
1.	dokumen_asli.txt uji_dok_ubah.txt	5	3	226	240	488	45.02 %
2.	token.txt token_ubah.txt	6	11	110	115	129	82.09 %
3.	case.txt case_ubah.txt	7	7	98	104	138	68.06 %
4.	komputer.txt komputer_ubah.txt	9	3	162	244	236	50.94 %
5.	dok_gis.txt dok_gis_ubah.txt	10	5	153	189	205	63.49 %

Dari tabel pengujian diatas, dapat disimpulkan bahwa :

1. Aplikasi ini tetap bisa mendeteksi kesamaan yang ada pada dokumen teks meskipun susunan kalimatnya diubah, sehingga tata bahasanya pun berubah.
2. Hasil yang paling bagus pada pengujian ini adalah dengan menentukan nilai *k-gram* dan basis bilangan yang tidak terlalu kecil dan tidak terlalu besar. Pada pengujian ini yang paling bagus adalah nilai *k-gram* : 7 dan nilai basis bilangan prima : 7 dan *similarity* yang dihasilkan adalah 68.06 %.
3. Hasil pengujian yang buruk pada tabel diatas adalah dengan menentukan nilai *k-gram* yang terlalu kecil dan terlalu besar. Pada tabel pengujian diatas hasil yang paling buruk dengan nilai *k-gram* : 5 dan 10.

5.4.4 Uji Dokumen yang Tidak Memiliki Kesamaan

Pengujian kali ini, dilakukan pengujian terhadap dua dokumen yang tidak memiliki kesamaan. Adapun hasil pengujiannya adalah sebagai berikut :

Pengujian 4.1

Dokumen **implementasi.txt**

```
Implementasi merupakan penerapan hasil perancangan sebuah aplikasi siap untuk dioperasikan. Pengujian pada aplikasi dilakukan untuk mengetahui kelebihan serta kekurangan dari aplikasi sehingga mampu memberi gambaran untuk selanjutnya.
```

Dokumen **ilmu_komputer.txt**

```
Pencarian string udah familiar dalam kehidupan dalam bidang ilmu komputer. Kita lihat saja berbagai kasus yang muncul, contohnya proses pembuatan makalah.
```

Pengujian4.2

Dokumen **case.txt**

```
Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf “a” sampai dengan huruf “z” yang diterima.
```

Dokumen **komputer.txt**

```
Komputer adalah alat yang dipakai untuk mengolah data menurut prosedur yang telah dirumuskan. Kata computer semula dipergunakan untuk menggambarkan orang yang perkerjaannya melakukan perhitungan aritmatika, dengan atau tanpa alat bantu, tetapi arti kata ini kemudian dipindahkan kepada mesin itu sendiri.
```

Pengujian 4.3

Dokumen **plagiarisme.txt**

```
Pengambilan karangan atau pendapat orang lain dan menjadikan seolah-olah karangan atau pendapatnya sendiri disebut sebagai plagiarisme.
```

Dokumen **GIS-1.txt**

```
GIS singkatan dari Geographic Information System atau Sistem Informasi Geografi. GIS merupakan suatu alat yang dapat digunakan untuk mengelola (input, manajemen, proses, dan output) data spasial atau data yang berefernsi geografis.
```

Pengujian 4.4

Dokumen **IR-1.txt**

```
Information Retrieval adalah studi tentang sistem pengindeksan, pencarian, dan mengingat data, khususnya teks atau bentuk tidak terstruktur lainnya. (Manning, 2009).
```

Dokumen **monitoring-2.txt**

```
Monitoring sebagai suatu proses mengukur, mencatat, mengumpulkan, memproses dan mengkomunikasikan informasi untuk membantu pengambilan keputusan manajemen program/proyek.
```

Pengujian 4.5

Dokumen **token.txt**

```
Tokenizing adalah proses penghilangan tanda baca pada kalimat sehingga menghasilkan kata-kata yang berdiri sendiri-sendiri.
```

Dokumen **uji3.txt**

```
Tindakan Penjiplakan atau plagiarisme sekarang ini
sangat merebak, apalagi dengan hadirnya teknologi yang
terus berkembang pesat. Dampak buruk dari perkembangan
teknologi tersebut dimanfaatkan oleh sebagian kalangan
untuk melakukan tindakan plagiarisme.
```

Untuk hasil *k-gram* dan hasil *hashing* pengujian 4 lihat lampiran D

Tabel 5.4 Pengujian Dokumen yang Tidak Memiliki Kesamaan

No.	Dokumen	*K-gram*	Basis Bilangan	*Hash* Sama	*Hash* Beda		*Similarity* (%)
					D 1	D 2	
1.	implementasi.txt ilmu_komputer.txt	5	11	0	194	144	0.00 %
2.	case.txt komputer.txt	6	7	0	105	247	0.00 %
3.	plagiarisme.txt GIS-1.txt	7	5	0	120	188	0.00 %
4.	IR-1.txt Monitoring-2.txt	8	3	0	135	150	0.00 %
5.	token.txt uji3.txt	9	5	0	97	216	0.00 %

Dari hasil pengujian pada tabel diatas, berapapun nilai *k-gram* dan nilai basis bilangan prima yang dimasukkan, tetaptidak ditemukan nilai *hash* yang sama sehingga hasil *similarity*-nya 0.00 %. Itu berarti bahwa antara dokumen asli dan dokumen uji, tidak terdapat kesamaan.Isi dari kedua dokumen benar-benar berbeda.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan pembahasan pada bab-bab sebelumnya maka pada bab terakhir ini penulis dapat mengambil kesimpulan diantaranya :

1. Aplikasi ini dapat digunakan untuk mendeteksi plagiarisme dengan baik, dengan mengimplementasikan metode *Rabin Karp*.
2. Jika nilai k-gram semakin kecil maka tingakat kesamaan dokumen semakin besar.
3. Besar kecilnya nilai basis bilangan tidak pengaruh terhadap tingkat kesamaan pada dokumen yang diuji.
4. Aplikasi ini dapat mendeteksi kesamaan dokumen walaupun telah dilakukan perubahan posisi teks dalam teks dokumen tersebut, meskipun mempengaruhi nilai *similarity*nya (persentase kesamaannya menurun).

6.2 Saran

Pada implementasi metode *Rabin Karp* untuk mendeteksi kesamaan pada dua dokumen yang diterapkan dalam bentuk aplikasi ini, masih memiliki banyak kekurangan karena keterbatasan kemampuan dan juga wawasan penulis yang masih kurang dalam merangcang aplikasi ini. Untuk itu ada beberapa saran yang mungkin dapat digunakan dalam pengembangan aplikasi ini untuk kedepannya.

1. Aplikasi pendeteksi kesamaan dokumen ini akan lebih baik apabila dapat menguji satu dokumen dengan banyak dokumen lainnya.
2. Pendeteksi kesamaan dokumen ini dapat dilakukan secara akurat dengan mempertimbangkan sinonim (persamaan kata) yang terdapat dalam dokumen teks.

3. Aplikasi ini akan lebih baik jika dilengkapi dengan waktu proses deteksi kesamaan dokumen pada saat pengujian.
4. Aplikasi ini akan lebih sempurna jika dapat menguji dokumen teks dalam bentuk gambar.

DAFTAR PUSTAKA

Admad Taufik, Dennis. 2012. *Sistem Pengukuran Tingkat Similaritas Dokumen*. Skripsi. Jakarta: Universitas Komputer Indonesia (UNIKOM) Jakarta.

Eldira, Entin Martiana K, Nur Rosyid M. 2011. *Web Mining Untuk Pencarian Dokumen Bahasa Inggris Menggunakan Hill Climbing Automatic Cluster*. Tugas Akhir. Surabaya: Institut Teknologi Sepuluh Nopember.

Jogiyanto. *Analisis Dan Desain*. Yogyakarta : Penerbit ANDI OFFSET, 2005.

Lesmana, David Indra. 2012. *Sistem Penilaian Otomatis Pada Jawaban Ujian Berbentuk Esai Menggunakan Metode Rabin Karp.* Tugas Akhir. Malang: Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Manning, Christopher D, Prabhakar Raghavan dan Hinrich Schütze. *An Introduction to Information Retrieval*. England: Cambridge University Press. 2009.

Munir, Rinaldi. *Algoritma Dan Pemograman*. Bandung: Informatika Bandung. 2007

Mutiara B, Sinta Agustina. 2011. *Aplikasi Anti Plagiatisme Dengan Algoritma Karp-Rabin Pada Penulisan Ilmiah Universitas Gunadarma*. Skripsi. Depok: Universitas Gunadarma Depok.

Nugroho, Eko. 2011. *Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoritma Rabin-Karp.* Tugas Akhir. Malang: Universitas Brawijaya Malang.

Pra Ramdhani, Puji. 2012. *Analisis Perbandingan Perfomansi Algoritma Zhu-Takaoka dan Algoritma Karp-Rabin Pada Pencarian Kata Di Rumah Kaca Buku Sunda*. Skripsi. Jakarta: Universitas Komputer Indonesia (UNIKOM) Jakarta.

Purwitasari,Diana. dkk . 2011. *Implementasi Deteksi Penjiplakan dengan Algoritma Winnowing Pada Dokumen Terkelompok*. Skripsi. Surabya: Institut Teknologi Sepuluh November (ITS) Surabaya.