

LAPORAN TUGAS BESAR

Penerapan String Matching dan Regular Expression dalam DNA Pattern Matching

Ditujukan untuk memenuhi salah satu tugas besar mata kuliah IF2211 Strategi Algoritma
pada Semester II Tahun Akademik 2021/2022

Disusun oleh:

Adiyansa Prasetya Wicaksana	13520044
Januar Budi Ghifari	13520132
Rania Dwi Fadhilah	13520142



PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2022

DAFTAR ISI

DAFTAR ISI	i
BAB I DESKRIPSI TUGAS.....	1
BAB II LANDASAN TEORI	5
BAB III ANALISIS PEMECAHAN MASALAH.....	7
BAB IV IMPLEMENTASI DAN PENGUJIAN.....	9
BAB V KESIMPULAN, SARAN, REFLEKSI.....	18
REFERENSI	ii

BAB I

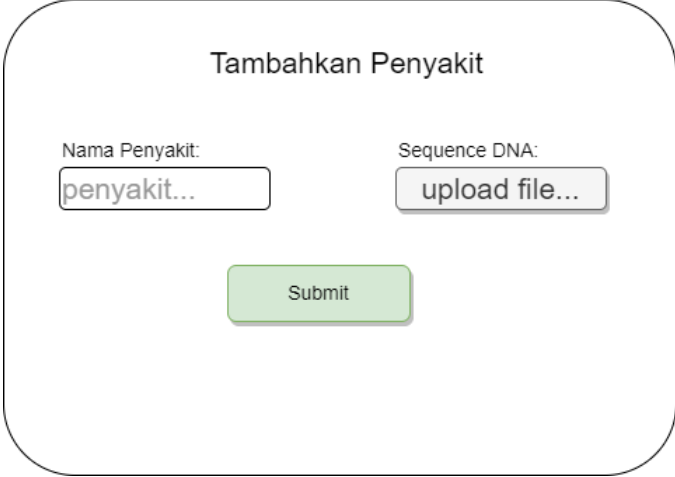
DESKRIPSI TUGAS

Dalam tugas besar ini, anda diminta untuk membangun sebuah aplikasi *DNA Pattern Matching*. Dengan memanfaatkan algoritma *String Matching* dan *Regular Expression* yang telah anda pelajari di kelas IF2211 Strategi Algoritma, anda diharapkan dapat membangun sebuah aplikasi interaktif untuk mendeteksi apakah seorang pasien mempunyai penyakit genetik tertentu. Hasil prediksi tersebut dapat disimpan pada basis data untuk kemudian dapat ditampilkan berdasarkan query pencarian.

Fitur-Fitur

Aplikasi:

1. Aplikasi dapat menerima *input* penyakit baru berupa nama penyakit dan *sequence* DNA-nya (dan dimasukkan ke dalam *database*).
 - a. Implementasi *input sequence* DNA dalam bentuk *file*.
 - b. Dilakukan sanitasi *input* menggunakan **regex** untuk memastikan bahwa masukan merupakan *sequence* DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, dan tidak ada spasi).
 - c. Contoh *input* penyakit:



Gambar 2. Ilustrasi Input Penyakit

2. Aplikasi dapat memprediksi seseorang menderita penyakit tertentu berdasarkan *sequence* DNA-nya.
 - a. Tes DNA dilakukan dengan menerima input nama pengguna, *sequence* DNA pengguna, dan nama penyakit yang diuji. Asumsi *sequence* DNA pengguna > *sequence* DNA penyakit.
 - b. Dilakukan sanitasi *input* menggunakan **regex** untuk memastikan bahwa masukan merupakan *sequence* DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, tidak ada spasi, dll).
 - c. Pencocokan *sequence* DNA dilakukan dengan menggunakan algoritma **string matching**.
 - d. Hasil dari tes DNA berupa tanggal tes, nama pengguna, nama penyakit yang diuji, dan status hasil tes. **Contoh: 1 April 2022 - Mhs IF - HIV - False**

- e. Semua komponen hasil tes ini dapat ditampilkan pada halaman web (*refer* ke poin 3 pada “Fitur-Fitur Aplikasi”) dan disimpan pada sebuah tabel *database*.
- f. Contoh tampilan web:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

<Tanggal> - <pengguna> - <penyakit> - <True/False>

Gambar 3. Ilustrasi Prediksi

- 3. Aplikasi memiliki halaman yang menampilkan urutan hasil prediksi dengan kolom pencarian di dalamnya. Kolom pencarian bekerja sebagai *filter* dalam menampilkan hasil.
 - a. Kolom pencarian dapat menerima masukan dengan struktur: `<tanggal_prediksi><spasi><nama_penyakit>`, contoh “13 April 2022 HIV”. **Format penanggalan dibebaskan**, jika bisa menerima >1 format lebih baik.
 - b. Kolom pencarian dapat menerima masukan hanya tanggal ataupun hanya nama penyakit. Fitur ini diimplementasikan menggunakan **regex**.
 - c. Contoh ilustrasi:
 - i. Masukan tanggal dan nama penyakit

1. 13 April 2022 - Fulan - HIV - True.

2. 13 April 2022 - Kamal - HIV - False

3. 13 April 2022 - Entah - HIV - False

4. 13 April 2022 - Jamal - HIV - True.

5. 13 April 2022 - Yubai - HIV - True.

6. 13 April 2022 - Hika - HIV - False.

Gambar 4. Ilustrasi Interaksi 1

- ii. Masukan hanya tanggal

13 April 2022

1. 13 April 2022 - Fulan - Diabetes - True.
2. 13 April 2022 - Kamal - Sinusitis - False.
3. 13 April 2022 - Enisah - Down Syndrome - False.
4. 13 April 2022 - Jamal - Polio - True.
5. 13 April 2022 - Yubai - TBC - True.
6. 13 April 2022 - Hika - Hepatitis A - False.

Gambar 5. Ilustrasi Interaksi 2

iii. Masukan hanya nama penyakit

HIV

1. 13 April 2022 - Fulan - HIV - True.
2. 14 April 2022 - Kamal - HIV - False.
3. 15 April 2022 - Enisah - HIV - False.
4. 16 April 2022 - Jamal - HIV - True.
5. 17 April 2022 - Yubai - HIV - True.
6. 18 April 2022 - Hika - HIV - False.

Gambar 6. Ilustrasi Interaksi 3

4. **(Bonus)** Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA

- Ketika melakukan tes DNA, terdapat persentase kemiripan DNA dalam hasil tes.
Contoh hasil tes: 1 April 2022 - Mhs IF - HIV - 75% - False
- Perhitungan tingkat kemiripan dapat dilakukan dengan menggunakan Hamming distance, Levenshtein distance, LCS, atau algoritma lainnya (dapat dijelaskan dalam laporan).
- Tingkat kemiripan DNA dengan nilai lebih dari atau sama dengan 80% dikategorikan sebagai **True**. Perlu diperhatikan mengimplementasikan atau tidak mengimplementasikan bonus ini tetap dilakukan pengecekan *string matching* terlebih dahulu.
- Contoh tampilan:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

<Tanggal> - <pengguna> - <penyakit> - <similarity> - <True/False>

Spesifikasi Program:

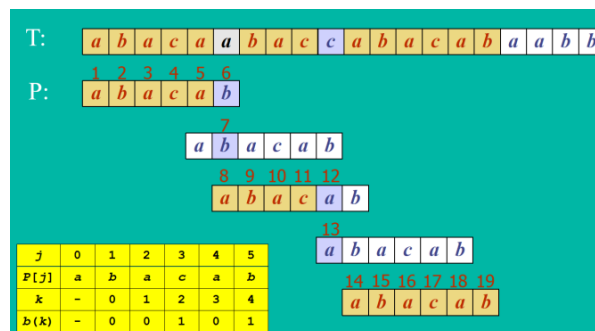
1. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
2. Implementasi Backend **wajib** menggunakan Node.js / Golang, sedangkan Frontend **disarankan** untuk menggunakan React / Next.js / Vue / Angular. Lihat referensi untuk selengkapnya.
3. Penyimpanan data **wajib** menggunakan basis data (MySQL / PostgreSQL / MongoDB).
4. Algoritma pencocokan string (KMP dan Boyer-Moore) **wajib** diimplementasikan pada sisi backend aplikasi.
5. Informasi yang **wajib** disimpan pada basis data:
 - a. Jenis Penyakit:
 - Nama penyakit
 - Rantai DNA penyusun.
 - b. Hasil Prediksi:
 - Tanggal prediksi
 - Nama pasien
 - Penyakit prediksi
 - Status terprediksi.
6. Jika mengerjakan bonus tingkat kemiripan DNA, simpan hasil tingkat kemiripan tersebut pada basis data.

BAB II

LANDASAN TEORI

2.1 Algoritma Knuth Morris Pratt

Algoritma *Knuth Morris Pratt* adalah salah satu algoritma yang kerap kali digunakan untuk mencari sebuah *string*. Algoritma ini dikembangkan pada tahun 1966 oleh Donald E. Knuth, James H. Morris, dan Vaughan R. Pratt.

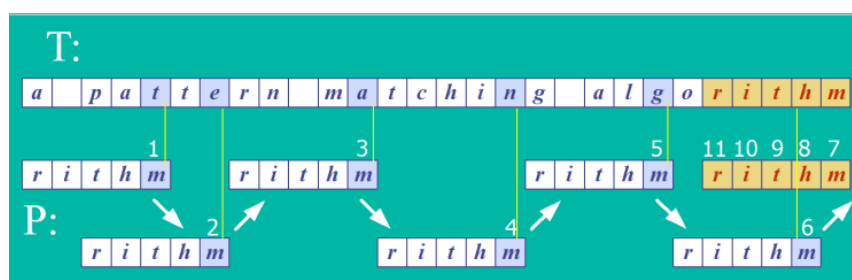


Gambar 2.1 Implementasi Algoritma KMP

Algoritma ini bekerja dengan cara mencari pola pada teks dengan urutan kiri ke kanan seperti *brute-force*, namun dengan menerapkan pergeseran pola yang lebih efisien. Algoritma ini bekerja dengan cara membuat fungsi pinggiran KMP yang didefinisikan sebagai ukuran *prefix* terbesar dari pola $P[0 \dots k]$ dan juga *suffix* dari $P[1 \dots k]$ dan kerap disebut *failure function*. Fungsi ini digunakan untuk menggeser posisi suatu pola agar tidak dilakukan pergeseran satu per satu seperti pada algoritma *brute-force*. Algoritma ini sangat cocok untuk digunakan pada dokumen-dokumen yang besar dengan variasi alfabet yang kecil.

2.2 Algoritma Bayer-Moore

Algoritma *Bayer-Moore* merupakan salah satu algoritma pencarian *string* yang dipublikasikan pada tahun 1977 oleh Robert S. Boyer dan J. Strother Moore. Terdapat dua teknik yang diimplementasikan pada algoritma ini. Teknik pertama adalah *looking-glass technique*, dimana pencarian P pada T dilakukan dengan penelusuran mundur yang dimulai dari akhir teks. Selain itu, diimplementasikan juga teknik *character-jump*, yang akan berlaku apabila ditemukan pola yang tidak sesuai dengan teks ($P[j] \neq T[i]$).

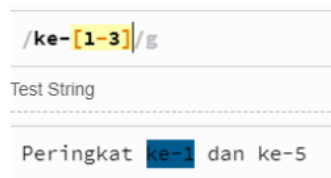


Gambar 2.2 Implementasi Algoritma Bayer-Moore

Algoritma ini dikenal sebagai algoritma yang paling efisien dan paling cocok ketika digunakan pada dokumen dengan variasi alfabet yang besar.

2.3 Regex

Algoritma *Regular Expression* atau yang biasa dikenal regex merupakan teknik pencarian pola dalam *string* dengan menggunakan pola tertentu. Pola tersebut dapat berupa urutan *string* ataupun serangkaian karakter yang mendefinisikan sebuah pola pencarian.



Gambar 2.3 Implementasi Regex

Pencocokan dengan teknik regex dilakukan dengan menyediakan *string literal* yang dicocokkan langsung dengan karakter yang sama atau metakarakter yang merupakan karakter khusus. Metakarakter dalam regex adalah `([{ \ ^ - = $! |]) ? * + . > .`. Ketika menggunakan metakarakter, maka perlu digunakan *backslash* `\`.

2.4 Aplikasi Web

Aplikasi *web* merupakan sebuah aplikasi yang dapat diakses dengan menggunakan *web* melalui suatu jaringan seperti Internet atau intranet. Aplikasi *web* memberikan kemudahan bagi para penggunaanya karena dapat diakses hanya dengan menggunakan internet dan *web browser*, tanpa perlu memasang *software* tambahan pada perangkat. Selain itu, *setup* dan juga pemeliharaan aplikasi web juga cenderung mudah dan efisien. Aplikasi *web* ini biasanya digunakan untuk *online ticketing*, *online payment*, *online shop*, platform sosial media, dan masih banyak lagi. Keunggulan dari aplikasi web ini selain mudah diakses dan dipelihara, juga lebih murah, lebih hemat penyimpanan, dan juga dapat digunakan pada *multi-platform*.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah Penyelesaian

a. Fitur menambah sekuens penyakit baru

Pada fitur ini, input yang diberikan oleh pengguna hanya 2, yaitu nama penyakit dan juga file. Kedua hal tersebut divalidasi oleh sistem agar tidak kosong. Kemudian, ketika file sudah diterima, maka akan divalidasi pula isi dari file tersebut. Validasi yang digunakan adalah pencocokkan dengan algoritma regex. Regex yang digunakan untuk validasi adalah `/^[ACGT]+$/g`. Apabila hasil dari pencocokkan teks dengan regex tersebut kosong, maka input salah dan website akan meminta memasukkan file baru dari pengguna. Namun, apabila hasil pencocokkan menghasilkan sebuah *array* hasil, maka dicek apakah panjang dari elemen *array* ke-0 tersebut sama dengan panjang masukan teks. Apabila sama, maka input file sudah benar.

Kemudian, setelah memastikan bahwa input file sudah benar, data yang masuk akan dikirimkan ke *database* dengan memasukkan id penyakit, nama penyakit, dan juga sekuens penyakit. Pada *database* ini, nama penyakit bersifat unik sehingga duplikasi nama penyakit dengan sekuens berbeda tidak dapat dilakukan.

b. Fitur mencocokkan sekuens DNA dengan sekuens penyakit

Pertama, pengguna memasukkan input pada seluruh kolom yang tersedia. Kemudian, akan dilakukan pengecekan pada masukan file dengan menggunakan algoritma *regex*. File yang dapat diterima hanya file yang berisi huruf AGCT dengan penulisan kapital dan tanpa spasi. Regex yang digunakan untuk validasi adalah `/^[ACGT]+$/g`. Apabila hasil dari pencocokkan teks dengan regex tersebut kosong, maka input salah dan website akan meminta memasukkan file baru dari pengguna. Namun, apabila hasil pencocokkan menghasilkan sebuah *array* hasil, maka dicek apakah panjang dari elemen *array* ke-0 tersebut sama dengan panjang masukan teks. Apabila sama, maka input file sudah benar.

Kemudian, setelah melewati proses tersebut, seluruh data akan masuk dan dikirim ke algo belakang. Pada tahap ini, input yang masuk akan dibandingkan dengan sekuens dari penyakit yang dipilih oleh pengguna. Terdapat dua algoritma yang dapat digunakan untuk membandingkan kedua sekuens, yaitu algoritma KMP dan Bayer-Moore. Apabila hasil dari pencocokan dengan algoritma tersebut adalah sama, maka program akan langsung mengembalikan kecocokan sebesar 100%. Namun, apabila tidak sama, maka kedua sekuens akan dicek kembali menggunakan algoritma Levenshtein yang dibalut dengan algoritma Bruteforce. Pada algoritma ini, sekuens DNA pengguna akan diiterasi sebanyak $m \cdot n$ kali dimana m adalah panjang dari sekuens DNA dan n adalah panjang dari sekuens penyakit. Kemudian, sekuens DNA akan dipotong agar panjangnya sesuai dengan sekuens penyakit dan dibandingkan dengan sekuens penyakit menggunakan algoritma Levenshtein. Algoritma Levenshtein akan menghasilkan jumlah karakter yang sama atau sudah pada tempatnya. Kemudian, di algoritma *bruteforce*, iterasi akan mencari hasil algoritma Levenshtein yang paling besar. Hasil yang paling besar ini yang kemudian akan menjadi nilai kecocokkan sekuens DNA input dengan sekuens penyakit.

c. Fitur melihat riwayat pencarian sekuens

Pertama, pengguna akan memasukkan input berupa pencarian. Format yang diterima adalah sebagai berikut :

1. Tanggal (c/: 13 April 2022)
2. Tanggal – Nama Penyakit (c/: 13 April 2022 Penyakit)

3. Nama Penyakit (c/: Penyakit)

Kemudian, program akan memvalidasi input yang masuk. Pertama-tama, input yang masuk akan di *parse* dan dikumpulkan menjadi array. Apabila panjang dari *array* yang masuk adalah 3, maka akan divalidasi menggunakan regex tanggal, yaitu `/[1-31]/g` kemudian regex bulan, yaitu nama-nama bulan, dan juga regex tahun, yaitu `/[0-9][0-9][0-9][0-9]/g`. Apabila berhasil lolos di semuanya, maka tipe *query* yang dicari pada *database* adalah tipe 1, yaitu pencarian tanggal saja. Apabila panjang dari *array* yang masuk lebih dari 4, maka 3 array pertama akan divalidasi menggunakan regex pencarian yang sama seperti tipe 1, dan array selanjutnya akan divalidasi apakah isinya hanya huruf saja (tidak boleh ada angka). Apabila lolos pada tahap ini, maka tipe *query* yang dicari pada *database* adalah tipe 2, yaitu pencarian menggunakan tanggal dan nama penyakit. Apabila tidak lolos pada kedua tahap itu, maka akan divalidasi apakah isi dari array tersebut adalah huruf semua. Apabila benar, maka tipe *query* yang dicari pada *database* adalah tipe 3, yaitu nama penyakit saja. Apabila tidak lolos pada semua tahap, maka akan mengembalikan tipe 0 yang berarti format salah dan pencarian tidak dapat dilakukan.

3.2 Fitur Fungsional dan Arsitektur Aplikasi Web yang Dibangun

Terdapat beberapa fitur fungsional yang dibangun pada aplikasi *web* ini, antara lain :

1. Menambahkan sekuens penyakit baru
2. Mencocokkan sekuens DNA *input* dengan sekuens penyakit yang sudah tersedia
3. Melihat riwayat pencarian sekuens

Adapun, *frontend* yang digunakan untuk membangun aplikasi *web* ini adalah React. Untuk *Backend* sendiri menggunakan ExpressJS yang dibantu oleh basis data mySQL.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

a. Struktur Data

Data disimpan pada database yang bernama 'tehtotolsourcecode'. Terdapat dua tabel yang digunakan untuk menyimpan data, yaitu tabel Penyakit dan juga tabel TesDNA. Tabel Penyakit terdiri atas id penyakit, nama penyakit, dan juga sekuens dari penyakit. Tabel ini digunakan sebagai acuan pencocokan *string* pada fitur 2 (Bab 3.2). Kemudian, terdapat tabel TesDNA yang menyimpan id pengguna, nama pengguna, penyakit, status, nilai kemiripan, tanggal pencarian, dan juga sekuens input. Tabel ini digunakan untuk menyimpan seluruh hasil pencarian yang kemudian dapat dicari kembali pada halaman riwayat.

b. Fungsi

- kmpMatch() : Algoritma string matching KMP
- boyerMooreMatch() : Algoritma string matching boyer-moore
- levenshtein() : Algoritma untuk menentukan kemiripan dengan menggunakan algoritma levenshtein
- brute_levenshtein() : Algoritma untuk menentukan kemiripan dengan menggunakan algoritma levenshtein secara brute force.
- monthToInt() : Merubah string month menjadi integer
- parseString() : Pencocokan regex untuk search history
- connection() : Fungsi untuk memulai koneksi ke database
- Penyakit.create() : Membuat penyakit pada database
- Penyakit.getAll() : Mengambil semua penyakit dari database
- Penyakit.findByName() : Mencari penyakit berdasarkan nama
- TesDNA.create() : Membuat TesDNA pada database
- TesDNA.getAll() : Mengambil semua TesDNA dari database
- TesDNA.getLatest() : Mengambil TesDNA terbaru
- TesDNA.findByName() : Mengambil TesDNA berdasarkan penyakit
- TesDNA.findByTanggal() : Mengambil TesDNA berdasarkan tanggal
- TesDNA.findByNameAndTanggal() : Mengambil TesDNA berdasarkan penyakit dan tanggal.
- createPenyakit() : Membuat penyakit dari request yang dikirim
- getPenyakit() : Mengambil semua penyakit dan mengirim response
- getPenyakitByName() : Mengambil penyakit berdasarkan nama dan mengirim response
- createTesDNAKMP() : Membuat TesDNA berdasarkan algoritma KMP dari request yang dikirim
- createTesDNABM() : Membuat TesDNA berdasarkan algoritma Boyer-Moore dari request yang dikirim
- getAllTesDNA() : Mengambil semua TesDNA dan mengirim response
- getLatestTesDNA() : Mengambil TesDNA terbaru dan mengirim response

- `searchTesDNA()` : Mencari TesDNA berdasarkan penyakit ataupun tanggal dan mengirim response.

c. Prosedur

- `appListen()` : Menerima request dari Frontend dan mengatur routing.

4.2 Penjelasan Tata Cara Penggunaan Program

Instalasi & Persiapan Program

1. Pada *command prompt*, ketik

```
https://github.com/raniadf/Tubes3_13520044.git
```

2. Buka folder hasil *clone*

3. Buat database dengan nama *tehbotolsourcecode*, kemudian ganti isi file `config.db.js` pada folder `./src/config` dengan mengubah passwordnya sesuai password dari akun database yang baru saja dibuat.

4. Install *requirements* dengan cara

a. Ketik `npm install` pada terminal folder `./src/client`

b. Ketik `npm install` pada terminal folder `./src`

Interface & Fitur Program

1. Buka folder hasil clone

2. Pada *command prompt*, ketik

```
cd src
```

3. Akses aplikasi Web dengan mengetik

```
npm run dev
```

4. Akses fitur-fitur pada aplikasi *web*

a. Menambahkan Sekuens Penyakit Baru



Disease Name

DNA Sequence

Submit

❗ DNA sequence must be only A, C, G, T. ✕

1. Akses page “Add Disease” dengan cara klik navigation menu button di pojok kiri atas web, lalu pilih “Add Disease”.
2. Masukkan nama penyakit baru yang ingin ditambahkan dalam input box “Disease Name”
3. Upload file yang berisikan sekuens dna dalam bentuk text file berekstensi “.txt” ke dalam input box “DNA Sequence”
4. Apabila isi file .txt tidak sesuai dengan yang diharapkan, maka akan muncul error message dan file tidak akan diterima
5. Klik tombol submit untuk meregister penyakit baru ke dalam database

b. Mencocokkan sekuens DNA *input* dengan sekuens penyakit yang sudah tersedia



Name

Disease

DNA Sequence

Technique

☐ KMP
☐ Bayer-Moore

Submit

Information

DNA Sequence Input

Consists of only ACGT Characters (all uppercase), with no whitespaces and length of sequence is less than length of disease sequence

✓ ACGTACGT

✗ ACGT ACGT

✗ acgtACgt

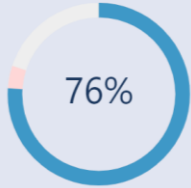
✗ AUGTACGT

Analysis Result

NAME Bonita Bonata

DATE 2022-04-29

DISEASE Kemiskinan

DETECTION RESULT


76%

0 - 79 % : NOT DETECTED (FALSE)
80 - 100 % : DETECTED (TRUE)

✓ Success! Your DNA test has been created. ✕

1. Akses page “DNA Test” dengan cara klik navigation menu button di pojok kiri atas web, lalu pilih “DNA Test”.
2. Masukkan nama pengguna yang ingin dicek dalam input box “Name”
3. Pilih penyakit yang ingin dicocokkan dengan memilih salah satu opsi pada box “Disease”
4. Upload file yang berisikan sekuens DNA dalam bentuk text file berekstensi “.txt” ke dalam input box “DNA Sequence”. Pastikan bahwa isi file hanya mengandung huruf AGCT (kapital dan tanpa spasi)
5. Apabila isi file .txt tidak sesuai dengan yang diharapkan, maka akan muncul error message dan file tidak akan diterima
6. Pilih teknik pencocokkan (KMP/Bayer-Moore)
7. Klik tombol submit untuk melihat hasil (Analysis Result)

c. Melihat riwayat pencarian sekuens



History



Jauari	AIDS ▾
Rania	AIDS ▾
Halo	AIDS ▾
Halo	AIDS ▾
Testing	AIDS ▾
Testing	AIDS ▾
Tes	AIDS ▾

1. Akses page “History” dengan cara klik navigation menu button di pojok kiri atas web, lalu pilih “History”.
2. Masukkan pencarian pada kolom input dengan format tanggal/ nama/ tanggal dan nama (C/: '23 April 2022' atau '23 April 2022 Penyakit' atau 'Penyakit')

4.3 Hasil & Analisis Pengujian

Uji Fitur 1 (Menambahkan sekuens penyakit baru)

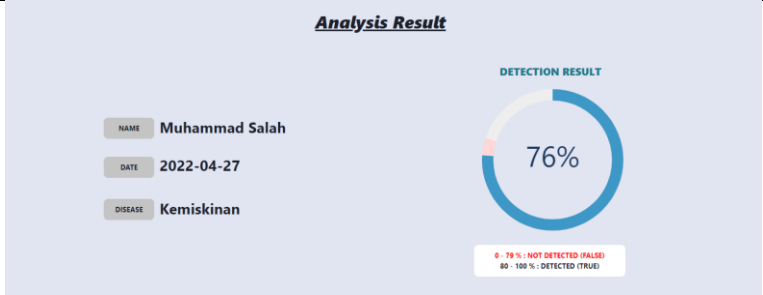
1	<u>Nama Penyakit</u>	Hemofilia
	<u>Sekuens Penyakit</u>	ACGTAGCTAGCTACGGTTAGCTA
	<u>Hasil Uji</u>	
	<u>Hasil pada database</u>	11 Hemofilia ACGTAGCTAGCTACGGTTAGCTA
2	<u>Nama Penyakit</u>	Kemiskinan
	<u>Sekuens Penyakit</u>	GGGGTCATTTC AACAGATATTGCTGATGGTTTAGGCGTACA ATGCCCTGAAGAATAATTAAGAAAAAAGCACCCCTCGTCG CCTAGAATTACCTACCGCG
	<u>Hasil Uji</u>	
	<u>Hasil pada database</u>	12 Kemiskinan GGGGTCATTTC AACAGATATTGCTGATGGTTTAGGCGTACAATGCCCTG AAGAATAATTAAGAAAAAAGCACCCCTCGTCGCTAGAATTACCTACCGCG

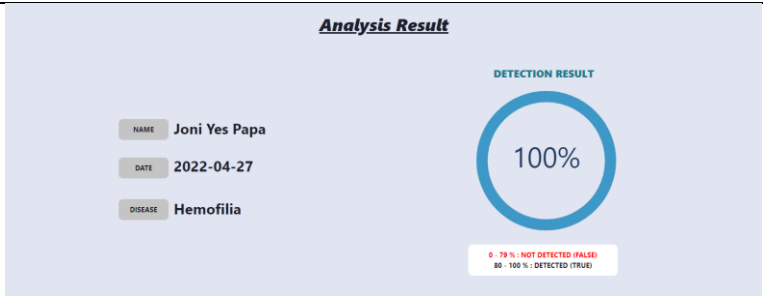
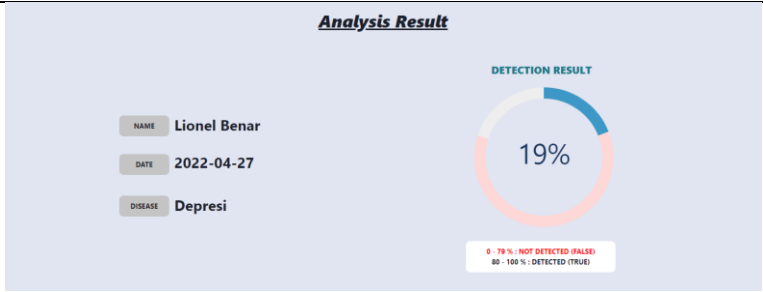
3	<u>Nama Penyakit</u>	Depresi
	<u>Sekuens Penyakit</u>	GCGCTCACCGACTGTTCCCAAAGTAACTCATCGTTCCGT CAAGGCCTGACTTACTTCCCAGGCCCTTCCATGCGCGGACC ATACCGTCCTAGTTCTTCGGTTATGTTTCCGATGTAGGAGT GAGCCTACCTCCGTTTTCGTCTTGTACCAATGAAAAAGCT ATGCACTTTGTACAGGGTGCCATCGGGTTTCTGAAC
	<u>Hasil Uji</u>	
	<u>Hasil pada database</u>	<div style="background-color: black; color: white; padding: 5px; font-family: monospace; font-size: 0.8em;"> 14 Depresi GCGCTCACCGACTGTTCCCAAAGTAACTCATCGTTCCGTCAAGGCCT GACTTACTTCCCAGGCCCTTCCATGCGCGGACCATACCGTCCTAGTTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCC TACCTCCGTTTTCGTCTTGTACCAATGAAAAAGCTATGCACTTTGTACAGGGTGCCATCGGGTTTCTGAAC </div>

Analisis :

Berdasarkan ketiga percobaan di atas, maka dapat diambil kesimpulan bahwasanya input sekuens penyakit berhasil masuk ke *database* dengan baik dan komponen yang diterima hanya sekuens yang tersusun oleh huruf-huruf AGCT (kapital dan tanpa spasi)

Uji Fitur 2 (Mencocokkan sekuens DNA *input* dengan sekuens penyakit yang sudah tersedia)


1	<u>Nama</u>	Muhammad Salah
	<u>Prediksi Penyakit</u>	Kemiskinan
	<u>Sekuens</u>	GGGGTCATTTCAACAGATATTGCTGATGGTTTAGG CGTACAATGCCCAGAGATGATGATGAGTAGATGAG ATAGCGATCGATAGTAGCGATCGATCGTGAAGAAT AATTAAGAAAAAAGCACCCCTCGTCGCCTAGAATT ACCTACCGCG
	<u>Teknik</u>	KMP
	<u>Hasil Uji</u>	<div style="text-align: center;"> Analysis Result  </div>
	<p style="text-align: center;"><u>Analisis</u></p> <p>Berdasarkan sekuens dari penyakit kemiskinan, yaitu GGGGTCATTTCAACAGATATTGCTGATGGTTTAGGCGTACAATGCCCTGAAGAATAATTTAAGAAAAAGCACCCTCGTCGCCTAGAATTCTACCGCG maka dapat disimpulkan bahwa sekuens penyakit memiliki kesamaan sebesar 76% dengan sekuens DNA dari Muhammad Salah.</p>	
2	<u>Nama</u>	Joni Yes Papa
	<u>Prediksi Penyakit</u>	Hemofilia
	<u>Sekuens</u>	ACGTAGCTAGCTACGGTTAGCTA
	<u>Teknik</u>	Bayer-Moore

3	Hasil Uji	
	Analisis	<p>Berdasarkan sekuens dari penyakit hemofilia, yaitu ACGTAGCTAGCTACGGTTA GCTA maka dapat disimpulkan bahwa kecocokan yang dimilikinya dengan sekuens DNA Joni Yes Papa adalah 100%.</p>
	Nama Prediksi Penyakit Sekuens Teknik Hasil Uji	Lionel Benar DEPRESI AA AA AA AA AA AA AA AA AA AA KMP 
	Analisis	<p>Berdasarkan sekuens dari penyakit depresi, yaitu GCGCTCACCGACTGTTCCCAAACTGTA ACTCATCGTTCCGTCAAGGCCTGACTTAACTTCCCGGCCCTTTCCATGCGCGGACC ATACCGTCCTAGTTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCCTACCTCCGT TTGCGTCTTGTTACCAATGAAAAAGCTATGCACTTTGTACAGGGTGCCATCG GGTTTCTGAAC maka dapat disimpulkan bahwa kesamaan sekuens DNA Lionel Benar dengan penyakit depresi hanyalah 19%.</p>

Uji Fitur 3 (Melihat riwayat pencarian sekuens)

1	Pencarian	29 April 2022
---	-----------	---------------

<p>Hasil Uji</p>	<div> <div>29 April 2022</div> <div>Q</div> <div> <div>Rania</div> <div>Depresi ^</div> <div> <div>Tanggal: 2022-04-29</div> <div>Penyakit: Depresi</div> <div>Kemiripan: 0%</div> <div>NEGATIVE</div> </div> </div> <div> <div>DEPRESYONG</div> <div>Patah Hati v</div> </div> <div> <div>Tessss</div> <div>Insomnia v</div> </div> <div> <div>Testing Bouz</div> <div>Hemofilia v</div> </div> <div> <div>Testing ae</div> <div>Kemiskinan v</div> </div> <div> <div>Testing</div> <div>Insomnia v</div> </div> <div> <div>Bora Nur Syah</div> <div>Hemofilia v</div> </div> <div> <div>Bonita Bonata</div> <div>Kemiskinan v</div> </div> </div>
<p>Hasil pada database</p>	<pre> MariaDB [tehbotoisourcecode]> select * from tesdna where DATE(Tanggal) = "2022-04-29" ; +-----+-----+-----+-----+-----+-----+ IDPengguna NamaPengguna Penyakit Status Kemiripan Tanggal Sekuens +-----+-----+-----+-----+-----+-----+ 29 Rania Depresi 0 0 2022-04-29 08:48:37 ACGCTTCAGATGTGACCATATACTTAGGCTGGATCTGTCCTCCGTGAATTTAACCTCACCACTACGAGATATGAGTAAGC CAAAAAGCACGTGGTG 30 DEPRESYONG Patah Hati 1 100 2022-04-29 08:50:22 TCTCAGATAGTGGGGATCCCGGGAAGGGCCTATATTTGCGGTCCAACCTTAGGCGTAAACCTCGATGCTACCTACTCAGACCC ACCCGCGCGGGGTAAATATGGCACTCATCCAGCTGGTTCTTGCGTTCTACGCAGCCACATGTTCT 31 Tessss Insomnia 0 0 2022-04-29 09:37:55 CGATGCT 32 Testing Bouz Hemofilia 1 100 2022-04-29 15:32:15 ACGTAGCTAGCTACGGTTAGCTA 33 Testing ae Kemiskinan 0 76 2022-04-29 15:37:11 GGGGTCATTTCAACAGATATTGCTGATGGTTTAGGCGTACAATGCCAGAGATGATGATGAGTAGATGAGATAGCGATCGATA GTAGCGATCGATCGTGAAGAATAATTAAGAAAAAGCACCCCTCGTCGCCTAGAATTACCTACCGCG 34 Testing Insomnia 0 0 2022-04-29 15:57:37 ACGTAGCTAGCTACGGTTAGCTA 35 Bora Nur Syah Hemofilia 1 100 2022-04-29 16:00:27 ACGTAGCTAGCTACGGTTAGCTA 36 Bonita Bonata Kemiskinan 0 76 2022-04-29 16:00:45 GGGGTCATTTCAACAGATATTGCTGATGGTTTAGGCGTACAATGCCAGAGATGATGATGAGTAGATGAGATAGCGATCGATA GTAGCGATCGATCGTGAAGAATAATTAAGAAAAAGCACCCCTCGTCGCCTAGAATTACCTACCGCG </pre>
<p>Pencarian</p>	<p>29 April 2022 Kemiskinan</p>
<p>Hasil Uji</p>	<div> <div>29 April 2022 Kemiskinan</div> <div>Q</div> <div> <div>Testing ae</div> <div>Kemiskinan ^</div> <div> <div>Tanggal: 2022-04-29</div> <div>Penyakit: Kemiskinan</div> <div>Kemiripan: 76%</div> <div>NEGATIVE</div> </div> </div> <div> <div>Bonita Bonata</div> <div>Kemiskinan v</div> </div> </div>

	<p><u>Hasil pada database</u></p> <pre> MariaDB [tehbotoisourcecode]> select * from tesdna where DATE(Tanggal) = "2022-04-29" and penyakit like "Kemiskinan" -> ; +-----+-----+-----+-----+-----+-----+ IDPengguna NamaPengguna Penyakit Status Kemiripan Tanggal Sekuens +-----+-----+-----+-----+-----+-----+ 33 Testing ae Kemiskinan 0 76 2022-04-29 15:37:11 GGGGTCATTTCAACAGATATTGCTGATGGTTTAGGCGTACAATGCCAGAGATGATGAGTAGAGTAGAGTAGCGATCGATA GTAGCGATCGATCGTGAAGAATAATTAAGAAAAAGCACCCCTCGTCGCCTAGAATTACCTACCGCG 36 Bonita Bonata Kemiskinan 0 76 2022-04-29 16:00:45 GGGGTCATTTCAACAGATATTGCTGATGGTTTAGGCGTACAATGCCAGAGATGATGAGTAGAGTAGAGTAGCGATCGATA GTAGCGATCGATCGTGAAGAATAATTAAGAAAAAGCACCCCTCGTCGCCTAGAATTACCTACCGCG +-----+-----+-----+-----+-----+-----+ 2 rows in set (0.013 sec) </pre>
<p><u>Pencarian</u></p> <p><u>Hasil Uji</u></p>	<p>Depresi</p> 
<p>3</p> <p><u>Hasil pada database</u></p>	<pre> +-----+-----+-----+-----+-----+-----+ IDPengguna NamaPengguna Penyakit Status Kemiripan Tanggal Sekuens +-----+-----+-----+-----+-----+-----+ 23 Lionel Benar Depresi 1 100 2022-04-28 00:11:52 GCGCTCACCGAC TGTTCACCAACTGTAACATCATCGTTCCGTCAAGGCCTGACTTACTTCCCGGCCCTTCCATGCGCGGACCATACCGTCCTAGTTCTTCGGTTATGT TTCCGATGTAGGAGTGAGCCTACCTCCGTTTGCCTCTTGTACCAATGAAAAAGCTATGCACCTTTGTACAGGGTGCCATCGGGTTTCTGAAC 24 Lionel Benar Depresi 0 19 2022-04-28 00:12:15 AAAAAAAAAAAAA AAA AAA AAA 25 Lionel Benar Depresi 0 19 2022-04-28 00:25:53 AAAAAAAAAAAAA AAA AAA AAA AAA +-----+-----+-----+-----+-----+-----+ </pre>

Analisis :

Berdasarkan ketiga percobaan di atas, maka dapat diambil kesimpulan bahwasanya hasil uji yang dilakukan pada *database* sudah sesuai dengan hasil pada *database*.

BAB V

KESIMPULAN, SARAN, DAN REFLEKSI

I. Kesimpulan

Pada tugas besar mata kuliah IF2211 Strategi Algoritma yang ke-3 ini, telah berhasil diimplementasikan hasil pembelajaran algoritma KMP dan Bayer-Moore dalam bentuk *website* pendeteksi DNA Penyakit. Pendekatan menggunakan algoritma KMP dan Bayer-Moore sendiri diimplementasikan pada perbandingan sekuens dari DNA yang diperoleh pada *input* dengan sekuens penyakit yang sudah tersimpan di dalam basis data.

Dengan demikian, kelompok dapat menyimpulkan bahwa dengan mengerjakan Tugas Besar III IF2211 Strategi Algoritma Semester 2 Tahun 2021/2022 ini, dapat diketahui bahwa mencocokkan sekuens DNA dengan sekuens penyakit dapat dilakukan dengan mengimplementasikan algoritma KMP dan Bayer-Moore.

II. Saran

Tugas besar IF2211 Strategi Algoritma Semester 2 Tahun 2021/2022 menjadi salah satu proses pembelajaran bagi kelompok dalam menerapkan ilmu-ilmu yang dipelajari pada perkuliahan ataupun dengan melakukan eksplorasi materi secara mandiri. Berikut ini adalah saran dari kelompok untuk pihak-pihak yang ingin melakukan atau mengerjakan hal serupa.

- Dibutuhkan *brainstorming* yang cukup dalam mengerjakan tugas ini. Dengan melakukan *brainstorming*, strategi yang digunakan akan menjadi lebih efisien.
- Melakukan riset yang mendalam terkait web-app development dengan cara membaca dokumentasi dan banyak berlatih. Dengan menerapkan hal ini, pekerjaan akan selesai dengan lebih cepat.
- Lakukan analisis setiap ada pergantian masif terhadap algoritma. Hal ini dapat membantu para *programmer* untuk mencari letak kesalahan dan hal-hal lainnya yang sekiranya masih bisa diimplementasikan.
- Dalam mengerjakan suatu tugas secara berkelompok, penting untuk memiliki strategi serta distribusi tugas yang baik dan efisien. Cara penulisan kode dan kemampuan menulis komentar menjadi hal yang sangat penting dalam mengerjakan kode pemrograman secara berkelompok. Dengan adanya komentar pada kode, anggota lain pada kelompok dapat memahami cara kerja suatu kode dengan lebih cepat. Kemampuan tersebut juga didukung dengan adanya version control system (VCS) yang baik untuk digunakan oleh programmer dalam membuat sebuah kode pemrograman secara bersamaan. Kelompok sangat menyarankan penggunaan 'Github' untuk digunakan sebagai version control system (VCS) dalam pengerjaan tugas besar, maupun pada pembuatan program yang lainnya.

III. Refleksi

Dengan mengerjakan tugas besar kali ini, kelompok menjadi lebih paham terkait penggunaan algoritma KMP dan juga Bayer-Moore. Selain itu, kelompok juga menjadi lebih familiar dengan teknik-teknik pembuatan *website*.

LINK PENTING

Github : https://github.com/raniadf/Tubes3_13520044

Video : <https://bit.ly/tehbotolsourcecode>

DAFTAR PUSTAKA

Tim Mata Kuliah IF2211 Strategi Algoritma. (2022). *Spesifikasi Tugas Besar*,
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf> , diakses 20 April 2022.

Munir, Rinaldi. (2021). *Pencocokan String (String/Pattern Matching)*.
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf> ,
diakses 20 April 2022.

Munir, Rinaldi. (2021). *String Matching dengan Regular Expressions*.
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf> , diakses 20 April 2022.

Towards Data Science. (2020) *Pattern Search with the Knuth-Morris-Pratt (KMP) algorithm*.
<https://towardsdatascience.com/pattern-search-with-the-knuth-morris-pratt-kmp-algorithm-8562407dba5b> , diakses 20 April 2022.

Geeks for Geeks. (2021). *Boyer-Moore Algorithm for Pattern Searching*.
<https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching/> , diakses 20 April 2022.

Chua Hock-Chuan. (2018). *Regular Expressions (Regex)*.
<https://www3.ntu.edu.sg/home/ehchua/programming/howto/Regexe.html> , diakses 20 April 2022.