Implementasi Algoritma Pencocokan String dalam Membangun Web Frequently Asked Questions (FAQ)

Muhammad Jafar Gundari / 13519197
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519197@std.stei.itb.ac.id

***Abstrak*— Dalam membuat sebuah website brand atau perusahaan dibuat sebuah daftar pertanyaan yang sering diajukan (FAQ) yang bertujuan untuk meningkatkan *user experience* audiens serta memperjelas informasi. Namun, tak jarang meskipun telah disediakan FAQ, tetap saja ada pertanyaan yang tak tersedia atau audiens malas untuk mencarinya satu persatu. Oleh karena itu penulis membuat sebuah website FAQ yang menerapkan algoritma pencocokan string boyermoore dimana audiens dapat mengetikkan sendiri pertanyaan tanpa harus melihat satu persatu serta pengelola website dapat mendaftarkan pertanyaan secara tidak terbatas.**

Kata kunci : pencocokan string, boyermoore, FAQ, string matching

I. PENDAHULUAN

Era digital seperti saat ini dalam membangun sebuah bisnis atau kegiatan biasanya memiliki sebuah website. Website ini sudah seharusnya informatif, nyaman dilihat dan mudah digunakan. Namun terkadang, diakibatkan tidak mungkin seluruh informasi disimpan ke dalam website, maka diperlukannya FAQ atau frequently asked questions.

Secara istilah FAQ adalah pertanyaan yang sering diajukan.Muncul masalah lain yaitu meskipun sudah tersedia FAQ pada website, masih saja terdapat pertanyaan yang tidak tercantumkan akibat FAQ jumlahnya terbatas ataupun audiens cenderung malas untuk melihat satu persatu pertanyaan yang dicantumkan pada FAQ. Oleh karena itu, diperlukannya sebuah laman yang dapat menjawab semua pertanyaan audiens serta audiens dapat menanyakannya secara langsung sesuai keinginan tanpa perlu melihat satu persatu pertanyaannya.

Satu pertanyaan audiens akan dijawab dengan satu jawaban saja yang paling sesuai. Untuk mendapat jawaban tersebut secara tepat dan cepat, digunakan algoritma pencocokan string. Algoritma pencocokan string secara umum adalah algoritma yang digunakan untuk mencari suatu kata/kalimat tertentu pada sebuah teks. Penulis kemudian berinisiatif membuat sebuah website FAQ dengan menggunakan FAQ dari website lintasarta sebagai contoh, Diharapkan dengan adanya website ini, dapat menyelesaikan permasalahan tersebut dan dapat dikembangkan lebih lagi tidak terbatas pada FAQ.

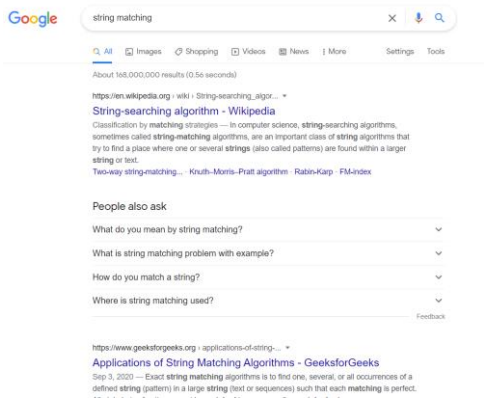
II. LANDASAN TEORI

A. *Pencocokan String*

Pencocokan string atau string matching adalah proses pencarian semua kemunculan string pendek P[0..n-1] yang disebut pattern di string yang lebih panjang T[0..m-1] yang disebut teks. Teks yaitu (long) string yang panjangnya n karakter. Sedangkan pattern yaitu (long) string yang panjangnya n karakter.

String Concepts misalkan panjang dari suatu String S adalah m. S = x0x1…xm-1. Sebuah prefix dari S adalah substring S[0..k], sebuah suffix dari S adalah substring S[k..m-1] dengan k adalah indeks apa pun antara 0 dan m-1. Misalkan diberikan sebuah string S = ‘pelajaran’. Contoh prefixnya ialah ‘p’, ‘pe’, ‘pel’, ‘pelaj’ sampai ‘pelajara’. Sedangkan suffixnya ialah ‘n’m ‘an’, ‘ran’, sampai ‘elajaran’.

Aplikasi penggunaannya dapat ditemukan pada Pencarian di dalam Editor Text, Web search engine (Misal: Google), Analisis Citra, Bionformatics (pada pencocokan rantai asam amino pada rantai DNA) dan lain sebagainya.



Gambar 1. Search engine merupakan salah satu aplikasi string mathcing

Sumber (dokumentasi Penulis)

B. *Algoritma Brute Force*

Algoritma yang paling mudah untuk diimplementasikan ialah brute force. Algoritma ini memabndingkan setiap

karakter pada pattern dengan teks secara *straight forward*. Pengecekan dilakukan pada setiap posisi karakter pattern dengan setiap posisi karakter teks sehingga membutuhkan waktu yang kuadratik.

Contoh 1:
Teks: NOBODY NOTICED HIM
Pattern: NOT

```
  NOBODY NOTICED HIM
1 NOT
2  NOT
3   NOT
4    NOT
5     NOT
6      NOT
7       NOT
8        NOT
```

Gambar 2. String matching algoritma brure forc
(Sumber : Algoritma Brute Force bagian 1 oleh Ir. Rinaldi Munir, M.T.)

Langkah-langkah dengan bruteforce sebagai berikut :
1. Mula-mula pattern disejajarkan (alignment) pada awal teks.
2. Dengan menelusuri dari kiri ke kanan pada pattern, bandingkan setiap karakter pada pattern dengan karakter yang bersesuaian di dalam teks sampai :
   a. semua karakter yang dibandingkan cocok atau sama (pencarian berhasil), atau
   b. dijumpai sebuah ketidakcocokan karakter (pencarian belum berhasil)
3. Bila pattern belum ditemukan kecocokannya dan teks belum habis, geser pattern satu karakter ke kanan dan ulangi kembali langkah 2.

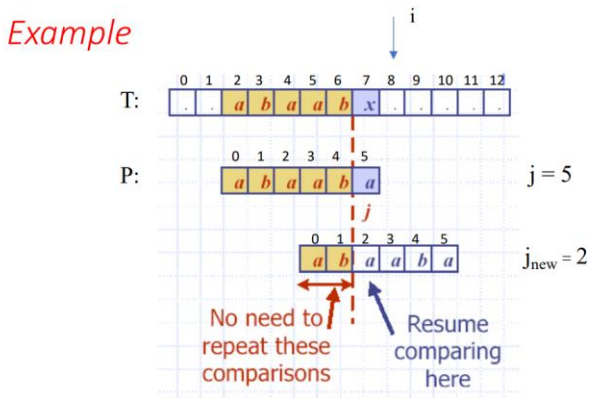
Average case untuk algoritma brute force dengan kompleksitas O(m + n). Dan jumlah perbadingan karakter pada worst case ialah jumlah perbandingan: m(n – m + 1) = O(mn)

C. Algoritma Knuth-Morris-Pratt

Dikembangkan secara terpisah oleh Donald Knuth dan Vaughan Pratt pada tahun 1970, serta secara independen oleh James H. Morris. Dipublikasikannya algoritma ini pada tahun 1977 oleh ketiga penemu tersebut. Diambil dari nama ketiga penemunya itulah nama algoritma ini lahir.

Algoritma KMP melihat pattern dengan urutan kiri ke kanan sama seperti brute force. Tetapi algoritma ini lebih pintar dengan menggeser pattern tidak selalu satu persatu.

Jika ditemukan *mismatch* diantara teks dan pattern P pada P[j], yakni T[i] ≠ P[j] maka pergeseran paling jauh ialah pada prefix terbesar dari P[0 .. j-1] yang juga merupakan suffix dari {[1 .. j-1].



Gambar 3 Pergeseran pada KMP
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

Untuk memproses KMP digunakan sebuah fungsi pinggiran. Fungsi pinggiran atau border function b(k) adalah ukuran prefix terpanjang dari P[0..k] yang juga suffix dari P[1..k]. Contoh untuk pattern P = abaaba, j = 012345. Border function biasanya dimuat kedalam sebuah table sebagai berikut

(k = j-1)

j	0	1	2	3	4	5
P[j]	*a*	*b*	*a*	*a*	*b*	*a*
k	-	0	1	2	3	4
b(k)	-	0	0	1	1	2

b(k) is the size of the largest border.

Gambar 4 Border function
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

Sebagai contoh b(4) = 2. Mengapa demikian ? Carilah prefix terbesar dari P[0..4] yang juga suffix dari P[1..4] yaitu “ab”. Ukuran “ab” adalah 2. Sehingga pencarian berikutnya dilakukan pergeseran sejauh 2 kali.

Langkah-langkah pencarian string dengan KMP sebagai berikut dengan memodifikasi algoritma brute force :
1. Mula-mula pattern disejajarkan (alignment) pada awal teks.
2. Dengan menelusuri dari kiri ke kanan pada pattern, bandingkan setiap karakter pada pattern dengan karakter yang bersesuaian di dalam teks sampai :
   a. semua karakter yang dibandingkan cocok atau sama (pencarian berhasil), atau
   b. dijumpai sebuah ketidakcocokan karakter (pencarian belum berhasil)
3. Jika terjadi mismatch pada P[j] yakni P[j] ≠ T[i] maka k = j-1 dan j = b(k). Dimaan b(k) didapat dari border function

Kompleksitas KMP sendiri ialah O(m+n) dengan detail sebagai berikut

Menghitung fungsi pinggiran O(m) Pencarian string O(n) Sangat cepat jika dibandingkan dengan algoritma brute force.

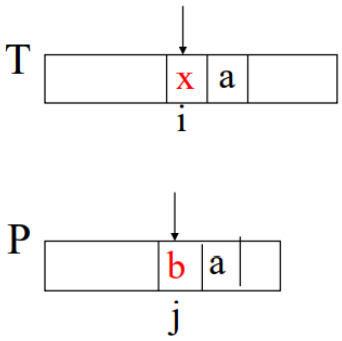
Keuntungan menggunakan algoritma KMP ialah algoritma ini tidak pernah bergerak mundur pada input teks T. Sehingga membuatnya baik untuk memroses berkas yang sangat besar yang dibaca dari perangkat luar atau melakui *network stream*.

Trade off (kekurangan) dari KMP sendiri yaitu kurang dapat bekerja dengan baik ketika jumlah aplhabetnya meningkat. Lebih banyak berpeluang mismatch dan cenderung ditemukan mismatch pada awal pattern dimana KMP lebih cepat untuk mismatch di akhir pattern.

D. Algoritma Boyer-Moore

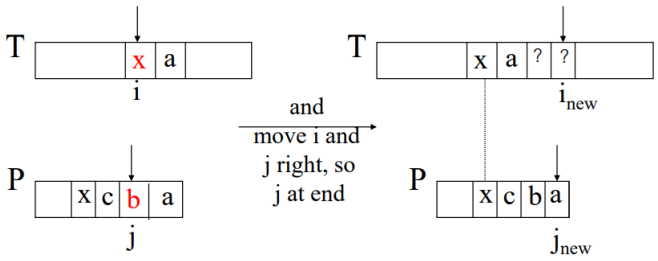
Algoritma boyer-moore pertama kali dikembangkan oleh Robert S.Boyer dan J. Strother Moore pada tahun 1977. Awal perbandingan dilakukan dimulai dari ujung kanan pattern dan bergerak ke kiri, tetapi untuk pergeseran tetap kiri ke kanan. Algoritma ini didasari oleh dua teknik

1. Teknik *Looking Glass*, yaitu teknik yang mencari pattern P pada T dengan bergerak mundur melalui P dimulai dari belakang.
2. Teknik *Character Jump*,yaitu ketika mismatch pada T[i] == x dan karakter T[i] != P[j] terdapat 3 kasus



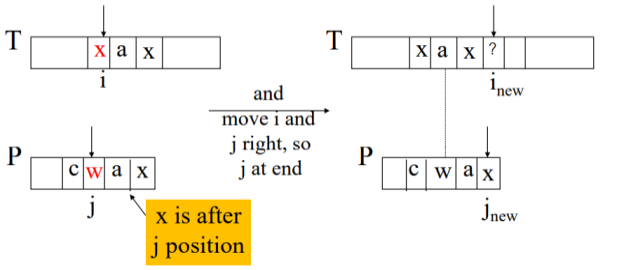
Gambar 5. Ilustrasi mismatch Character Jump
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

a. Case pertama
Jika P mengandung x dimanapun, maka lakukan pergeseran P sehingga x pada T[i] bersesuaian dengan karakter x yang terakhir muncul pada P



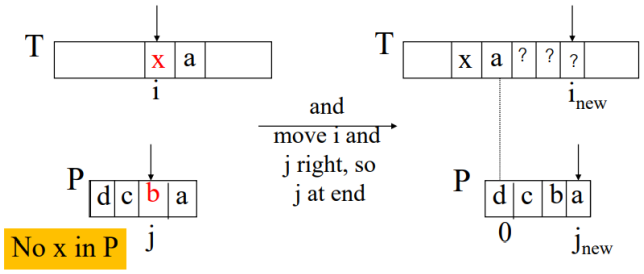
Gambar 6. Contoh kasus 1
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

b. Case kedua
Jika P mengandung x disuatu tempat, namun terletak pada sebelah kanan P[j]. Maka geser ke kanan sejauh satu karakter kepada T[i+1]



Gambar 7. Contoh kasus kedua
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

c. Case ketiga
Jika kasus pertama dan kedua tidak memenuhi, maka lakukan pergeseran sehingga P[0] bersesuaian dengan T[i+1]



Gambar 8. Contoh kasus ketiga
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

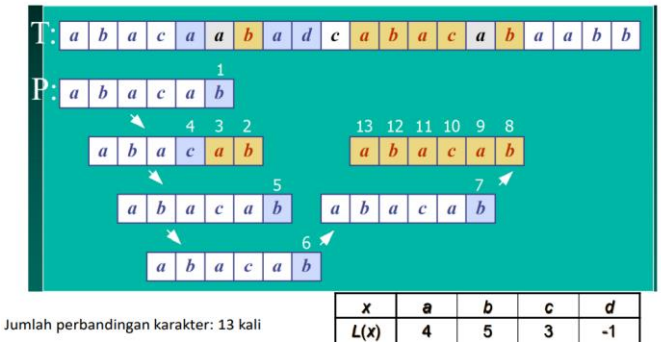
Untuk memeriksa pada teknik character jump tersebut, dibutuhkan suatu fungsi bantuan yaitu last occurrence function. Sebelum dilakukan string matching, algoritma BM memroses pattern P dan alphabet A untuk membangun last occurrence fungtion L(). L() memetakan semua huruf pada A menjadi integer. L(x), dimana x adalah huruf pada A, didefinisikan sebagai index terbesar I yang mana P[i] == x atau -1 jika tidak ditemukan pada P.
Contoh L(),
Diberikan A = { a, b, c, d }, dan P :"abacab"

P					
a	b	a	c	a	b
0	1	2	3	4	5

Didapat L sebagai berikut

x	a	b	c	d
l(x)	4	5	3	-1

d bernilai -1 karena tidak ditemukan pada P. Pada boyermoore, fungsi last occurrence ini dihitung ketika input P sudah ada. Biasanya L() disimpan kedalam sebuah array.



x	a	b	c	d
L(x)	4	5	3	-1

Gambar 9. BoyerMoore example
Sumber (Slide kuliah Pencocokan String 2021, Ir. Rinaldi Munir, M.T)

Pada gambar 9 terlihat nomor 1 terjadi mismatch, dan termasuk kasus keddua karena terdapat b pada P, namun last occurrence pada sebelah kanan. Untuk nomor 5 terjadi kasus pertama. Dan untuk nomor 6 terjadi kasus ketiga.

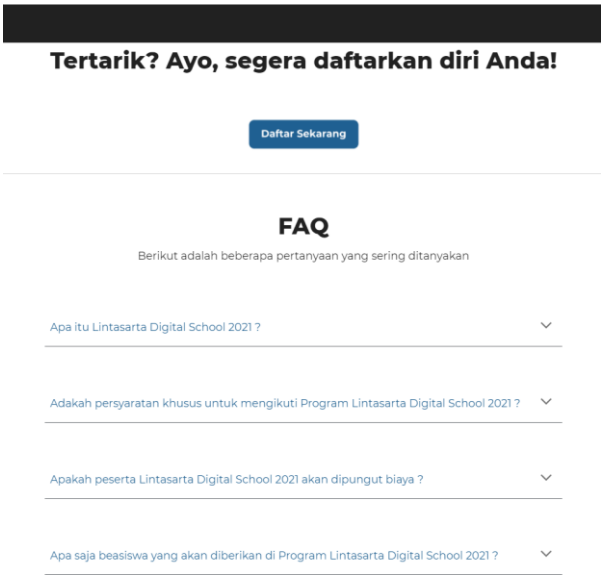
Analisis kompleksitas BM pada kasus terburuk membutuhkan waktu O(nm + A). Tetapi berbeda dengan KMP ketika A besar justru cepat, namun lambat ketika A kecil. Contoh bagus untuk teks bahasa inggris, dan buruk untuk binary. Dibandingkan dengan brute force, BM jauh lebih cepat untuk English text.

E. Frequently Asked Questions (FAQ)

Ketika membuat sebuah *event*/kegiatan/bisnis yang memiliki website tentunya harus memiliki FAQ untuk memudahkan target audience dalam mengakses event kita. Secara istilah FAQ adalah pertanyaan yang sering diajukan.Pada awalnya FAQ dibuat untuk menjawab pertanyaan terkait website. Namun saat ini FAQ banyak digunakan untuk menjelaskan lebih detail terkait produk, layanan, atau kegunaan sebuah Aplikasi/website.

Beberapa fungsi dari FAQ antara lain

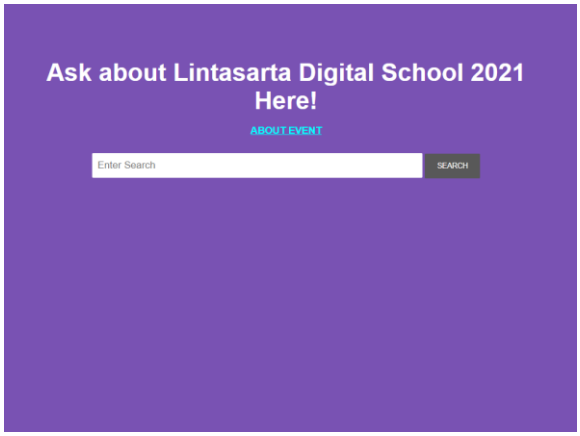
1. Meningkatkan SEO (Search engine optimization)
2. Menjawab pertanyaan general dari konsumen
3. Menghemat waktu konsumen
4. Membangun kepercayaan target audience



Gambar 10. FAQ pada website Lintasarta
Sumber (https://lintasartadigischool.dicoding.com/)

III. Implementasi dan Pengujian

Untuk mencoba menyelesaikan permasalahan kali ini, penulis membuat sebuah website sederhana dimana pengguna dapat mengetikkan pertanyaannya secara langsung tanpa terikat harus seperti apa pertanyaannya (tidak exact matching). Website kemudian akan menampilkan jawaban yang paling relevan dengan pertanyaan dari user. Website ini menggunakan Bahasa PHP sebagai server side dan HTML,CSS sederhana untuk client-side.



Gambar 11. Tampilan awal website FAQ

Sumber (Dokumentasi penulis)

Data uji FAQ diperoleh dari website https://lintasartadigischool.dicoding.com/ . yang berjumlah 12 pertanyaan dan jawaban. Meskipun tidak begitu banyak, namun penguji yakin bahwa data uji ini cukup.Data kemudian disimpan pada sebuah file resources.php dan dalam bentuk 2D associative array php.

Algoritma string matching yang digunakan penulis ialah algoritma Boyer-moore. Algoritma tersebut dipilih dikarenakan cocok sesuai dengan kelebihannya yaitu bagus untuk teks

Bahasa Indonesia (alphabet besar) dibandingkan kedua lainnya yaitu brute force dan KMP. Berikut algoritma boyermoore yang digunakan dan dibuat dalam Bahasa PHP

Penulis membuat dua fungsi pembantu yaitu calculatetabelLo untuk membuat array last occurrence dari pattern dan getLastOccurence untuk mengambil nilai L(x).

```
function calculatetabelLo($pattern)
{
  $lastoccurence = array();
  $lowerpattern = strtolower($pattern);
  for ($i = 0; $i < strlen($lowerpattern);
$i++) {
    $lastoccurence[$pattern[$i]] = $i;
  }
  return $lastoccurence;
}
```

Gambar 12. Fungsi calculatetabelLo
Sumber (Dokumentasi penulis)

```
// mengembalikan nilai L(x)
function getlastoccurence($tabelLx, $charT)
{
  if (array_key_exists($charT, $tabelLx)) {
    return $tabelLx[$charT];
  } else {
    return -1;
  }
}
```

Gambar 13. Fungsi getLastOccurence
Sumber (Dokumentasi Penulis)

```
// mengembalikan index pertama text yang match
dengan pattern, -1 jika tidak ditemukan
function boyermoore($pattern, $text)
{
  // Simpan array last occurence untuk karakter pada
pattern
  $last = calculatetabelLo($pattern);

  // Pencocokan
  $n = strlen($text);
  $panjangpattern = strlen($pattern);
  $panjangtext = strlen($text);
  if ($panjangpattern > $panjangtext) {
    return -1;
  }
  $j = $panjangpattern - 1;
  $i = $j;
  if ($j < 0) {
    return -1;}
  do {
    if ($pattern[$j] == $text[$i]) {
      if ($j == 0) {
        return $i;
      } else {
        // looking glass
        $i--;
        $j--;}
    } else {
      // character jump
      $lo = getlastoccurence($last, $text[$i]);
      $i = $i + $panjangpattern - min($j, $lo + 1);
      $j = $panjangpattern - 1;
    }} while ($i <= $n - 1);
  return -1;}
```

Gambar 14. Fungsi BoyerMoore
Sumber (Dokumentasi Penulis)

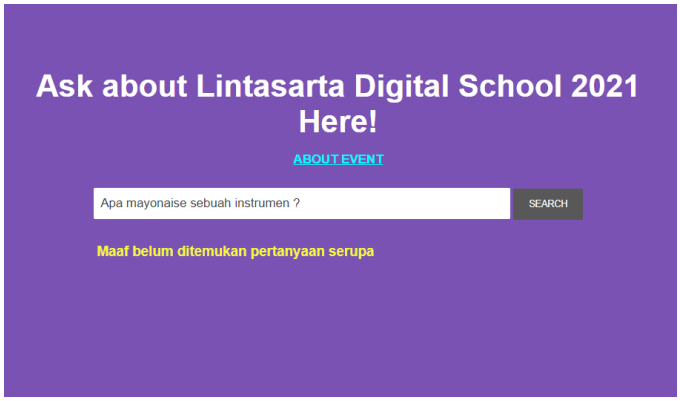
Cara penggunaan

- Dikarenakan website belum dihosting, maka pengujian dilakukan pada browser dan buka http://localhost/FAQ-matching/index.html
- Pada form pencarian “search”, ketikkan pertanyaan yang ingin ditanyakan. Tidak perlu menekan tombol search karena selagi menulis akan langsung ditampilkan rekomendasi pertanyaan dan jawaban yang sesuai.

Gambar 15. Tampilan pencarian sukses

Sumber (Penulis)

- Jika sudah terdapat pertanyaan serupa dengan pertanyaan user, maka akan menghasilkan seperti diatas. Namun jika tidak akan menampilkan seperti berikut



Gambar 16. Tampilan pencarian tidak ditemukan

Sumber (Penulis)

Cara kerja program :

1. User memasukan query pada entry form sebut sebagai variable input.
2. Jika input mengandung "?" maka ganti karakter tersebut dengan . agar tidak ikut sebagai kata.
3. Input kemudian dipisahkan perkata yang dideteksi dari adanya spasi atau " ". Kemudian disimpan pada array words
4. Untuk setiap pertanyaan pada data resources, dilakukan pengecekan kemunculan setiap kata pada words menggunakan algoritma boyermoore. Selagi dihitung jumlahnya.

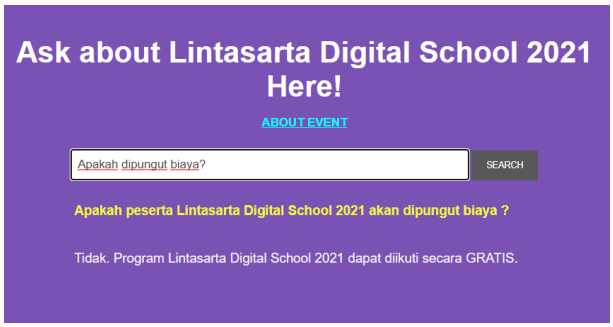
   Berikut contoh struktur pertanyaan pada data resources

```
      array(
                "question"=>"Apa itu
Lintasarta Digital School 2021 ?",
                "answer"=>"Lintasarta
Digital School 2021 adalah sebuah kegiatan
CSR Lintasarta di pilar Pintar melalui
pemberian beasiswa pelatihan coding secara
online. Tujuannya adalah untuk mencetak
programmer muda yang siap bersaing di dunia
ekonomi digital seperti saat ini. "
        ),
```

5. Input dengan jumlah kecocokan terbanyak dan kecocokan kata > 1 kata, ditampilkan sebagai hasil.
6. Jika tidak ada yang memenuhi kondisi tersebut, ditampilkan pesan tidak ditemukan.

Pengujian

1. Query : Apakah dipungut biaya?



2. Query : Beasiswa lintasarta apa aja dapetnya ?

3. Query : Siapa saya ?

IV. SIMPULAN

Algoritma pencocokan string adalah algoritma yang digunakan untuk mencari kecocokan string (pattern) pada teks. Aplikasinya terdapat pada editor text, web search engine, analisis citra, pada pencocokan rantai asam amino pada rantai DNA. Terdapat beberapa algoritma pencocokan string yaitu brute force, Knuth-Morris-Pratt dan Boyer-Moore. Dalam membuat website FAQ penulis menggunakan algoritma Boyer-Moore dengan cara kerja yaitu query dari pengguna dipisahkan perkata kemudian dengan algoritma Boyer-Moore dilakukan pencocokan ada tidaknya pertanyaan tersebut pada data website. Dengan adanya website ini audiens dapat dengan mudah bertanya dan menemukan jawabannya serta pemilik website dapat menambah FAQ sebanyak mungkin.

V. UCAPAN TERIMA KASIH

Puji syukur kepada Allah SWT. Tuhan yang Maha Esa karena atas ridha nya penulis diberi kesehatan dan kelancaran

akal untuk dapat menyelesaikan makalah yang berjudul "Implementasi Algoritma Pencocokan String dalam Membangun Web Frequently Asked Questions (FAQ)". Terima kasih pula kepada dosen Strategi algoritma 2020/2021, Dosen penulis, Bapak Dr. Ir. Rinaldi Munir, MT. yang telah memberikan ilmunya untuk menyokong dasar-dasar teori yang dibutuhkan dalam menyusun makalah ini. Semoga pembahasan mengenai makalah ini tidak berhenti disini namun dapat dikembangkan untuk lebih baik lagi. Penulis menyadari makalah ini belum sempurna sehingga penulis memohon maaf sebesar-besarnya.

LINK VIDEO YOUTUBE

https://youtu.be/Jf_WypMRGd4

REFERENSI

[1] Munir,Rinaldi."Pencocokan String". 2020. . https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf. Diakses pada 10 Mei 2021.

[2] Arif. "Kenali Apa Itu FAQ dan Fungsinya untuk Sebuah Website". https://qwords.com/blog/faq-adalah . Diakses pada 10 Mei 2021.

[3] Anonim. "Boyer Moore Algorithm for Pattern Searching". 2021. https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching/ . Diakses pada 10 Mei 2021.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 11 Mei 2021

Muhammad Jafar Gundari 13519197