Penerapan Algoritma String Matching dalam Sistem Pemberian Rekomendasi Obat

Jauza Lathifah Annassalafi - 13521030
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (13521030@std.stei.itb.ac.id):

***Abstract*—Memberikan obat yang sesuai dengan tepat sangatlah krusial dalam proses pengobatan. Setiap penyakit memiliki ciri-ciri dan gejala yang unik, sehingga memerlukan pendekatan yang spesifik dalam pemilihan obat yang tepat. Namun, dengan beragamnya obat yang tersedia, untuk membantu mengurangi jumlah obat yang perlu ditinjau, penulis mengembangkan sebuah program dengan menggunakan algoritma String Matching Boyer-Moore untuk mencocokkan gejala-gejala yang telah diidentifikasi. Dengan memanfaatkan algoritma ini, program dapat mempersempit kemungkinan obat yang perlu dipertimbangkan. Dengan demikian, program ini memberikan pendekatan yang efisien dalam memilih obat yang tepat berdasarkan gejala yang ada. Selain itu, dibuat juga opsi untuk menampilkan obat berdasarkan masukkan kandungan obat.**

Keywords—String Matching; Rekomendasi Obat; Boyer Moore; Algoritma

I. Pendahuluan

Dalam dunia medis, pemberian obat yang tepat sangatlah penting dalam proses pengobatan. Setiap penyakit memiliki karakteristik dan gejala yang berbeda, sehingga memerlukan pendekatan yang spesifik dalam pemilihan obat yang tepat. Namun, dengan banyaknya jenis obat yang tersedia, ini bisa menjadi rumit dan memakan waktu.

Untuk mengatasi tantangan tersebut di era digital saat ini, penggunaan teknologi dan pendekatan komputasional dapat memberikan kontribusi yang signifikan dalam meningkatkan efisiensi dalam sistem pemberian rekomendasi obat. Salah satu pendekatan yang digunakan adalah dengan menggunakan algoritma string matching dalam sistem pemberian rekomendasi obat.

Algoritma string matching adalah metode yang digunakan untuk mencari kecocokan pola atau substring dalam sebuah teks. Pola dapat berupa gejala-gejala penyakit yang diinputkan, sedangkan teks merupakan database yang berisi informasi tentang gejala-gejala penyakit dan obat-obat yang sesuai. Dalam konteks pemberian obat, algoritma ini dapat digunakan untuk mencocokkan gejala yang dimiliki dengan database obat yang ada, sehingga dapat memberikan rekomendasi obat yang sesuai.

Salah satu algoritma string matching yang banyak digunakan adalah algoritma Boyer-Moore. Algoritma ini memiliki keunggulan dalam menemukan kecocokan pola dengan melakukan pergeseran yang lebih cerdas berdasarkan karakter-karakter yang tidak cocok. Dengan demikian, algoritma Boyer-Moore dapat membantu mempersempit kemungkinan penyakit berdasarkan gejala yang diinputkan oleh pasien.

Pada pembuatan program, akan dirancang untuk memberikan rekomendasi obat berdasarkan masukkan gejala yang akan diolah menggunakan string matching terhadap database. Selain itu, akan dibuat opsi untuk menampilkan obat berdasarkan masukkan kandungan obat juga.

Tujuan dari makalah ini adalah untuk memberikan pemahaman yang lebih baik tentang penggunaan algoritma string matching dalam konteks sistem pemberian rekomendasi obat. Dengan pemahaman ini, diharapkan masyarakat, tenaga medis, dan pengembang sistem kesehatan dapat memanfaatkan teknologi informasi dengan lebih baik dalam mempercepat proses diagnosis dan pemberian rekomendasi obat yang tepat.

II. Landasan Teori

A. Algoritma String Matching

String Matching adalah suatu algoritma yang digunakan untuk mencari pola/pattern yang terdapat pada suatu teks. Persoalan string Matching dirumuskan sebagai berikut :

1. Sebuah teks (text), yaitu sebuah (long) string yang panjangnya n karakter.

   Contoh: “aabadbbaddced”

2. Pattern, yaitu sebuah string dengan panjang m karakter yang akan dicari dalam teks. Panjang string pattern haruslah lebih kecil bila dibandingkan dengan panjang teks.

   Contoh: “add”

Akan dicari sebuah pattren “add” pada teks “aabadbbaddced”, sehingga ditemukan pada indeks teks ke-7 (indeks dimulai dari 0).

Algoritma string matching dapat diklasifikasikan menjadi tiga bagian berdasarkan arah pencariannya, yaitu:

1. From left to right
   Bergerak dari kiri ke kanan, mengikuti arah alami membaca teks. Algoritma-algoritma yang termasuk dalam kategori ini antara lain adalah Brute Force, dan algoritma Knuth-Morris-Pratt.
2. From right to left
   Bergerak dari kanan ke kiri, arah yang umumnya menghasilkan hasil yang lebih efisien secara parsial. Contoh dari algoritma ini adalah Boyer-Moore.
3. In a specific order
   Ada algoritma yang memiliki arah pencarian yang ditentukan secara spesifik oleh algoritma tersebut, yang teoretis menghasilkan hasil yang optimal. Algoritma-algoritma yang termasuk dalam kategori ini antara lain adalah algoritma Colussi dan algoritma Crochemore-Perrin.

B. Brute Force

Algoritma Brute Force pada String Matching adalah salah satu metode untuk menemukan pola/pattern dalam sebuah teks. Cara kerjanya adalah dengan membandingkan setiap karakter dari pola/pattern dengan karakter-karakter yang berurutan dalam teks. Dimulai dari posisi awal teks, akan dibandingkan setiap karakter pola/pattern dengan karakter teks yang sesuai. Jika semua karakter sesuai, pola/pattern dianggap ditemukan.

Namun, jika ada perbedaan antara karakter pola/pattern dan karakter teks yang sedang dibandingkan, algoritma Brute Force akan ke kanan satu langkah dan memulai pencocokan ulang dari awal pola/pattern di posisi baru. Proses ini akan terus berlanjut hingga entitas pola ditemukan dalam teks atau sampai akhir teks tercapai.

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 2.2.1. String Matching Brute Force

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

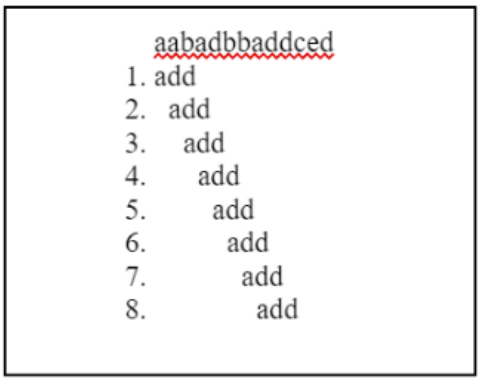
Misal diberikan teks dan pola/pattern sebagai berikut:

Teks: "aabadbbaddced"

Pattern: "add"

Berikut adalah langkah-langkah yang dilakukan menggunakan algoritma Brute Force:

1. Bandingkan setiap karakter pada pattern di bagian awal text
2. Apabila karakter pattern dan karakter pada teks tidak terjadi mismatch sampai akhir pattern, maka pencarian selesai dan pattern ditemukan dalam text.
3. Apabila karakter pattern dan karakter pada teks terjadi mismatch, maka dilakukan perbandingan karakter pattern dari awal dengan indeks karakter text pencocokan sebelumnya ditambah satu
4. Ulangi Langkah 2 dan 3.



Gambar 2.2.2. Contoh String Matching Brute Force

(Sumber: Penulis)

Algoritma brute force akan melakukan pergeseran hanya satu indeks ke kanan apabila terjadi ketidakcocokan antara karakter pattern dan teks yang sedang di bandingkan. Worst case nya adalah ketika pattern yang cocok berada pada akhir teks, seperti teks: “aaaaaaaaaaaaaaaaaaaaaaaaah” dan pattern: “aaah” dengan jumlah perbandingan $m(n - m + 1) = O(mn)$. Lalu, untuk kompleksitas kasus terbaik adalah O(n) yang terjadi apabila karakter pertama pattern tidak pernah sama dengan karakter teks yang dicocokkan, sehingga jumlah perbandingan maksimal n kali.

Jika kita ingin mencari pola yang muncul lebih dari satu kali dalam teks, kita dapat melanjutkan langkah-langkah di atas setelah menemukan pola pertama. Ini akan memungkinkan pencarian pola berikutnya yang mungkin muncul setelah pola pertama.

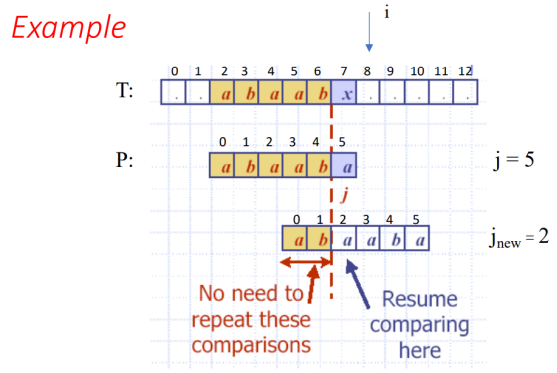
C. Algoritma Knuth-Moris-Pratt(KMP)

Algoritma Knuth-Moris-Pratt mirip dengan brute force, yaitu menemukan pola/pattern dalam sebuah teks yang bergerak mulai dari kiri ke kanan. Namun, perbedaannya terletak pada pergeseran yang dilakukan oleh algoritma KMP yang lebih efisien daripada algoritma Brute Force. Jika terjadi ketidakcocokan antara karakter P[j] dan T[i], algoritma KMP

melakukan pergeseran yang lebih tepat dengan menggunakan informasi tentang pola itu sendiri.

Perbedaannya terletak pada pergeseran pola yang dilakukan oleh algoritma KMP yang lebih cerdas daripada algoritma Brute Force. Jika terjadi ketidakcocokan antara karakter Pattern[j] dan Teks[i], pergeseran pola yang dilakukan adalah sebesar panjang prefiks terpanjang dari Pattern[0..j-1] yang juga merupakan sufiks dari Pattern[1..j-1]. Hal ini dilakukan untuk mengurangi jumlah perbandingan karakter yang tidak perlu.

Algoritma KMP dapat lebih efisien dalam mencocokkan pola dengan teks, terutama ketika terdapat variasi karakter yang luas. Algoritma ini mengurangi jumlah perbandingan yang tidak perlu, sehingga kompleksitasnya umumnya lebih baik daripada algoritma Brute Force.



Gambar 2.3.1. String Matching KMP

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Algoritma KMP (Knuth-Morris-Pratt) menggunakan border function, juga dikenal sebagai failure function, untuk menghitung jumlah pergeseran yang harus dilakukan oleh pola saat terjadi ketidakcocokan. Border function b(k) didefinisikan sebagai ukuran terbesar prefiks dari pola P[0..k] yang juga merupakan sufiks dari P[i..k]. Dalam konteks ini, k adalah posisi karakter pada pola sebelum terjadinya ketidakcocokan (ketidakcocokan terjadi pada P[j], sehingga k = j-1).

• Contoh lain: P = abababababca
j = 0123456789

(k = j-1)

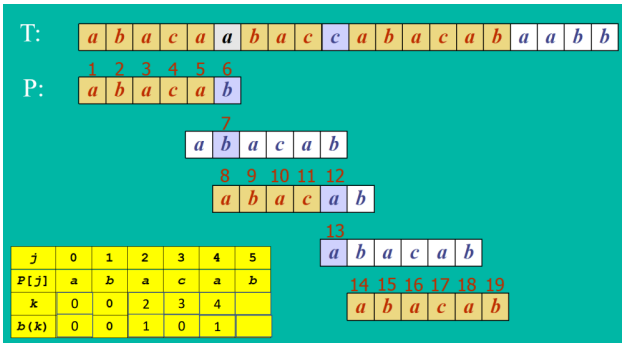
j	0	1	2	3	4	5	6	7	8	9
P [j]	a	b	a	b	a	b	a	b	c	a
k	0	1	2	3	4	5	6	7	8	
b[k]	0	0	1	2	3	4	5	6	0	

Gambar 2.3.2. Border Function

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Algoritma KMP menggunakan border function untuk meningkatkan efisiensi pergeseran yang terjadi dibandingkan dengan algoritma brute force. Jika terjadi ketidakcocokan pada P[j] dan T[i], algoritma KMP akan mengubah P[j] menjadi P[b(k)], di mana k adalah j-1 berdasarkan border function. Setelah itu, pencarian dan pencocokan pola akan dilanjutkan dengan pola yang sudah dipindahkan.

Dengan menggunakan border function, algoritma KMP dapat menghindari perbandingan ulang karakter-karakter yang sebenarnya sudah cocok, sehingga mempercepat proses pencocokan pola. Dengan memanfaatkan informasi tentang hubungan prefiks dan sufiks dalam pola, algoritma KMP mampu melakukan pergeseran yang lebih tepat dan efisien, mengurangi jumlah perbandingan yang tidak perlu dilakukan.



Gambar 2.3.3. Contoh String Matching dengan KMP

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Algoritma KMP memiliki kompleksitas waktu O(m+n), di mana m adalah panjang pola dan n adalah panjang teks dengan O(m) adalah menghitung border function dan O(n) adalah pencarian string. Namun, algoritma KMP kurang efisien saat variasi alfabetnya semakin luas, karena hal ini meningkatkan kemungkinan terjadinya ketidakcocokan pada awal pola P.

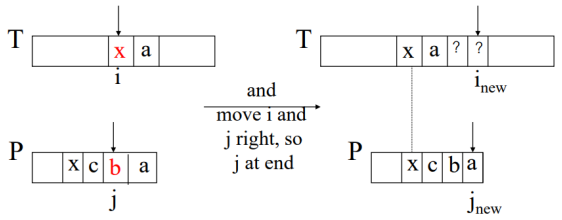
D. Algoritma Boyer Moore

Berbeda dengan algoritma sebelumnya, algoritma Boyer Moore menemukan pola/pattern dalam sebuah teks yang bergerak mulai dari kanan ke kiri. Algoritma Boyer-Moore dalam pencocokan string didasarkan pada dua teknik utama, yaitu teknik "looking-glass" dan teknik "character-jump".

Teknik "looking-glass" digunakan untuk mencari pola P pada teks T dengan melakukan pencarian mundur melalui pola P yang dimulai dari ujung pola tersebut. Dalam teknik ini, pencocokan dimulai dari karakter terakhir pola dan bergerak mundur melalui pola secara bertahap.

Teknik "character-jump" digunakan saat terjadi ketidakcocokan antara karakter P[j] dan T[i], dan T[i] == x (x adalah karakter pada teks). Dalam situasi ini, terdapat tiga kasus yang mungkin terjadi:

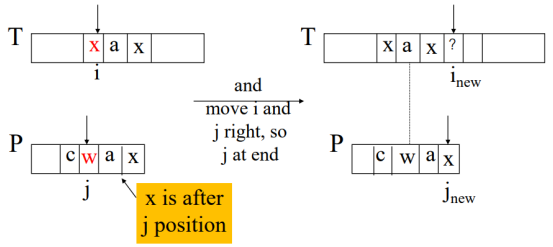
1. Kasus 1: Jika pola P mengandung karakter x, maka pola P akan digeser ke kanan sehingga karakter x yang terakhir muncul pada posisi yang sejajar dengan karakter T[i]. Dengan melakukan pergeseran ini, kita dapat melewatkan beberapa karakter yang tidak perlu untuk dicocokkan.



Gambar 2.3.4. Teknik character-jump untuk kasus 1

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

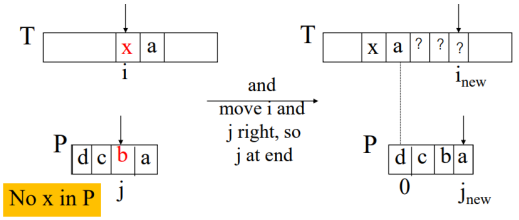
2. Kasus 2: Jika pola P mengandung karakter x tetapi tidak memungkinkan pergeseran ke kanan untuk meluruskan karakter x dengan T[i], maka pola P akan digeser ke kanan sebanyak 1 karakter. Dalam kasus ini, pergeseran satu karakter dilakukan untuk menghindari kesalahan pencocokan yang bisa terjadi.



Gambar 2.3.5. Teknik character-jump untuk kasus 2

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

3. Kasus 3: Jika kedua kasus sebelumnya tidak berlaku, maka pola P akan digeser sehingga karakter P[0] sejajar dengan karakter T[i+1]. Pergeseran dilakukan untuk mencoba posisi baru yang lebih potensial untuk pencocokan selanjutnya.



Gambar 2.3.6. Teknik character-jump untuk kasus 3

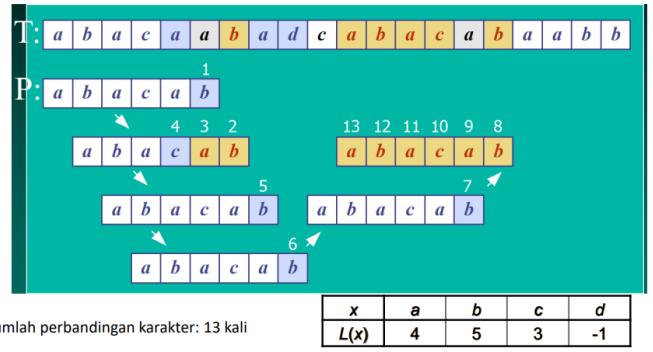
(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Dengan menggunakan teknik "looking-glass" dan "character-jump", algoritma Boyer-Moore dapat meningkatkan efisiensi pencocokan string dengan mengurangi jumlah perbandingan yang harus dilakukan.

Selain itu, diperlukan juga penggunaan fungsi Last Occurrence. Fungsi ini digunakan untuk memetakan setiap karakter dalam alfabet A ke sebuah integer yang merupakan indeks terakhir di mana karakter tersebut muncul dalam pola P. Jika karakter x tidak ada dalam pola P, maka nilai yang diberikan adalah -1.

Fungsi Last Occurrence, yang biasa disimbolkan sebagai L(x), digunakan saat terjadi ketidakcocokan yang sesuai dengan Kasus 1 pada teknik "character-jump". Jika terdapat lebih dari satu karakter x dalam pola P, fungsi ini membantu dalam memilih karakter x yang paling terakhir muncul.

Dengan menggunakan fungsi Last Occurrence, algoritma Boyer-Moore dapat menentukan pergeseran yang optimal untuk mencocokkan karakter pada teks dengan pola. Informasi tentang posisi terakhir munculnya karakter dalam pola membantu dalam menghindari perbandingan yang tidak perlu, sehingga meningkatkan efisiensi algoritma.



Gambar 2.3.6. Contoh Penyelesaian String Matching dengan KMP

(https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf)

Algoritma Boyer-Moore merupakan pilihan yang lebih baik ketika terdapat variasi alfabet yang luas dan tidak monoton. Kompleksitas waktu algoritma ini pada kasus terburuk adalah O(mn+A), di mana m adalah panjang pola, n adalah panjang teks, dan A adalah jumlah alfabet yang digunakan. Perlu dicatat bahwa kompleksitas ini termasuk juga jumlah alfabet dalam perhitungan, karena dalam algoritma Boyer-Moore terdapat penggunaan fungsi Last Occurrence yang terkait dengan alfabet.

Selain itu, algoritma Boyer-Moore juga lebih cepat daripada brute force dalam melakukan pencocokan string. Dengan menggunakan teknik looking-glass, character-jump, dan fungsi Last Occurrence, algoritma ini mampu mengurangi jumlah perbandingan yang tidak perlu dan melakukan pergeseran yang lebih efisien. Hal ini menjadikan algoritma Boyer-Moore sebagai pilihan yang lebih cepat dan efektif

dalam mencari pola pada teks, terutama ketika variasi alfabetnya luas dan tidak monoton.

III. Implementasi Algoritma

Pada makalah ini, penulis memilih untuk menggunakan algoritma string matching Boyer Moore dalam membuat sistem pemberian rekomendasi obat. Pemilihan algoritma Boyer Moore dengan pertimbangan algoritma ini lebih cepat dan efektif dalam mencari pola pada teks, terutama ketika variasi alfabetnya luas dan tidak monoton. Dalam pengimplementasian membuat program penulis memilih untuk menggunakan bahasa pemrograman python.

Berikut merupakan potongan program yang sudah dibuat, yaitu berupa algoritma Boyer Moore.

TABLE I. Table Algoritma Boyer Moore
(sumber: Penulis)

```
def tableLastOccurance(pattern):
    lastOccurance = {}
    for i in range(len(pattern)):
        lastOccurance[pattern[i]] = i
    return lastOccurance

def boyer_moore(text, pattern):
    n = len(text)
    m = len(pattern)
    lastOccurance =
tableLastOccurance(pattern)
    i = m - 1

    while i < n:
        j = m - 1
        while j >= 0 and text[i] ==
pattern[j]:
            i -= 1
            j -= 1

        if j == -1:
            return i + 1

        if text[i] in lastOccurance:
            i += m - min(j, 1 +
lastOccurance[text[i]])
        else:
            i += m

    return -1
```

Pada implementasi algoritma ini, tujuan utamanya adalah mencari kecocokan pola antara sebuah pola (pattern) dan teks (text). Fungsi tersebut akan mengembalikan indeks pertama di mana karakter awal pola ditemukan dalam teks, yaitu ketika T[i] == P[0]. Jika tidak ada kecocokan pola yang ditemukan, fungsi akan mengembalikan nilai -1.

Penulis membuat dua data uji, yaitu data uji untuk rekomendasi obat berdasarkan gejala dan data uji untuk rekomendasi obat berdasarkan kandungan obat. Data uji yang digunakan dalam implementasi ini merupakan sampel data yang diambil dari internet dan jumlahnya terbatas. Data uji obat berdasarkan gejala terdiri dari 12 obat beserta gejala-gejalanya dan data uji obat berdasarkan kandungan obat terdiri dari 5 obat beserta kandungan-kandungannya. Untuk pengembangan lebih lanjut, dapat dibuat sebuah database khusus untuk menyimpan data tersebut. Contoh di bawah hanya merupakan contoh data sampel yang digunakan dalam implementasi algoritma.

TABLE II. Table Obat Beserta Gejala - Gejalanya
(sumber: Penulis)

Alpara	flu, demam, sakit kepala, hidung tersumbat, bersin, batuk.
Antangin	masuk angin, mual, perut kembung, badan meriang, kelelahan.
Antasida	nyeri ulu hati, kembung, mual.
Antimo	mual, muntah, pusing, sakit kepala, sakit perut, diare.
Hypromellose	mata kering, mata perih, mata gatal, mata merah.
Loperamide	diare, flu, tinja encer, tinja berdarah, sakit perut, mual, muntah.
Cataflam	nyeri sendi, nyeri haid, migrain.
Chlorhexidine	sariawan, radang gusi.
Amoxicillin	sakit gigi

Omeprazole	nyeri ulu hati, mual, kembung.
Oralite	diare, muntah, demam, dehidrasi.
Degirol	sakit tenggorokan, radang gusi, radang amandel, sariawan.

TABLE III. TABLE OBAT BESERTA KANDUNGANNYA (SUMBER: PENULIS)

Caviplex	vitamin A, vitamin B, vitamin C, vitamin D, vitamin E, kalsium, magnesium, zat besi, zinc.
Fortiboost D3 1000 IU	vitamin D3, cholecalciferol.
Mefinal	Asam mefenamat
Vitacimin	vitamin C
Natur E	minyak biji gandum, minyak biji bunga matahari, vitamin E.

Dalam pengembangan selanjutnya, data uji dapat diperluas dengan menambahkan lebih banyak nama penyakit dan gejala-gejalanya. Database tersendiri dapat dibuat untuk mengelola dan menyimpan data ini dengan lebih efisien. Hal ini memungkinkan untuk memperluas cakupan dan ketersediaan data dalam aplikasi yang menggunakan algoritma ini untuk sistem pemberian rekomendasi obat.

Secara garis besar, cara kerja program adalah akan mencocokkan gejala/kandungnya yang dimasukkan oleh user dengan data uji. Apabila terdeteksi terdapat kecocokan makan program akan mengeluarkan rekomendasi obat secara terurut berdasarkan obat yang paling sesuai dengan gejala/kandungan yang telah diinput.

Dibawah ini akan dijelaskan cara kerja program yang lebih terperinci.

1. Pembacaan data uji dari file external,

   data uji berasal dari file external txt yang akan dibaca dan diformat sedemikian rupa sehingga dapat dimasukkan ke dalam array of array dengan kolom pertama berisikan nama obat dan kolom kedua berisikan gejala/kandungannya.

2. Setelah program dijalankan, pertama-tama program akan meminta user untuk memilih antara menampilkan rekomendasi obat berdasarkan gejalanya atau menampilkan rekomendasi obat berdasarkan kandungannya.

```
Selamat datang!
 1. Rekomendasi obat berdasarkan masukkan gejala yang Anda rasakan.
 2. Rekomendasi obat berdasarkan kandungan obat.

Masukkan pilihan Anda:
```

Gambar 3.1. Tampilan Program 1

(Sumber: Penulis)

   User dapat mengetik angka 1 untuk mendapatkan rekomendasi obat berdasarkan gejala yang dimasukkan. Dan ketik angka 2 untuk mendapatkan rekomendasi obat berdasarkan kandungan yang dimasukkan

3. Apabila user memasukkan angka 1, program akan meminta untuk memasukkan gejala-gejalanya. Penulisan antar-gejala dipisah dengan koma spasi (,)

```
Selamat datang!
 1. Rekomendasi obat berdasarkan masukkan gejala yang Anda rasakan.
 2. Rekomendasi obat berdasarkan kandungan obat.

Masukkan pilihan Anda: 1
Masukkan gejala:
 Contoh: sakit perut, mual, muntah, demam, pusing
        -
```

Gambar 3.2. Tampilan Program 2

(Sumber: Penulis)

   Lalu, apabila user memasukkan angka 1, program akan meminta untuk memasukkan kandungan obat. Penulisan antar-kandungan dipisah dengan koma spasi (,)

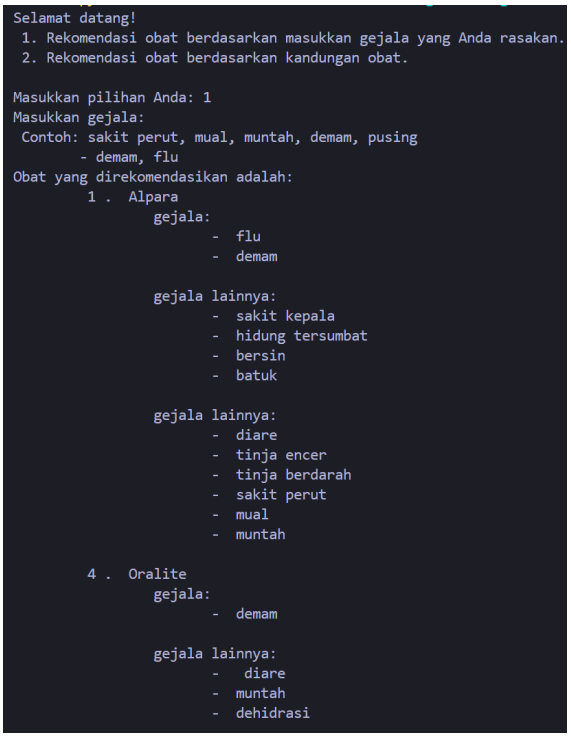
```
Selamat datang!
 1. Rekomendasi obat berdasarkan masukkan gejala yang Anda rasakan.
 2. Rekomendasi obat berdasarkan kandungan obat.

Masukkan pilihan Anda: 2
Masukkan kandungan obat:
 Contoh: vitamin C, kalsium
        -
```

Gambar 3.3. Tampilan Program 3

(Sumber: Penulis)

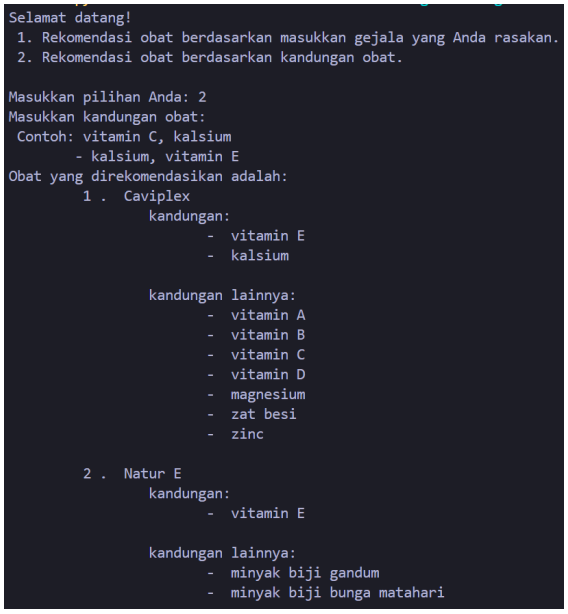
4. Program akan melakukan pencocokan string dari setiap gejala yang dimasukkan user sebagai pola/pattern dengan gejala yang dimiliki setiap masing-masing obat sebagai text. Untuk setiap gejala yang match akan ditampilkan ke layar berupa nama obat beserta gejala yang dimasukkan dan gejala lainnya selain dari gejala yang dimasukkan oleh user sebagai bahan pertimbangan. Obat akan diurutkan berdasarkan banyaknya gejala yang match.



Gambar 3.4. Tampilan Program 4

(Sumber: Penulis)

Sama seperti rekomendasi obat berdasarkan masukkan gejalanya, program akan menampilkan ke layar berupa nama obat beserta kandungan yang dimasukkan dan kandungan lainnya selain dari kandungan yang dimasukkan oleh user sebagai bahan pertimbangan. Obat akan diurutkan berdasarkan banyaknya kandungan yang match.



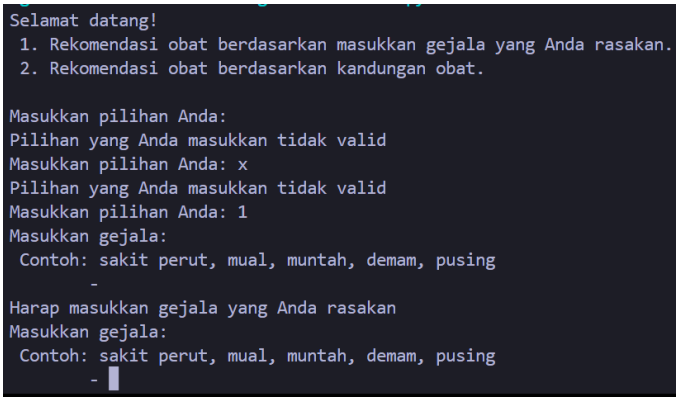
Gambar 3.5. Tampilan Program 5

(Sumber: Penulis)

IV. Hasil Eksperimen dan Pembahasan

Berikut merupakan beberapa Eksperimen yang dilakukan terhadap program dan data uji yang telah dibuat.

1. Eksperimen 1: Masukan string kosong dan tidak valid

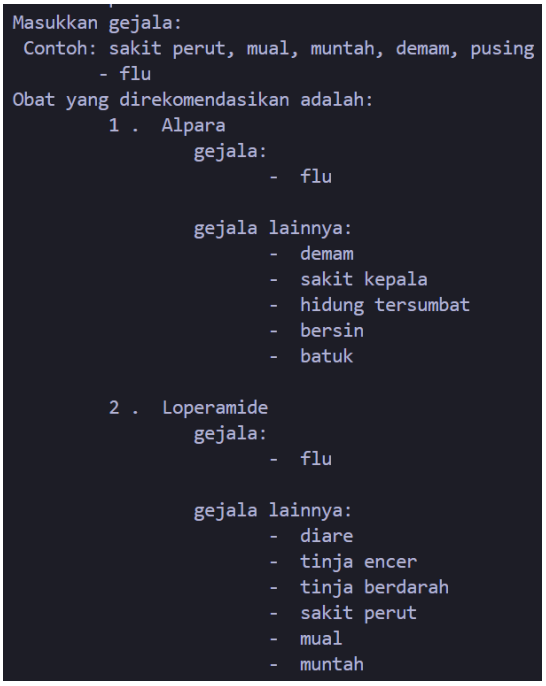


Gambar 4.1. Eksperimen 1

(Sumber: Penulis)

Pada saat dilakukan input string kosong ataupun tidak valid, program akan meminta user untuk memasukkan inputan hingga valid.

2. Eksperimen 2: Masukan gejala “flu”



Gambar 4.2. Eksperimen 2

(Sumber: Penulis)

Dari hasil pengujian, tertampil di layar obat yang sesuai dengan gejala flu adalah obat Alpara dan Loperamide. Program akan menampilkan juga gejala lainnya sebagai bahan pertimbangan.

3. Eksperimen 3: Masukan gejala “flu, demam, bersin, sakit kepala”

```
Masukkan gejala:
 Contoh: sakit perut, mual, muntah, demam, pusing
        - flu, demam, bersin, sakit kepala
Obat yang direkomendasikan adalah:
        1 .  Alpara
                gejala:
                        -  flu
                        -  demam
                        -  sakit kepala
                        -  bersin

                gejala lainnya:
                        -  hidung tersumbat
                        -  batuk

        2 .  Antimo
                gejala:
                        -  sakit kepala
                        -  demam

                gejala lainnya:
                        -  mual
                        -  muntah
                        -  pusing
                        -  sakit perut
                        -  diare

        3 .  Loperamide
                gejala:
                        -  flu

                gejala lainnya:
                        -  diare
                        -  tinja encer
                        -  tinja berdarah
                        -  sakit perut
                        -  mual
                        -  muntah

        4 .  Oralite
                gejala:
                        -  demam

                gejala lainnya:
                        -   diare
                        -  muntah
                        -  dehidrasi
```

Gambar 4.3. Eksperimen 3

(Sumber: Penulis)

Dari hasil pengujian, tertampil di layar obat yang sesuai dengan gejala flu adalah obat Alpara, antimo, Loperamide, dan Oralite. Program akan menampilkan juga gejala lainnya sebagai bahan pertimbangan. Selain itu obat akan terurut berdasarkan banyaknya gejala yang match. Dapat dilihat dari gambar diatas obat alpara berada di urutan nomor 1 karena dari 4 gejala yang dimasukkan terdapat 4 gejala yang sesuai.

4. Eksperimen 4: Masukan “a”

```
Masukkan gejala:
 Contoh: sakit perut, mual, muntah, demam, pusing
        - a
Maaf, belum ada obat yang dapat direkomendasikan berdasarkan gejala yang Anda
masukkan
```

Gambar 4.4. Eksperimen 4

(Sumber: Penulis)

Pada kasus ini sudah terdapat penanganan. Ada kemungkinan gejala memiliki karakter a, sehingga bisa saja gejala tersebut cocok dengan pattern “a”.

5. Eksperimen 5: Tidak ada obat yang sesuai

```
Masukkan gejala:
 Contoh: sakit perut, mual, muntah, demam, pusing
        - luka bakar
Maaf, belum ada obat yang dapat direkomendasikan berdasarkan gejala yang Anda
masukkan
```

Gambar 4.5. Eksperimen 5

(Sumber: Penulis)

Apabila gejala yang dimasukkan tidak dikenali atau tidak terdapat pada database, program akan mengeluarkan pesan error.

6. Eksperimen 6: Masukan kandungan “vitamin A”

```
Selamat datang!
 1. Rekomendasi obat berdasarkan masukkan gejala yang Anda rasakan.
 2. Rekomendasi obat berdasarkan kandungan obat.

Masukkan pilihan Anda: 2
Masukkan kandungan obat:
 Contoh: vitamin C, kalsium
        - vitamin E, kalsium
Obat yang direkomendasikan adalah:
         1 .  Caviplex
                 kandungan:
                         -  vitamin E
                         -  kalsium

                 kandungan lainnya:
                         -  vitamin A
                         -  vitamin B
                         -  vitamin C
                         -  vitamin D
                         -  magnesium
                         -  zat besi
                         -  zinc

          2 .  Natur E
                 kandungan:
                         -  vitamin E

                 kandungan lainnya:
                         -  minyak biji gandum
                         -  minyak biji bunga matahari
```

Gambar 4.6. Eksperimen 6

(Sumber: Penulis)

Sama seperti eksperimen 3, namun pada eksperimen ini, program diuji pada bagian rekomendasi obat berdasarkan kandungan obat.

The template is designed so that author affiliations are not repeated each time for multiple authors of the same affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization). This template was designed for two affiliations.

V. Simpulan dan Saran

Berdasarkan hasil Eksperimen terhadap program yang telah dibuat, dapat disimpulkan bahwa implementasi algoritma string matching Boyer-Moore dapat membantu dalam merekomendasikan obat yang sesuai dengan gejala/kandungannya.Keberhasilan program ini sangat bergantung pada kelengkapan database obat beserta gejala dan kandungannya. Oleh karena itu, disarankan untuk menggunakan database yang lebih konkret dan komprehensif daripada yang sudah ada saat ini. Hal ini akan meningkatkan

akurasi dan keandalan sistem dalam memberikan rekomendasi obat berdasarkan gejala yang diberikan.

Dalam pengembangan lebih lanjut, perlu dilakukan evaluasi dan pengujian yang lebih luas untuk memastikan kinerja dan efektivitas algoritma Boyer-Moore dalam aplikasi pemberian rekomendasi obat dan masih diperlukan penyesuaian dan penanganan kasus-kasus khusus yang mungkin muncul.

Ucapan Terima Kasih

Penulis ingin mengucapkan terima kasih kepada semua pihak yang telah memberikan bantuan dalam menyelesaikan makalah ini. Rasa syukur penulis juga disampaikan kepada Allah SWT yang telah memberikan kesehatan dan kesempatan untuk menulis makalah ini. Penulis juga ingin secara khusus mengucapkan terima kasih kepada keluarga, teman, dan Bapak Ir. Rila Mandala, M.Eng., Ph.D.,, selaku dosen pengampu mata kuliah IF2211 Strategi Algoritma kelas K3, serta seluruh tim dosen strategi algoritma yang telah memberikan kontribusi dalam kegiatan perkuliahan. Terima kasih juga kepada teman-teman penulis. Penulis juga berterima kasih kepada pembuat referensi yang telah berbagi ilmu pengetahuan.

Referensi

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

[2] https://www.alodokter.com/obat-a-z

[3] https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching/

[4] https://www.geeksforgeeks.org/python-program-for-kmp-algorithm-for-pattern-searching-2/

[5] Sonita, Anisya, and Khairunnisyh Khairunnisyh. "IMPLEMENTASI ALGORITMA STING MATCHING PADA RANCANG BANGUN APLIKASI PENDETEKSI OBAT DAN MAKANAN." *Prosiding University Research Colloquium*, January 21, 2019, 124–28.

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, 22 Mei 2023

Jauza Lathifah Annassalafi 13521030