Algoritma Pencocokan *String* dan *Regular Expression* pada Roboguru

Claudia - 13520076
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13520076@std.stei.itb.ac.id

***Abstract*—Dengan perkembangan teknologi yang sangat maju, tanpa disadari teknologi telah membantu dan memudahkan berbagai aspek dalam kehidupan manusia, salah satunya adalah dalam aspek pendidikan. Saat ini siswa tidak perlu kebingungan jika ingin mempelajari suatu materi atau menanyakan soal yang tidak dapat mereka kerjakan karena sudah ada banyak teknologi yang dapat membantu, salah satunya adalah Roboguru. Siswa dapat menanyakan soal yang ingin diselesaikan dan Roboguru akan melakukan pencarian soal dengan menggunakan string matching dan regular expression dan memberikan solusi atau pendekatan terkait soal tersebut.**

Keywords—String Matching; Regular Expression; Roboguru

I. PENDAHULUAN

Perkembangan teknologi telah mempermudah kehidupan manusia dalam mengakses segala sesuatu. Dalam bidang pendidikan sendiri, perkembangan teknologi membuat proses belajar-mengajar tetap dapat terlaksana dengan baik walaupun di tengah pandemi COVID-19. Pembelajaran yang dulunya harus dilakukan secara tatap muka bisa dilakukan secara *online*. Situs-situs pembelajaran *online* juga sudah berkembang dengan pesat, memberikan kemudahan dalam mengakses pembelajaran di mana pun dan kapan pun.

Saat ini, ada banyak sekali inovasi dalam bidang pendidikan. Banyak *start up* yang berlomba-lomba untuk memberikan fitur-fitur baru yang dapat membantu siswa dalam proses pembelajaran. Siswa bisa dengan mudah mencari video pembelajaran terkait materi yang ingin dipelajari, mendapatkan guru privat, mengakses latihan-latihan soal, dan lain-lain. Bahkan, siswa tidak perlu kebingungan lagi untuk menanyakan soal-soal yang tidak dapat mereka selesaikan karena dengan bantuan teknologi siswa dapat mendapatkan jawaban atas soal tersebut atau setidaknya mendapatkan gambaran untuk mengerjakan soal yang ditanyakan.

Roboguru adalah sebuah fitur dari aplikasi pembelajaran Ruangguru yang berupa mesin pencarian yang dapat membantu siswa dalam menjawab soal dari berbagai mata pelajaran yang ada di tingkat SD, SMP, SMA, maupun UTBK. Roboguru dapat menerima masukan berupa string maupun gambar, namun yang akan dibahas dalam makalah ini adalah masukan string. Dalam mencari jawaban yang paling mendekati soal, digunakan algoritma pencocokan *string* atau *string matching* dan *regular expression*.

II. TEORI DASAR

A. *Pencocokan String*

String adalah sebuah deret simbol dan umumnya dapat dipandang sebagai tipe data yang digunakan untuk menyimpan barisan dari karakter sedangkan *pattern* adalah string dengan panjang tertentu yang akan dicari keberadaannya dalam sebuah teks (*string*)

Pencarian sebuah *pattern* di dalam sebuah teks dapat dilakukan dengan menggunakan sebuah algoritma yang dinamakan dengan algoritma pencocokan string atau *string matching*. Algoritma ini mencari apakah sebuah pattern merupakan substring dari sebuah teks. Misalkan teks adalah string dengan panjang m dan pattern adalah string dengan panjang n maka asumsikan n < m.

Jika ada sebuah string $S = X_0X_1...X_{m-1,}$ maka:

- *Prefix* dari S adalah substring S[0 .. k] dengan k adalah sebuah indeks di antara 0 sampai m-1. Misalkan S = “stima”, maka prefix dari S = “s”, “st”, “sti”, “stim”, “stima”
- *Suffix* dari S adalah substring S[k .. m-1] dengan k adalah sebuah indeks di antara 0 sampai m-1. Misalkan S = “stima”, maka suffix dari S = “a”, “ma”, “ima”, “tima”, “stima”

Penerapan dari algoritma pencocokan string adalah pada pencarian *pattern* di dalam *text editor*, pencarian di *web search engine* seperti Google, dalam analisis citra, ekstraksi informasi maupun dalam bidang bioinformatika seperti pencocokan rantai asam amino pada rantai DNA.

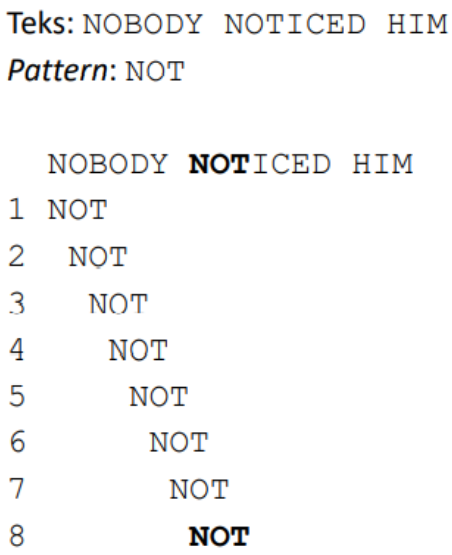
Beberapa algoritma yang umumnya digunakan untuk melakukan pencocokan string adalah algoritma *brute-force*, algoritma Knuth-MorrisPratt (KMP), algoritma Boyer-Moore, dan *Regular Expression*.

B. Algoritma brute-force

Algoritma *brute-force* melakukan pencocokan *string* dengan cara mengecek apakah pola pada *pattern* P terdapat pada teks T dimulai dari karakter pertama T sampai karakter ke m – n + 1 atau sampai ditemukan *pattern* yang sesuai pada teks.

Secara garis besar, cara kerja algoritma brute-force adalah sebagai berikut:

1. Pattern disejajarkan pada awal teks
2. Telusuri pattern dari kiri sampai ke kanan dengan membandingkan karakter pada pattern dengan karakter dengan urutan bersesuaian pada teks
3. Setelah melakukan perbandingan, jika semua karakter pada pattern cocok dengan karakter pada teks maka pencarian berhasil
4. Jika ketika melakukan perbandingan ditemukan karakter yang tidak cocok maka pencarian bellum berhasil dan geser pattern satu karakter ke kanan. Lalu, ulangi kembali langkah 2.



Gambar 1. Ilustrasi String Matching dengan Algoritma brute-force
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf

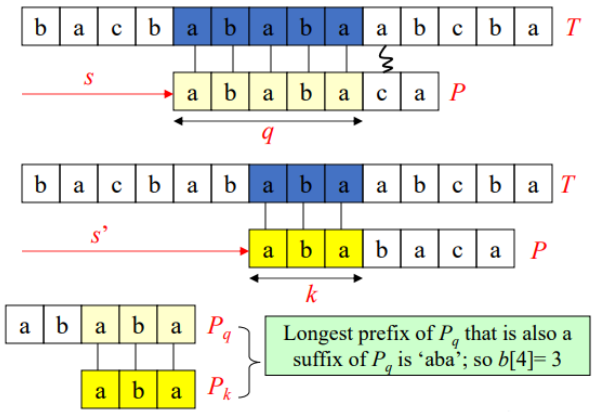
Algoritma *brute-force* akan lebih efektif jika karakter yang dicek bervariasi (A..Z, a..z, 1..9) sedangkan jika karakter yang dicek hanya terdiri dari beberapa jenis saja seperti 0 dan 1 maka algoritma *brute-force* kurang efektif.

Kasus terburuk dengan algoritma ini adalah ketika dilakukan *pattern matching*, karakter yang berbeda adalah pada karakter terakhir sehingga harus dilakukan perbandingan sebanyak m – n + 1 kali.

C. Algoritma Knuth-Morris-Pratt (KMP)

Algoritma ini memiliki kemiripan dengan algoritma *brute-force* yaitu pengecekan *string* tetap dilakukan dari kiri ke kanan. Namun, yang membedakan antara algoritma KMP dengan algoritma *brute-force* adalah jumlah pergeseran yang dilakukan ketika terjadi *mismatch* sehingga algoritma KMP bisa dikatakan lebih cerdas daripada algoritma *brute-force*. Pada algoritma *brute-force*, jumlah pergeseran yang dilakukan adalah sebesar 1 ketika terjadi *mismatch* namun dalam algoritma KMP, jumlah pergeseran akan dihitung terlebih dahulu sehingga tidak perlu dilakukan proses pencocokan string terhadap semua karakter pada teks.

Algoritma ini ditemukan oleh Donald Erwin Knuth, J. H. Moris, dan V. R. Pratt sehingga dinamakan Algoritma Knuth-Morris-Pratt.



Gambar 2. Ilustrasi String Matching dengan Algoritma Knuth-Morris-Pratt
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

Secara garis besar, cara kerja algoritma Kruth-Morris-Pratt adalah sebagai berikut:

1. Pattern disejajarkan pada awal teks
2. Telusuri pattern dari kiri sampai kanan dengan membandingkan karakter pada pattern dengan karakter dengan urutan bersesuaian pada teks
3. Setelah melakukan perbandingan, jika semua karakter pada pattern cocok dengan karakter pada teks maka pencarian berhasil
4. Jika ketika melakukan perbandingan ditemukan karakter yang tidak cocok, maka hitung fungsi pinggiran (border/failure function) untuk menentukan berapa pergeseran yang harus dilakukan. Ulangi kembali langkah 2.

Fungsi pinggiran adalah ukuran terbesar dari prefix P[0..k] yang juga merupakan suffix dari P[2..k] dengan k = j – 1, yang mana j adalah posisi terjadinya mismatch di pattern P. Contoh perhitungan fungsi pinggiran adalah sebagai berikut.

• 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 3. Perhitungan Fungsi Pinggiran
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

Kelebihan dari algoritma ini adalah tidak perlu bergerak mundur sehingga cocok untuk memproses file atau input berukuran besar. Kelemahan algoritma ini adalah algoritma KMP tidak terlalu efektif jika ukuran *alphabet* cukup besar karena akan memperbesar peluang untuk terjadinya *mismatching*.

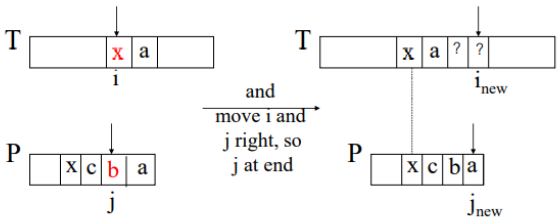
D. Algoritma Boyer-Moore

Tidak seperti algoritma brute-force dan KMP, algoritma Boyer-Moore melakukan perbandingan string yang bergerak dari kanan ke kiri. Pattern matching dalam algoritma ini terdiri atas 2 teknik yaitu:

1. Teknik looking-glass yaitu mencari pattern P pada teks T dari belakang melalui P
2. Teknik character-jump yaitu ketika terjadi mismatching di T[i] == x (karakter di P[j] != T[i])

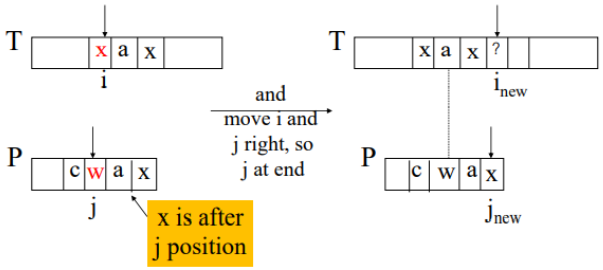
Ada 3 kemungkinan teknik character-jump yang harus dicek secara berurutan:

1. Jika P mengandung x, geser P ke kanan untuk mensejajarkan last occurrence dari x di P dengan T[i]



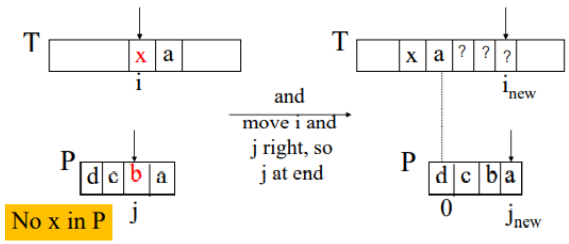
Gambar 4. Teknik 1 Character-jump
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

2. Jika P mengandung x, tetapi P tidak bisa digeser ke kanan untuk mensejajarkan last occurrence dari x di P dengan T[i], maka geser P ke kanan sebanyak 1 karakter ke T[i+1]



Gambar 5. Teknik 2 Character-jump
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

3. Jika kemungkinan 1 dan 2 tidak bisa diaplikasikan, maka geser P agar P[0] sejajar dengan T[i+1]

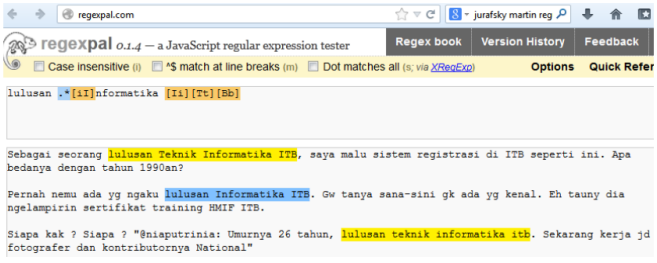


Gambar 6. Teknik 3 Character-jump
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

Algoritma Boyer-Moore lebih efektif ketika ukuran dari alphabet besar dan jauh lebih efektif daripada algoritma brute-force dalam melakukan searching pada bahasa Inggris.

E. Regular Expression

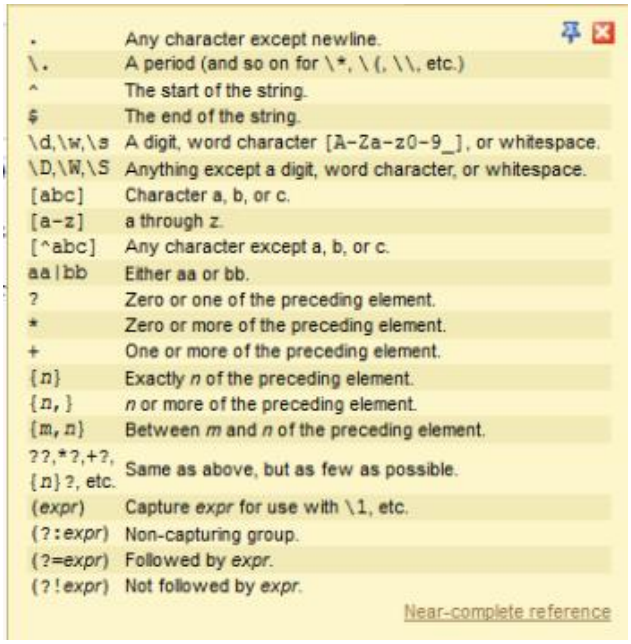
Regular Expression adalah sebuah algoritma yang dapat digunakan untuk mencari pattern atau pola tertentu pada teks. Perbedaaan algoritma ini dengan algoritma-algoritma lain adalah selain bisa mencari pattern pada teks dengan *exact matching* (pattern P harus persis di teks T), regular expression bisa mencari pattern yang memiliki pola-pola serupa di teks T sehingga tidak terbatas hanya pada satu pola saja.



Gambar 7. Contoh Regular Expression
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-

[2019/String-Matching-dengan-Regex-2019.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf)

Regular expression bisa digunakan dalam membuat chatbot, fitur searching pada website, dan lain-lain. Sintaks yang bisa digunakan untuk membuat regular expression adalah sebagai berikut.



.	Any character except newline.
\.	A period (and so on for *, \(, \\, etc.)
^	The start of the string.
$	The end of the string.
\d,\w,\s	A digit, word character [A-Za-z0-9_], or whitespace.
\D,\W,\S	Anything except a digit, word character, or whitespace.
[abc]	Character a, b, or c.
[a-z]	a through z.
[^abc]	Any character except a, b, or c.
aa|bb	Either aa or bb.
?	Zero or one of the preceding element.
*	Zero or more of the preceding element.
+	One or more of the preceding element.
{n}	Exactly *n* of the preceding element.
{n,}	*n* or more of the preceding element.
{m,n}	Between *m* and *n* of the preceding element.
??,*?,+?, {n}?, etc.	Same as above, but as few as possible.
(expr)	Capture *expr* for use with \1, etc.
(?:expr)	Non-capturing group.
(?=expr)	Followed by *expr*.
(?!expr)	Not followed by *expr*.

Near-complete reference

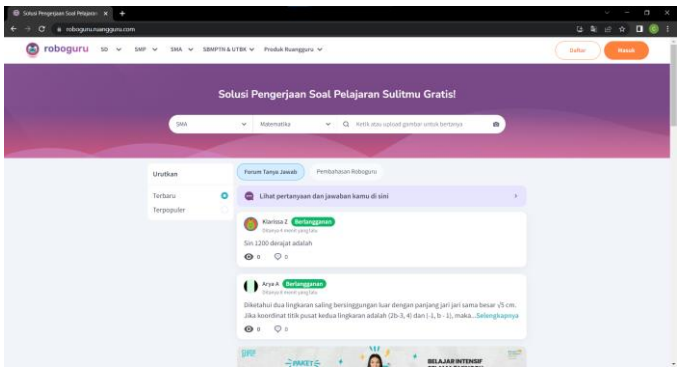
Gambar 8. Sintaks Regular Expression
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf

III. PEMBAHASAN

A. *Roboguru*

Ruangguru adalah sebuah perusahaan teknologi di Indonesia yang berfokus pada bidang pendidikan. Perusahaan ini didirikan oleh Belva Devara dan Iman Usman pada tahun 2014. Ruangguru menyediakan berbagai layanan belajar berbasis teknologi, seperti layanan kelas virtual, platform ujian online, video belajar berlangganan, les privat, platform tanya-jawab soal, dan lain-lain. Ruangguru percaya bahwa teknologi dapat memudahkan siswa untuk mendapatkan pembelajaran yang berkualitas di mana pun mereka berada dan meyakini bahwa teknologi dapat membantu siswa, guru, orang tua untuk menjalankan aktivitas dengan efisien dan efektif.

Salah satu produk dari Ruangguru adalah Roboguru, sebuah fitur untuk mencari solusi pengerjaan soal pelajaran. Roboguru akan membantu siswa menemukan jawaban dan memahami bagaimana proses menjawab soal-soal yang sulit. Roboguru dapat diakses melalui web atau aplikasi yang dapat diunduh.

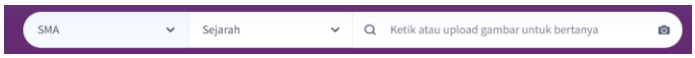


Gambar 9. Website Roboguru

Siswa dapat bertanya soal dari berbagai macam pelajaran yang ada pada tingkat SD, SMP, SMA, maupun UTBK/SBMPTN. Setelah memasukkan pertanyaan, Roboguru akan mencari dan menampilkan 10 soal yang memiliki konsep yang mirip dengan query serta pembahasan dari soal tersebut sehingga siswa tidak perlu kebingungan lagi dalam mengerjakan soal.

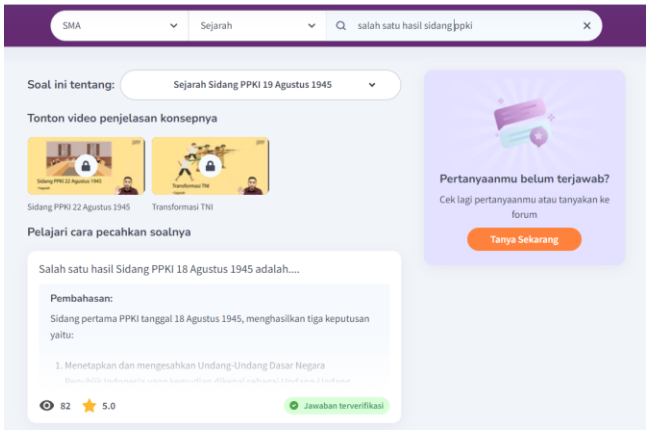
B. *Algoritma String Matching dan Regex pada Roboguru*

Ketika siswa mencari soal yang ingin ditemukan jawabannya pada situs Roboguru, siswa dapat memilih kategori tingkat yaitu SD/SMP/SMA/UTBK lalu mata pelajaran yang ingin dicari untuk mempermudah pencarian soal.



Gambar 10. Kategori Pencarian Soal

Setelah itu siswa dapat memasukkan query yang ingin dicari jawabannya, misalnya input yang dimasukkan adalah untuk mata pelajaran Sejarah SMA yaitu “salah satu hasil sidang PPKI” maka hasil keluaran dari Roboguru adalah sebagai berikut.



Gambar 11. Hasil Pencarian Soal Sejarah

Dapat dilihat bahwa hasil keluaran searching pada soal tidak melakukan *exact matching* pada query sehingga algoritma string matching yang digunakan bukanlah algoritma brute-force, algoritma Kruth-Morris-Pratt, maupun algoritma Boyer-Moore melainkan menggunakan *regular expression*. Dengan menggunakan regular expression, hasil pencarian dapat mengeluarkan soal-soal yang memiliki pola atau pattern yang mirip dengan query beserta jawabannya. Namun, penggunaan regular expression tidak dilakukan untuk melakukan parsing karena input dari user berkategori sama, tidak dipisahkan menjadi kategori-kategori lain lagi. Berikut merupakan salah satu input lain pada kategori pelajaran Matematika SMA.

Gambar 11. Hasil Pencarian Soal Matematika

Dari hasil pencarian, pattern tidak harus berada di awal kalimat, bisa di mana pun yang penting memiliki pola yang mirip. Dapat dilihat juga bahwa penggunaan huruf kapital tidak mempengaruhi hasil pencarian. Selain itu, jika query yang diinput tidak memenuhi string matching dengan data yang ada pada database, maka Roboguru akan mengeluarkan output bahwa tidak ada pembahasan yang memenuhi.

Gambar 10. Hasil Pencarian Soal Tidak Ditemukan

IV. KESIMPULAN

Algoritma string matching sangat memiliki banyak manfaat dan aplikasi, terutama penggunaan *Regular Expression* yang banyak digunakan pada mesin pencarian. Selain itu, algoritma string matching lain juga memiliki kelebihan dan kelemahannya masing-masing sehingga harus diaplikasikan dengan baik agar memberikan hasil yang optimal.

VIDEO LINK AT YOUTUBE

https://youtu.be/RBN5KwEONIU

UCAPAN TERIMA KASIH

Puji syukur penulis ucapkan kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya makalah ini dapat diselesaikan dengan baik dan tepat waktu. Penulis ingin mengucapkan terima kasih kepada dosen pengampu mata kuliah Strategi Algoritma atas bimbingannya selama ini. Ucapan terima kasih juga penulis ucapkan kepada keluarga dan teman-teman penulis yang memberikan dukungan dalam pengerjaan makalah ini. Penulis meminta maaf jika terdapat kesalahan kata atau ucapan. Akhir kata, semoga makalah ini dapat bermanfaat bagi kita semua.

REFERENSI

[1] *https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf* diakses pada 19 Mei 2022 pukul 19.25

[2] *https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf* diakses pada 20 Mei 2022 pukul 20.08

[3] *https://www.ruangguru.com/* diakses pada 20 Mei 2022 pukul 21.46

[4] *https://catatananalgo.wordpress.com/2016/10/02/algoritma-string-matching-pencocokan-string/* diakses pada 21 Mei 2022 pukul 22.31

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 2022

Claudia (13520076)