

Laporan Tugas Besar 3
IF2211 Strategi Algoritma
Penerapan String Matching dan Regular Expression dalam DNA
Pattern Matching



Disusun Oleh:
TubesDNAku

| | |
|------------------|----------|
| Farhan Hafiz | 13520027 |
| Ilham Pratama | 13520041 |
| Haidar Ihzaulhaq | 13520150 |

Program Studi Teknik Informatika
Institut Teknologi Bandung
2022

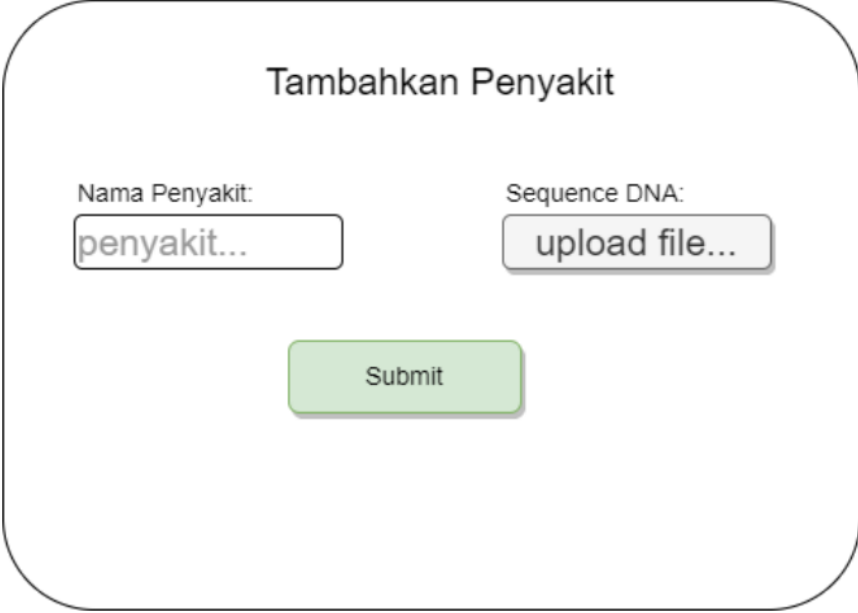
Bab 1

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:



The image shows a web form titled "Tambahkan Penyakit". It contains two input fields: "Nama Penyakit:" with a placeholder text "penyakit..." and "Sequence DNA:" with a placeholder text "upload file...". Below these fields is a green "Submit" button.

Gambar 1. 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 2. 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: , 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:
 - Masukan tanggal dan nama penyakit

13 April 2022 HIV

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 3. Ilustrasi interaksi 1

- Masukan hanya tanggal

13 April 2022

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

2. 13 April 2022 - Kamal - Sinusitis - False.

3. 13 April 2022 - Entah - 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 4. Ilustrasi interaksi 2

- Masukkan hanya nama penyakit

The image shows a web interface for HIV testing results. At the top, there is a button labeled "HIV". Below this button, there are six text boxes, each containing a test result. The results are as follows:

| No. | Date | Name | Result |
|-----|---------------|-------|--------------|
| 1. | 13 April 2022 | Fulan | HIV - True. |
| 2. | 14 April 2022 | Kamal | HIV - False. |
| 3. | 15 April 2022 | Entah | HIV - False. |
| 4. | 16 April 2022 | Jamal | HIV - True. |
| 5. | 17 April 2022 | Yubai | HIV - True. |
| 6. | 18 April 2022 | Hika | HIV - False. |

Gambar 5. Ilustrasi interaksi 3

4. (Bonus) Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA
 - a. Ketika melakukan tes DNA, terdapat persentase kemiripan DNA dalam hasil tes. Contoh hasil tes: 1 April 2022 - Mhs IF - HIV - 75% - False
 - b. Perhitungan tingkat kemiripan dapat dilakukan dengan menggunakan Hamming distance, Levenshtein distance, LCS, atau algoritma lainnya (dapat dijelaskan dalam laporan).
 - c. 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.
 - d. Contoh tampilan:

Tes DNA

Nama Pengguna:
<pengguna>

Sequence DNA:
upload file...

Prediksi Penyakit:
<penyakit>

Submit

Hasil Tes

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

Gambar 6. Ilustrasi prediksi dengan persen kemiripan

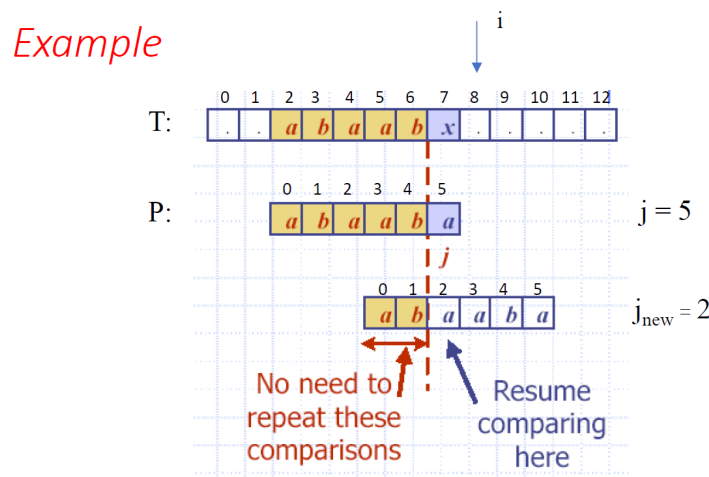
Bab 2

Landasan Teori

2.1 Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt atau lebih sering disingkat KMP adalah salah satu metode melakukan string/pattern matching. Algoritma ini berperilaku mirip dengan algoritma paling sederhana atau brute force dalam hal memindai teks dan pattern dari kiri ke kanan (awal ke akhir), namun dinilai lebih efektif karena ketika menemukan ketidakcocokkan (mismatch) dapat menggeser lebih dari 1 space.

Sebagai contoh, ketika pattern $P[j]$ mengalami mismatch pada teks $T[j]$, pattern dapat “digeser” sejauh prefiks $P[0..j-1]$ terbesar yang juga merupakan sufiks dari $P[1..j-1]$.



Gambar 2.1. Contoh pergeseran algoritma KMP

Oleh karena itu, salah satu elemen algoritma ini adalah Border Function ($b[k]$), yaitu sebuah fungsi yang terlebih dahulu memproses pattern untuk menemukan kecocokan antara prefiks-prefiks P dengan P sendiri. Sebelum dilakukan string matching, akan dibuat tabel yang berisi j sebagai posisi indeks pada P , $P[j]$ sebagai alfabet P pada posisi j , k sebagai $j-1$, dan $b[k]$ sebagai hasil dari Border Function pada k .

| | | | | | | |
|--------|---|---|---|---|---|---|
| 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 |

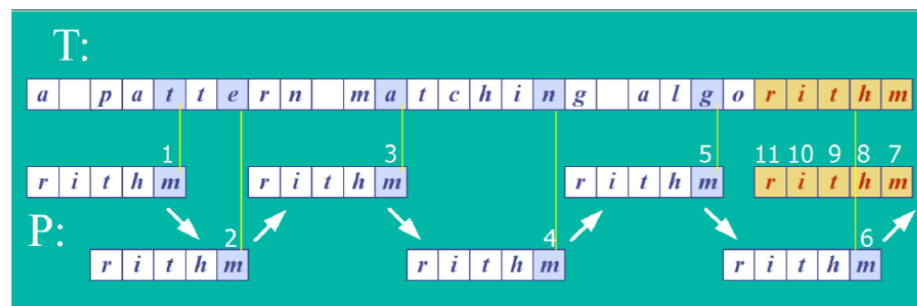
$b(k)$ is the size of the largest border.

Gambar 2.2. Contoh border function

Kompleksitas algoritma ini adalah $O(m + n)$, dengan $O(m)$ kompleksitas border function, dan $O(n)$ kompleksitas pattern matching untuk kasus terburuk. Adapun, kelebihan KMP adalah tidak pernah bergerak mundur selama melakukan pattern matching, sehingga cocok digunakan untuk memproses file-file berukuran besar. Sedangkan, kelemahannya adalah untuk ukuran alfabet yang besar, akan seringkali terjadi mismatch sehingga kompleksitasnya menyerupai brute force (hanya menggeser 1 space).

2.2 Algoritma Boyer-Moore

Algoritma Boyer-Moore merupakan salah satu algoritma yang digunakan dalam proses *string matching* atau pencarian suatu *pattern* pada sebuah text. Algoritma Boyer-Moore ini berbeda dengan algoritma brute force dan Knuth-Morris-Pratt. Algoritma dapat dikatakan berbeda dengan brute force dan Knuth-Morris-Pratt karena algoritma ini memindai pattern dari kanan ke kiri, sedangkan untuk teksnya tetap memindai dari kiri ke kanan.



Gambar 2.3. Contoh pemindaian pattern dengan algoritma BM

Algoritma ini memanfaatkan dua elemen, yaitu *Character Jump Technique* dan *Last Occurrence Function*. *Character Jump Technique* menentukan seberapa banyak P akan bergeser tergantung posisi terakhir huruf tersebut dilihat dari P. Dengan ketentuan sebagai berikut.

- Jika karakter mismatch terakhir ditemukan di kiri lokasi mismatch, geser sampai karakter tersebut ter-align / lurus dengan karakter mismatch.
- Jika karakter mismatch terakhir ditemukan di kanan lokasi mismatch, geser seluruh pattern sebanyak 1 space (seperti brute force).
- Jika tidak ada karakter mismatch pada keseluruhan P, geser seluruh P sampai indeks 0 melewati karakter mismatch.

Untuk melakukan character jump technique, dibutuhkan *last occurrence function* yang melakukan preprocessing pattern dan membuat array kemunculan tiap huruf T pada P. Tabel yang dihasilkan berisi x atau semua karakter unik pada P, dan $L(x)$ yaitu fungsi yang menghasilkan kemunculan terakhir x pada P. Jika x tidak terdapat pada P, $L(x)$ akan menghasilkan -1.

| | | | | |
|-------------|----------|----------|----------|----------|
| <i>x</i> | <i>a</i> | <i>b</i> | <i>c</i> | <i>d</i> |
| <i>L(x)</i> | 4 | 5 | 3 | -1 |

Gambar 2.4. Contoh Last Occurrence Function

2.3 Regular Expression

Regular Expression atau biasa disebut regex merupakan sebuah string atau *pattern* yang mendefinisikan sebuah pola pencarian sehingga dapat digunakan untuk melakukan *string matching*, *locate* (pencarian), dan manipulasi teks. Regex memiliki notasi umum yang berlaku hampir sama untuk di semua bahasa pemrograman. Berikut adalah notasi umum regex yang digunakan pada pengerjaan tugas besar ini:

Character Classes

| Character | Keterangan | Contoh Penggunaan | Contoh String |
|-----------|---|-------------------|---|
| [...] | Salah satu karakter ada pada bracket | [AEIOU] | One uppercase vowel |
| [...] | Salah satu karakter ada pada bracket | T[ao]p | Tap or Top |
| - | Indikator range | [a-z] | One lowercase letter |
| [x-y] | Salah satu karakter berada di range x sampai y | [A-Z]+ | GREAT |
| [...] | Salah satu karakter ada pada bracket | [AB1-5w-z] | One of either: A,B,1,2,3,4,5,w,x,y,z |
| [x-y] | Salah satu karakter berada di range x sampai y | [~~]+ | Characters in the printable section of the ASCII table . |
| [^x] | Karakter yang bukan x | [^a-z]{3} | A1! |
| [^x-y] | Karakter yang tidak berada di antara range x sampai y | [^ ~~]+ | Characters that are not in the printable section of the ASCII table . |
| [\d\D] | Karakter yang merupakan digit ataupun non digit. | [\d\D]+ | Any characters, including new lines, which the regular dot doesn't match |

Sumber: <https://www.rexegg.com/regex-quickstart.html>

Quantifier

| Quantifier | Keterangan | Contoh Penggunaan | Contoh String |
|------------|--------------------------|-------------------|----------------|
| + | Satu atau lebih karakter | Version \w-\w+ | Version A-b1_1 |
| {3} | Tepat 3 karakter | \D{3} | ABC |
| {2,4} | 2 sampai 4 karakter | \d{2,4} | 156 |
| {3,} | Tiga karakter atau lebih | \w{3,} | regex_tutorial |

| | | | |
|---|-------------------------------|----------|--------|
| * | Nol karakter atau lebih | A*B*C* | AAACC |
| ? | Sekali atau tidak sama sekali | plurals? | plural |

Sumber: <https://www.rexegg.com/regex-quickstart.html>

Anchor dan Boundary

| Anchor | Keterangan | Contoh Penggunaan | Contoh String |
|--------|---|-------------------|-----------------------------|
| ^ | Awal dari suatu string (terletak diluar bracket) | ^abc.* | abc (line start) |
| \$ | Akhir dari suatu string | .*? the end\$ | this is the end |
| \A | Awal dari suatu string | \Aabc[\d\D]* | abc (string... ...start) |
| \z | Akhir dari string | the end\z | this is...\n...the end |
| \Z | Akhir dari string sebelum final line break | the end\Z | this is...\n...the end\n |
| \G | Awal dari suatu string atau akhir dari match sebelumnya | | |
| \b | Batas kata | Bob.*\bcat\b | Bob ate the cat |
| \B | Batas untuk bukan kata | c.*\Bcat\B.* | copycats |

Sumber: <https://www.rexegg.com/regex-quickstart.html>

2.4 Longest Common Subsequence (LCS)

Longest Common Subsequence adalah sebuah permasalahan pencarian subsequence terpanjang yang terdapat diantara dua string/pattern yang dibandingkan. Sebagai contoh, diberikan sebuah string “abcdgh” dan string “aedfhr”, maka LCS dari kedua string tersebut adalah 3 (“adh”). Permasalahan ini salah satunya dapat diselesaikan dengan cara dynamic programming. Solusi yang digunakan yaitu, pertama anggap dua string yang dibandingkan adalah *pattern* dan *text* dengan m adalah pattern.length dan n adalah text.length. Fungsi LCS memiliki 4 parameter, yaitu pattern, text, m, n dengan m dan n akan selalu berubah pada tiap rekursinya. Base case dari rekursi ini adalah ketika m atau n sudah mencapai 0 (artinya string *pattern* atau *text* sudah selesai diperiksa semua) dan akan mengembalikan nilai 0. Untuk rekursi yang dilakukan, jika karakter ke-m pada *pattern* sama dengan karakter ke-n pada *text* maka akan dilakukan pengecekan LCS pada karakter selanjutnya dan menambahkan 1 pada hasil. Jika tidak sama, maka akan dilakukan perbandingan nilai yang lebih maksimal antara LCS dengan m-1 dan n atau LCS dengan m dan n-1. Lebih lengkapnya, model rekursi dari LCS adalah sebagai berikut:

| | | | |
|-----------|---|---------------------------------------|-----------------------------|
| Base case | : | 0, | jika m = 0 atau n = 0 |
| Rekursi | : | * 1 + LCS(pattern, text, m-1, n-1), | jika pattern[m] = text[n] |
| | : | * 0 + max(LCS(pattern, text, m-1, n), | LCS(pattern, text, m, n-1)) |

2.5 Penjelasan Aplikasi Web (TubesDNAku)

TubesDNAku merupakan suatu aplikasi yang berbasis website, yang memanfaatkan javascript sebagai bahasa pemrograman utamanya dan react js sebagai penyusun GUI-nya. Untuk media penyimpanan datanya aplikasi ini menggunakan mysql.

TubesDNAku merupakan suatu aplikasi yang dapat digunakan untuk mendiagnosis seseorang menderita suatu penyakit dengan menggunakan dengan memasukkan uraian rantai DNA dari orang yang akan diperiksa. TubesDNAku mempunyai 3 fitur utama yaitu :

1. *Test* , fitur ini merupakan suatu fitur yang digunakan untuk mengecek penyakit seseorang.
2. *Search* , fitur ini merupakan fitur yang digunakan untuk mencari data yang ada dalam database.
3. *Upload* , fitur merupakan fitur yang digunakan untuk menambahkan jenis penyakit baru kedalam database.

Bab 3

Analisis Pemecahan Masalah

3.1 Langkah - Langkah Penyelesaian Masalah Setiap Fitur

- **Fitur Tes**

- 1) Aplikasi menerima masukan berupa rangkaian DNA pengguna dan jenis penyakit yang ingin didiagnosa
- 2) Jika penyakit yang didiagnosa tidak ada didalam database, maka aplikasi akan mengeluarkan pesan kesalahan
- 3) Jika penyakit yang didiagnosa ada di dalam database, maka aplikasi akan mulai mencocok kan DNA pengguna dengan DNA jenis penyakit yang akan didiagnosa. Pencocokan ini menggunakan algoritma Knuth-Morris-Pratt, Boyer-Moore, dan LCS.
- 4) Jika kecocokan lebih atau sama dengan 80% maka sistem akan mengeluarkan hasil bahwa pengguna menderita penyakit yang didiagnosa
- 5) Jika kecocokan kurang dari 80% maka aplikasi akan mengeluarkan hasil bahwa pengguna tidak menderita penyakit yang didiagnosa
- 6) Setelah pencocokan dilakukan, aplikasi akan menyimpan hasil pemeriksaan ke dalam database

- **Fitur Search**

- 1) Aplikasi menerima inputan berupa tanggal dan/atau nama penyakit
- 2) Aplikasi akan mencari ke dalam database apakah ada data yang tanggal dan/atau penyakit yang sesuai dengan inputan
- 3) Jika ada maka aplikasi akan menampilkan semua data yang sesuai dengan inputan
- 4) Jika tidak ada maka aplikasi akan menampilkan pesan kesalahan

- **Fitur Upload**

- 1) Aplikasi menerima inputan berupa nama penyakit dan rangkaian DNA sequence dari penyakit tersebut
- 2) Aplikasi akan menambahkan inputan tersebut kedalam database

3.2 Fitur Fungsional dan Arsitektur Aplikasi Web yang Dibangun

- **Tes Page**

- **TextField Input**

TextField Input digunakan untuk menginputkan data pengguna seperti nama yang diuji DNA-nya dan nama penyakit yang akan didiagnosa, serta file DNA yang diuji.

- **Submit button**

Submit button digunakan untuk menjalankan proses pengujian setelah semua textfield input terisi.

- **Search Page**

- **TextField Input**

- TextField Input digunakan untuk menginputkan pencarian yang diinginkan, seperti tanggal, nama penyakit, atau tanggal dan nama penyakit.

- **Submit button**

- Submit button digunakan untuk menjalankan proses pencarian setelah semua textfield input terisi.

- **Upload Page**

- **TextField Input**

- TextField Input digunakan untuk menginputkan data penyakit yang ingin ditambahkan, berupa nama penyakit dan file DNA penyakit.

- **Submit button**

- Submit button digunakan untuk menjalankan proses upload penyakit setelah semua textfield input yang terisi.

- **About Page**

- Berisi informasi anggota kelompok.

Bab 4

Implementasi dan Pengujian

4.1 Spesifikasi Teknis Program

a. Struktur Data

Struktur Data yang digunakan secara utama dalam pengambilan dan pengiriman data ke basis data adalah JSON yaitu Javascript Object Notation. Contoh JSON yang digunakan pada program yaitu hasil_prediksi dan jenis_penyakit. Untuk hasil_prediksi sendiri seperti di bawah ini :

```
Let hasil_prediksi = {  
    tanggal_prediksi: "2022-04-25"  
    nama_pasien: "Hafiz"  
    penyakit_prediksi: "Fever"  
    status_prediksi : "Yes"  
    tingkat_kemiripan: 90;  
};
```

Sedangkan untuk jenis_penyakit, seperti dibawah ini :

```
Let jenis_penyakit = {  
    Nama_penyakit = "Fever"  
    Dna_penyusun = "AGCTAGCT"  
};
```

b. Fungsi dan prosedur

Fungsi dan prosedur untuk algoritma KMP matching pada kmpMatching.js

| Nama Fungsi dan Prosedur | Keterangan |
|----------------------------|---|
| computeFail(pattern) | Fungsi yang digunakan untuk membuat border function dari kmpMatching |
| kmpMatching(pattern, teks) | Fungsi yang digunakan untuk menentukan apakah pattern bagian dari test, jika pattern bagian dari text maka fungsi akan me-return indeks dimana pattern terletak pada teks |

Fungsi dan prosedur untuk algoritma BM Matching pada bmmatching.js

| Nama Fungsi dan Prosedur | Keterangan |
|--------------------------|--|
| buildLast(pattern) | Fungsi yang digunakan untuk menentukan |

| | |
|---------------------------|---|
| | indeks terakhir suatu karakter yang ada di text. |
| min(a,b) | Fungsi untuk mereturn nilai minimum antara a dan b |
| bmmatching(pattern, text) | Fungsi yang digunakan untuk menentukan apakah pattern merupakan substring dari sebuah text dengan menggunakan algoritma Boyer-Moore. Apabila ditemukan pattern pada text, pada akan mengembalikan nilai index pertama pattern pada text. Apabila tidak ada, maka akan mengembalikan nilai -1. |

Fungsi dan prosedur untuk algoritma LCS pada lcs.js

| Nama Fungsi dan Prosedur | Keterangan |
|---------------------------------|---|
| lcs(pattern, text, m, n) | Fungsi yang digunakan untuk menentukan longest common subsequence antara pattern dan text. Nantinya fungsi ini digunakan untuk menghitung kemiripan yang ada antara pattern dengan text |
| max(a,b) | Fungsi untuk mengembalikan nilai terbesar antara a dan b |

Fungsi dan prosedur lainnya yang digunakan dalam pembuatan aplikasi

| Nama Fungsi dan Prosedur | Keterangan |
|---------------------------------|--|
| dnaRegex(text) | Fungsi ini digunakan untuk mencocokkan string DNA yang diinputkan oleh user. Apabila string DNA tidak sesuai (mengandung karakter selain AGCT), maka fungsi akan mengembalikan nilai false, jika sesuai akan mengembalikan nilai true. |
| searchRegex(text) | Fungsi ini digunakan untuk mencocokkan string pencarian yang diinputkan oleh user. Apabila string pencarian tidak sesuai (tidak sesuai format, antara nama-penyakit, tanggal, atau tanggal nama-penyakit), maka fungsi akan |

| | |
|--|--|
| | mengembalikan nilai 4 (yang diolah menjadi false di program utama), jika sesuai akan mengembalikan nilai 1/2/3 (sesuai ketentuan pencarian). |
|--|--|

4.2 Tata Cara Penggunaan Program

- 1) Pastikan pada perangkat tempat program digunakan telah terinstall javascript dan npm
- 2) Pastikan pada perangkat telah terinstall beberapa dependencies yang diperlukan untuk menjalankan frontend dan backend antara lain react-router-dom, axios, express, body-parser, mysql, nodemon, dan cors. Apabila belum, dapat melakukan instalasi terlebih dahulu dengan npm install <T> dengan T adalah dependency yang diinginkan
- 3) Untuk menghubungkan antara Frontend dan Backend, masuk ke directory utama program lalu masuk ke directory Backend
- 4) Pada directory Backend buka terminal lalu ketikkan **npm run devStart** untuk menghubungkan ke Frontend
- 5) Untuk menggunakan program pastikan anda telah masuk kedalam directory utama program. Setelah masuk ke directory utama program lalu masuk ke directory Frontend
- 6) Pada directory Frontend buka terminal lalu ketikkan **npm start**
- 7) Tunggu hingga program membuka browser dan program bisa digunakan

4.3 Hasil Pengujian

- 1) Menambahkan Penyakit Baru
 - Tampilan Awal

The screenshot shows a web application interface for adding a new disease. The header includes the logo 'TubesDNAku' and navigation links 'Test', 'Search', 'Upload', and 'About'. The main heading is 'Add New Disease'. Below this, there is a form with the following elements:

- A text input field labeled 'New Disease'.
- A section labeled 'DNA Sequence' containing a 'Choose File' button and a text field showing the file path 'C:\Users\user\Downloads\DNA Sequence.txt'.
- A blue 'Submit' button.

- Gagal menambahkan penyakit baru

TubesDNAku [Test](#) [Search](#) [Upload](#) [About](#)

Add New Disease

New Disease
Itchy

DNA Sequence
Choose File

Submit

Upload Failed. DNA Sequence can only filled by A G T C

- Berhasil menambahkan penyakit baru

TubesDNAku [Test](#) [Search](#) [Upload](#) [About](#)

Add New Disease

New Disease
Itchy

DNA Sequence
Choose File Red-ItchyToucan.txt

Submit

Upload new disease succesful

- Bukti penyakit baru telah masuk ke dalam database

The screenshot shows a database query interface with a query editor at the top and a result grid below. The query is `SELECT * FROM dnadb.jenis_penyakit;`. The result grid displays the following data:

| nama_penyakit | dna_penyusun |
|---------------|----------------------|
| Cough | ATGTCGTATCGT |
| Fever | AGCTAGCT |
| Flu | AAGGCCTTAA |
| Headhace | ATTCCGGCCTTA |
| Itchy | ATTTTATTTTGTTTT |
| Stomachache | AGAGAGAG |
| Toothache | AAAAAAAAAAAAAAAAAAAA |
| NULL | NULL |

2) Melakukan tes untuk prediksi penyakit terhadap seseorang

- Gagal melakukan tes karena penyakit prediksi tidak ada di database

The screenshot shows the TubesDNAku web application interface. The form includes the following fields and elements:

- Username:** Input field containing "Haftz".
- Disease Prediction:** Input field containing "HIV".
- DNA Sequence:** Input field with a "Choose File" button and a file path "c:\dna\test\..._dna\haftz.txt".
- Submit:** A blue button to execute the test.
- Result:** A message stating "There is no disease found".

- Berhasil melakukan tes dengan prediksi salah

TubesDNAku [Test](#) [Search](#) [Upload](#) [About](#)

DNA Testing

Username
Hafiz

Disease Prediction
Cough

DNA Sequence
Choose File No file chosen

Submit

2022-4-29 - Hafiz - Cough - 75% - No

- Berhasil melakukan tes dengan prediksi benar

TubesDNAku [Test](#) [Search](#) [Upload](#) [About](#)

DNA Testing

Username
Hafiz

Disease Prediction
Cough

DNA Sequence
Choose File No file chosen

Submit

2022-4-29 - Hafiz - Cough - 92% - Yes

- Bukti hasil tes baru telah masuk ke dalam database

jenis_penyakit tes jenis_penyakit hasil_prediksi jenis_penyakit hasil_prediksi x

Limit to 1000 rows

1 • SELECT * FROM dnadb.hasil_prediksi;

Result Grid

| | id | tanggal_prediksi | nama_pasien | penyakit_prediksi | status_prediksi | tingkat_kemiripan |
|--|------|------------------|-------------|-------------------|-----------------|-------------------|
| | 16 | 2022-04-29 | Haidar | Stomachache | No | 50 |
| | 17 | 2022-04-29 | farhan | Stomachache | No | 50 |
| | 18 | 2022-04-29 | Ilham | Stomachache | No | 50 |
| | 19 | 2022-04-29 | Ilham | Stomachache | No | 50 |
| | 20 | 2022-04-29 | Siapa ya | Stomachache | No | 100 |
| | 21 | 2022-04-29 | Monica | Cough | Yes | 92 |
| | 22 | 2022-04-29 | Halo | Cough | Yes | 92 |
| | 23 | 2022-04-29 | Hambin | Cough | No | 75 |
| | 24 | 2022-04-29 | Budi | Cough | No | 75 |
| | 25 | 2022-04-29 | Sinta | Cough | Yes | 100 |
| | 26 | 2022-04-29 | Hafiz | Cough | No | 75 |
| | 27 | 2022-04-29 | Hafiz | Cough | Yes | 92 |
| | NULL | NULL | NULL | NULL | NULL | NULL |

Form Editor
Field Types
Query Stats

- 3) Menampilkan hasil prediksi
- Tidak ditemukan hasil prediksi

TubesDNAku [Test](#) [Search](#) [Upload](#) [About](#)

Search

Input

HIV

Submit

There is no data match

- Hasil prediksi berdasarkan tanggal

Search

Input

2022-04-24T17:00:00.000Z - Farhan - Fever - 50% - No
2022-04-24T17:00:00.000Z - Anonim - Flu - 90% - Yes

- Hasil prediksi berdasarkan penyakit prediksi

Search

Input

2022-04-24T17:00:00.000Z - Farhan - Fever - 50% - No
2022-04-25T17:00:00.000Z - Hafiz - Fever - 100% - Yes
2022-04-28T17:00:00.000Z - Budi - Fever - 25% - No

- Hasil prediksi berdasarkan tanggal dan penyakit prediksi

Search

Input

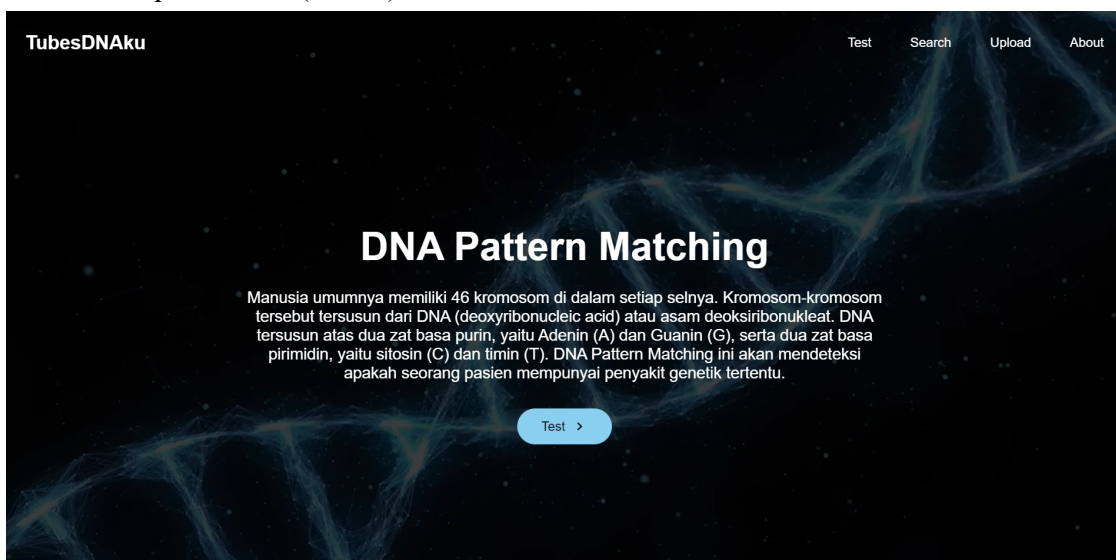
2022-04-25 Flu

Submit

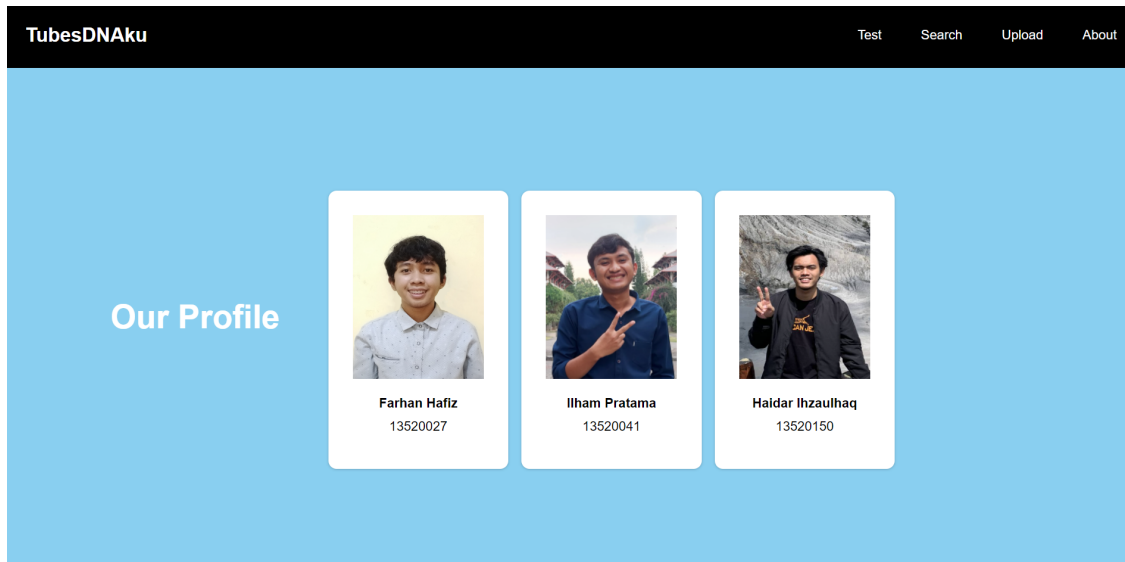
2022-04-24T17:00:00.000Z - Anonim - Flu - 90% - Yes

4) Tampilan Tambahan

- Tampilan Awal (Home)



- Tampilan About (Profile)



4.4 Analisis Hasil Pengujian

1) Menambahkan penyakit baru

Program memiliki beberapa kasus uji dengan hasil yang berbeda. Pada bagian 4.3 ditampilkan gagal menambahkan penyakit baru dan berhasil menambahkan penyakit baru. Gagal menambahkan penyakit baru terjadi ketika mencoba memasukkan penyakit baru yaitu 'Itchy' dengan file 'seed-ItchyFalse.txt' yang berisi 'AABBBCCDDD'. Hal ini gagal karena isi dari file tersebut menghasilkan DNA yang tidak valid karena mempunyai huruf selain A,G,C,T. Sedangkan berhasil menambahkan penyakit baru terjadi ketika mencoba memasukkan penyakit dengan file 'seed-ItchyTrue.txt' yang berisi 'ATTTTATTTTGTTTT'. Hal ini berhasil karena isi dari file telah valid dan penyakit baru beserta dna penyusunnya akan dimasukkan ke dalam database pada tabel jenis_penyakit.

2) Melakukan tes untuk prediksi penyakit terhadap seseorang

Program memiliki beberapa kasus uji dengan hasil yang berbeda. Pada bagian 4.3 ditampilkan tidak menemukan penyakit prediksi, berhasil melakukan tes dengan hasil salah dan berhasil melakukan tes dengan hasil benar. Program menampilkan tidak menemukan penyakit prediksi apabila nama penyakit prediksi yang diinginkan tidak ada dalam database jenis_penyakit. Program menampilkan berhasil melakukan tes dengan hasil salah apabila DNA sequence yang diupload memiliki kemiripan kurang dari 80% dari dna penyusun penyakit. Program menampilkan berhasil melakukan tes dengan hasil benar apabila DNA sequence yang diupload memiliki kemiripan lebih dari sama dengan 80% dari dna penyusun penyakit. Kedua hasil tersebut baik salah ataupun benar akan dimasukkan ke dalam database pada tabel hasil_prediksi.

3) Menampilkan hasil prediksi

Program memiliki beberapa kasus uji dengan hasil yang berbeda. Pada bagian 4.3 ditampilkan tidak menemukan hasil prediksi, serta berhasil menampilkan hasil prediksi berdasarkan masukkan tanggal, nama penyakit, atau pun keduanya. Jika masukkan tidak valid (tidak sesuai antara ketiga kriteria) maka akan menampilkan bahwa input tidak valid. Program menampilkan tidak menemukan hasil prediksi karena pada database tidak ada hasil yang sesuai dengan input pengguna. Sebaliknya, program akan menampilkan seluruh data yang sesuai dengan database sesuai dengan input pengguna.

Bab 5

Kesimpulan dan Saran

5.1 Kesimpulan

Algoritma Knuth-Morris-Pratt dan Boyer-Moore dapat digunakan untuk mencari suatu pattern tertentu pada suatu teks. Proses pencarian pattern ini dilakukan dengan memanfaatkan algoritma yang mirip dengan algoritma brute force, namun lebih efisien dengan kompleksitas waktu $O(m+n)$. Dengan menggunakan kedua algoritma ini bisa dihasilkan berbagai macam aplikasi yang sangat berguna bagi kehidupan. Contohnya seperti aplikasi yang dibuat pada laporan ini.

5.2 Saran

Saran yang dapat kami berikan adalah kami berharap kedepannya tugas seperti ini tetap dipertahankan karena tugas ini menuntut mahasiswa untuk bereksplorasi terkait dengan bidang studinya sehingga dapat membuat mahasiswa semakin berkembang secara pribadi maupun sosial.

5.3 Link Video Demo dan Repository

- Video demo : <https://youtu.be/LB1UPRudPKE>
- Repository : https://github.com/Fnhafiz/Tubes3_13520027.git

DAFTAR PUSTAKA

- <https://www.rexegg.com/regex-quickstart.html>. Quick-Start: Regex Cheat Sheet. Diakses pada 27 April 2022
- “Pencocokan String (String/Pattern Matching)”. Informatika.stei.itb.ac.id. Diakses pada 27 April 2022
- “String Matching dengan Regular Expression”. Informatika.stei.itb.ac.id. Diakses pada 27 April 2022
- <https://www.geeksforgeeks.org/longest-common-subsequence-dp-4/>. Longest Common Subsequence. Diakses pada 29 April 2022