

LAPORAN TUGAS BESAR

IF2211 STRATEGI ALGORITMA

Pemanfaatan Pattern Matching dalam Membangun Sistem Deteksi Individu Berbasis
Biometrik Melalui Citra Sidik Jari



Disusun oleh:

Renaldy Arief Susanto	13522022
Abdul Rafi Radityo Hutomo	13522089
Rayhan Fadhlan Azka	13522095

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2024

DAFTAR ISI

DAFTAR ISI	2
BAB I	
DESKRIPSI TUGAS	3
BAB II	
LANDASAN TEORI	4
2.1 String Matching	4
2.2 Algoritme Knuth-Morris Pratt	4
2.3 Algoritme Boyer Moore	5
2.4 Regular Expression (Regex)	5
2.5 Teknik Pengukuran Persentase Kemiripan	6
2.6 Aplikasi Desktop Menggunakan .NET dan WinForms	6
BAB III	
ANALISIS PEMECAHAN MASALAH	8
3.1 Langkah-Langkah Pemecahan Masalah	8
3.2 Proses Penyelesaian Solusi dengan Algoritme KMP dan BM	9
3.2.1 Knuth-Morris Pratt (KMP)	9
3.2.2 Boyer Moore (BM)	9
3.3 Fitur Fungsional dan Arsitektur Aplikasi Desktop yang Dibangun	10
3.4 Contoh Ilustrasi Kasus	10
BAB IV	
IMPLEMENTASI DAN PENGUJIAN	12
4.1 Spesifikasi Teknis Program	12
4.1.1 Struktur Data, Fungsi, dan Prosedur Controller GUI	12
4.1.2 Struktur Data, Fungsi, dan Prosedur Algoritme	13
4.1.3 Struktur Data, Fungsi, dan Prosedur Interaksi Basis Data	14
4.2 Penjelasan Tata Cara Penggunaan Program	14
4.3 Hasil Pengujian	15
4.4 Analisis Hasil Pengujian	20
BAB V	
KESIMPULAN DAN SARAN	21
5.1 Kesimpulan	21
5.2 Saran	21
LAMPIRAN	22
DAFTAR PUSTAKA	23

BAB I

DESKRIPSI TUGAS

Di era digital ini, keamanan data dan akses menjadi semakin penting. Perkembangan teknologi membuka peluang untuk berbagai metode identifikasi yang canggih dan praktis. Beberapa metode umum yang sering digunakan seperti kata sandi atau pin, namun memiliki kelemahan seperti mudah terlupakan atau dicuri. Oleh karena itu, biometrik menjadi alternatif metode akses keamanan yang semakin populer. Salah satu teknologi biometrik yang banyak digunakan adalah identifikasi sidik jari. Sidik jari setiap orang memiliki pola yang unik dan tidak dapat ditiru, sehingga cocok untuk digunakan sebagai identitas individu.

Pattern matching merupakan teknik penting dalam sistem identifikasi sidik jari. Teknik ini digunakan untuk mencocokkan pola sidik jari yang ditangkap dengan pola sidik jari yang terdaftar di database. Algoritme pattern matching yang umum digunakan adalah Bozorth dan Boyer-Moore. Algoritme ini memungkinkan sistem untuk mengenali sidik jari dengan cepat dan akurat, bahkan jika sidik jari yang ditangkap tidak sempurna.

Dengan menggabungkan teknologi identifikasi sidik jari dan pattern matching, dimungkinkan untuk membangun sistem identifikasi biometrik yang aman, handal, dan mudah digunakan. Sistem ini dapat diaplikasikan di berbagai bidang, seperti kontrol akses, absensi karyawan, dan verifikasi identitas dalam transaksi keuangan.

Di tugas ini, kami mengimplementasikan sistem yang dapat melakukan identifikasi individu berbasis biometrik dengan menggunakan sidik jari. Metode yang akan digunakan untuk melakukan deteksi sidik jari adalah Boyer-Moore dan Knuth-Morris-Pratt. Selain itu, sistem ini akan dihubungkan dengan identitas sebuah individu melalui basis data sehingga harapannya terbentuk sebuah sistem yang dapat mengenali identitas seseorang secara lengkap hanya dengan menggunakan sidik jari.

BAB II

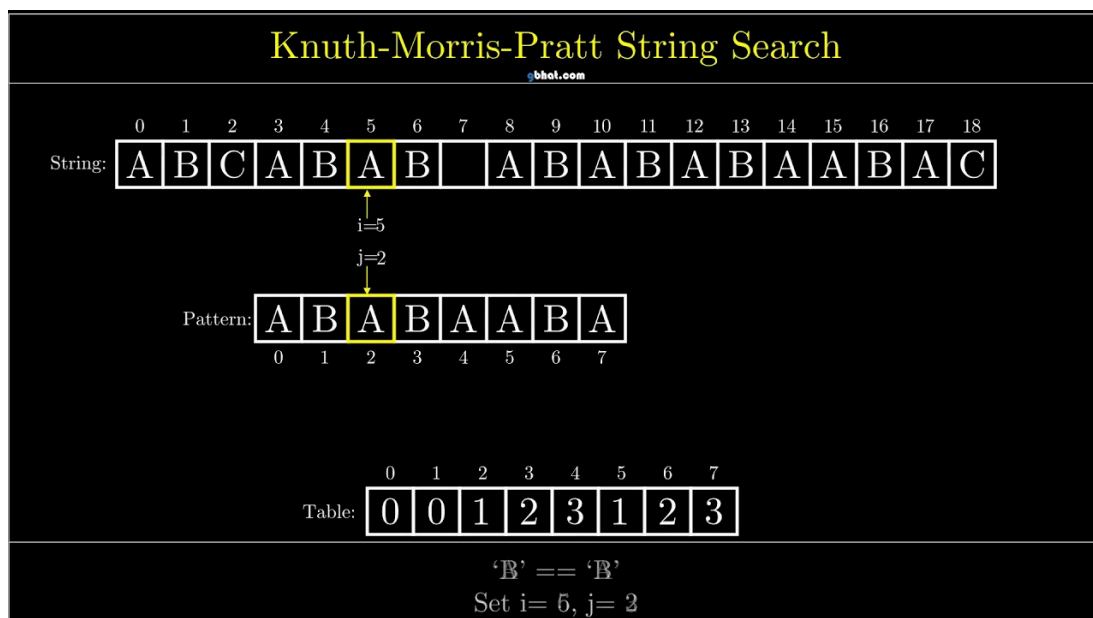
LANDASAN TEORI

2.1 String Matching

String matching adalah proses mencari keberadaan atau kemunculan suatu pola (pattern) tertentu di dalam sebuah teks (string). String matching sendiri mempunyai berbagai kegunaan, diantaranya adalah pencarian pada editor teks, web search engine (misal : Google), analisis citra, dan bioinformatika (misal : pencocokan rantai DNA).

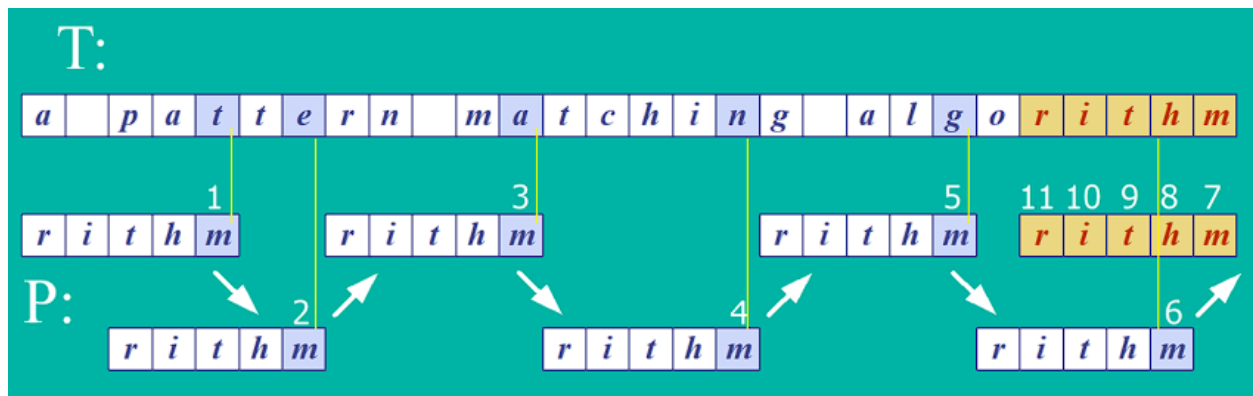
2.2 Algoritme Knuth-Morris Pratt

Algoritme Knuth-Morris-Pratt (KMP) adalah salah satu algoritme yang efisien untuk string matching. Algoritme ini diperkenalkan oleh Donald Knuth, Vaughan Pratt, dan James H. Morris pada tahun 1977. Algoritme KMP bekerja dengan menggunakan informasi dari pattern itu sendiri yang disini dinamakan *border function* atau fungsi pinggiran yang berguna untuk menghindari perbandingan ulang yang tidak perlu saat terjadi ketidakcocokan. Algoritme ini bekerja dengan kompleksitas $O(N + M)$, dimana N adalah panjang teks, dan M adalah panjang pattern. Algoritme KMP bekerja dengan optimal ketika ukuran alphabetnya kecil (misal : perbandingan binary), algoritme ini bekerja tidak terlalu baik ketika ukuran alphabet besar karena akan sering terjadi ketidakcocokan di awal perbandingan.



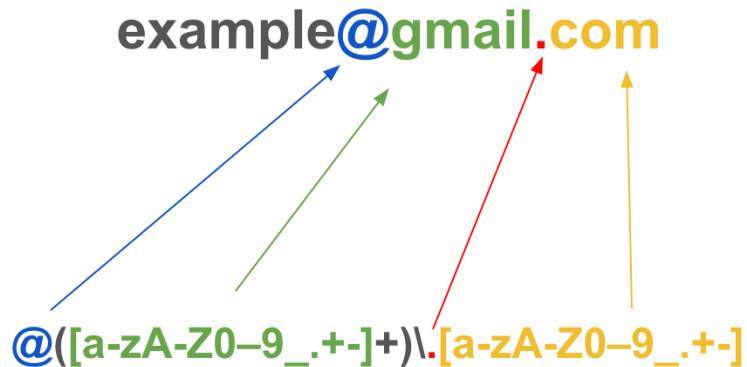
2.3 Algoritme Boyer Moore

Algoritme Boyer-Moore adalah salah satu algoritme string matching yang efisien dan banyak digunakan, terutama ketika pola yang dicari relatif panjang dibandingkan dengan teks. Algoritme ini dibagi menjadi dua teknik, yaitu looking glass technique (temukan P (pattern) di T (teks) dengan bergerak mundur melalui P, dimulai dari ujungnya) dan character jump technique (Teknik ini menggunakan tabel Last Occurrence untuk menentukan seberapa jauh pola dapat digeser saat terjadi ketidakcocokan). Algoritme ini rata-rata bekerja dengan kompleksitas $O(N + M)$, dimana N adalah panjang teks, dan M adalah panjang pattern, namun pada kasus terburuk, algoritme ini dapat bekerja dengan kompleksitas $O(NM + A)$, dengan A adalah ukuran alfabet. Algoritme Boyer-Moore bekerja dengan cepat jika alfabet (A) besar, namun lambat ketika alfabetnya kecil, contohnya adalah algoritme ini bagus untuk teks bahasa Inggris, buruk untuk binary.



2.4 Regular Expression (Regex)

Regular Expressions, atau yang lebih dikenal dengan sebutan Regex, adalah sebuah alat yang sangat bagus untuk mencocokkan pola teks atau string. Regex bekerja dengan mencocokkan teks dengan pattern atau pola yang diberikan. Regex memungkinkan pengguna untuk mencari, mengedit, dan memanipulasi teks dengan cara yang efisien dan fleksibel.



2.5 Teknik Pengukuran Persentase Kemiripan

Dalam pencarian sidik jari, terkadang sidik jari yang dimasukkan oleh pengguna tidak terlalu baik kualitasnya, baik terdapat node atau kecacatan pada beberapa pixel, oleh karena itu terdapat kejadian dimana sidik jari yang dimasukkan tidak ada yang exact match dengan sidik jari yang terdapat di basis data, oleh karena itu diperlukan suatu teknik untuk mengukur persentase kemiripan dua sidik jari. Dalam hal ini, kami menggunakan algoritme Longest Common Subsequence (LCS) untuk mencari tingkat kemiripan antara dua buah sidik jari. Algoritme LCS sendiri bekerja dengan cara mencari sequence terpanjang diantara dua buah masukan, contohnya jika masukan adalah “AGGTAB” dan “GXTXAYB” maka longest common subsequencenya adalah “AGGTAB” dan “GXTXAYB” dengan panjang longest common subsequence nya adalah 4. Dalam mengukur persentase kemiripan dua buah string, kami menggunakan rumus

$$\frac{lcs}{len(str1) + len(str2)} \cdot 100$$

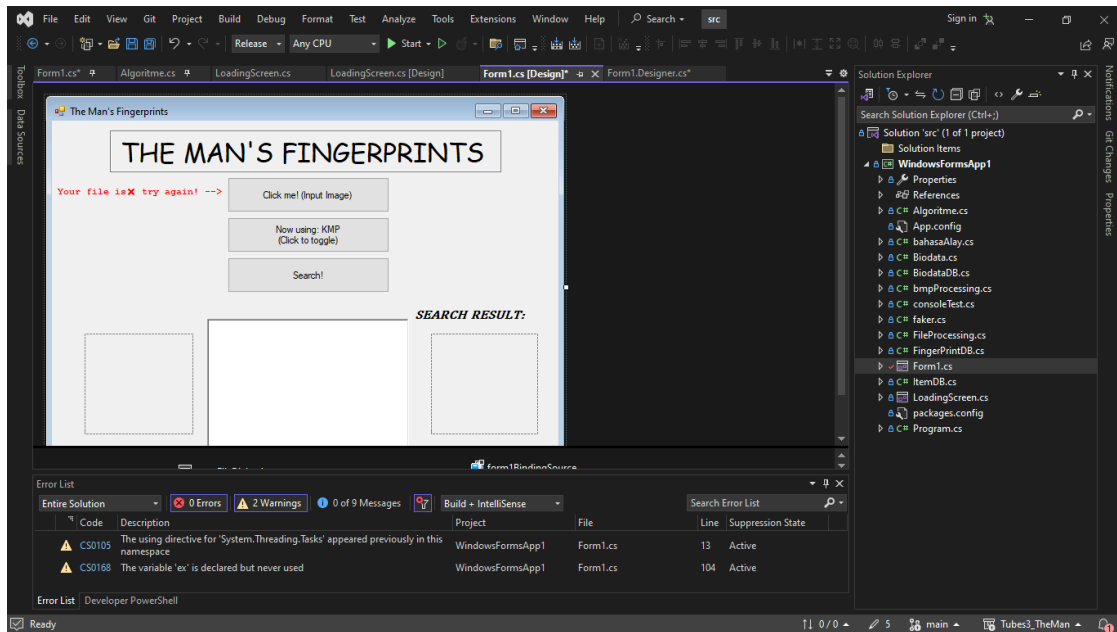
Setelah didapat tingkat kemiripan ini, barulah kita bisa menentukan sidik jari mana yang paling mirip.

2.6 Aplikasi Desktop Menggunakan .NET dan WinForms

Aplikasi desktop dikembangkan menggunakan kakas Windows Forms (disingkat WinForms). WinForms sendiri dibangun menggunakan *framework* **.NET**, yaitu sebuah *framework* pengembangan perangkat lunak yang bersifat *open-source*. Penggunaan WinForms pada aplikasi kami bersifat sederhana dan hanya menggunakan fitur-fitur dasar seperti penggunaan elemen-elemen stok yang sudah tersedia.

Oleh karena kakas WinForms berbasis bahasa C#, seluruh pemrograman dibuat dalam bahasa C# juga. Ini mencakup kode untuk logika algoritme, GUI, dan pengelolaan database (akan dijelaskan lebih lanjut pada bab 3 dan 4). Kakas WinForms sudah terintegrasi pada IDE

Visual Studio sehingga pengembangan GUI aplikasi dilakukan dengan mudah menggunakan fitur *design view* yang terdapat pada Visual Studio.



Namun daripada itu, kekurangan dari penggunaan WinForms adalah aplikasi tidak secara langsung kompatibel dengan OS non-Windows. Pengguna Linux, contohnya, harus menggunakan *adapter software* agar aplikasi dapat dijalankan (contohnya *wine*).

2.6 Kriptografi Kunci Publik

Kriptografi kunci publik, juga dikenal sebagai kriptografi asimetris, adalah bidang sistem kriptografi yang menggunakan pasangan kunci terkait. Setiap pasangan kunci terdiri dari kunci publik dan kunci privat yang sesuai. Pasangan kunci dibuat dengan algoritme kriptografi berdasarkan masalah matematika yang disebut fungsi searah. Keamanan kriptografi kunci publik bergantung pada kerahasiaan kunci privat; kunci publik dapat didistribusikan secara terbuka tanpa mengurangi keamanan.

Dalam sistem enkripsi kunci publik, siapa saja yang memiliki kunci publik dapat mengenkripsi pesan, menghasilkan ciphertext, tetapi hanya mereka yang mengetahui kunci privat yang sesuai yang dapat mendekripsi ciphertext untuk mendapatkan pesan asli.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah-Langkah Pemecahan Masalah

Pada persoalan kali ini, kami membuat program yang dapat mencocokkan suatu sidik jari dengan banyak sidik jari yang berada di basis data, dan mendapatkan biodata orang dengan sidik jari tersebut. Dalam menyelesaikan persoalan itu, kami menemukan bahwa citra sidik jari dapat dikonversikan menjadi string binary (kumpulan kata yang berisi hanya 0 dan 1), dalam hal ini string binary tersebut sangat panjang, sehingga kami mengkonversikan lagi menjadi ASCII 8 bit, setelah mendapatkan ASCII 8 bit tersebut, kami dapat melakukan proses string matching dengan menggunakan algoritme Knuth-Morris Pratt (KMP), Boyer Moore (BM), atau Longest Common Subsequence (LCS). Secara detail, langkah-langkah yang kami gunakan dalam memecahkan permasalahan ini adalah :

1. Konversikan seluruh gambar sidik jari di database dan sidik jari orang yang ingin dicocokkan menjadi ASCII, yaitu dengan cara sebagai berikut.
2. Dalam mengonversikan gambar, pertama iterasi seluruh pixel pada gambar bitmap, dan dapatkan nilai grayscale dari masing-masing pixelnya, nilai grayscale dari sebuah pixel berada di antara 0-255, lalu dari nilai grayscale tersebut, akan dikonversi lagi menjadi 0 atau 1. 0 adalah pixel yang melambangkan warna hitam dan 1 warna putih, dalam kasus ini, jika nilai grayscale < 128 maka akan dikonversi menjadi 0 dan sebaliknya.
3. Kelompokkan binary 0 dan 1 tersebut sebanyak 8 buah, untuk tiap 8 digit binary akan menghasilkan 1 huruf ASCII, jika kita menggunakan dataset bawaan dengan ukuran per gambar 100x100 pixel, maka akan menghasilkan string ASCII dengan panjang kurang lebih 1250, untuk string yang akan menjadi pattern atau sidik jari input, binary yang diambil hanya 64 pixel, dan jika di convert menjadi ASCII maka menghasilkan string ASCII dengan panjang 8, string inilah yang digunakan sebagai pattern.
4. Lakukan pencocokan sesuai dengan algoritme yang ingin digunakan oleh user (KMP/BM) dengan pattern string berupa ASCII dengan panjang 8 tadi, dan teks berupa ASCII representasi seluruh pixel. Pencocokan ini dilakukan sampai menemukan gambar yang cocok. Jika menemukan gambar yang paling cocok, tahap 5 dan 6 tidak perlu dilakukan.
5. Jika sudah melakukan iterasi ke seluruh gambar dan tidak menemukan teks yang exact match dengan patternnya, maka akan pencarian diulangi namun

menggunakan Longest Common Subsequence. Pada bagian ini, gambar yang ingin dicocokkan dan gambar di database diambil 64 digit ASCII ditengah, lalu lakukan pencocokan dengan LCS.

6. Ambil 50 sidik jari dengan tingkat kemiripan tertinggi dari pencocokan dengan LCS pada tahap 5, lalu kali ini cocokkan ulang dengan LCS lagi, namun ASCII yang digunakan adalah seluruh gambar, bukan hanya 64 digit ASCII. Teknik pemfilteran ini digunakan agar hasil yang didapatkan akurat, lalu simpan sidik jari paling cocok serta tingkat kemiripannya.
7. Karena data nama yang terdapat di tabel sidik_jari dibuat corrupt, maka kami menggunakan regex untuk membuat pattern regex berdasarkan nama alay yang terdapat di tabel sidik_jari, dari pattern regex tersebut lalu akan dicocokkan ke tabel biodata, lalu akan ditampilkan informasi biodata orang tersebut beserta sidik jari yang bersesuaian.

3.2 Proses Penyelesaian Solusi dengan algoritme KMP dan BM

Pencarian solusi dengan menggunakan algoritme KMP dan BM bersifat exact match, jika tidak ditemukan pattern yang exact match pada , maka akan dicari ke yang paling mirip menggunakan LCS, seperti yang telah dijelaskan pada sub bab 3.1.

3.2.1 Knuth-Morris Pratt (KMP)

1. Dengan menggunakan pattern, buatlah tabel LPS (longest prefix suffix). Tabel LPS ini nantinya digunakan sebagai acuan dalam pergeseran mencocokkan pattern dan string.
2. Inisialisasi nilai i dan j menjadi nol. Nilai i merupakan indeks pada teks yang dicari, sedangkan nilai j merupakan indeks pada pattern.
3. Lakukan perbandingan dari kiri ke kanan. Jika karakter yang dibandingkan sama, tambah nilai i dan j. Jika ditemukan ketidaksesuaian karakter, geser pattern ke prefix terbesar dari teks $[0..j-1]$ yang merupakan suffix dari $P[1..j-1]$. Pencarian pergeseran ini mengacu kepada tabel LPS yang telah dibuat sebelumnya.
4. Lakukanlah pencarian hingga pattern ditemukan atau teks yang dicari sudah sampai akhir.

3.2.2 Boyer Moore (BM)

1. Dengan menggunakan pattern, buatlah tabel Last Occurence berukuran 256 (sesuai dengan 8 bit ASCII), lalu isi tabel dengan index kemunculan terakhir tiap karakter.
2. Algoritme ini melakukan perbandingan dari kanan. Inisialisasi nilai s dan j menjadi 0. Nilai s merupakan indeks pada teks, sedangkan nilai j merupakan indeks pada pattern.

3. Misalkan P merupakan pattern yang dicari, sedangkan T merupakan teks tempat pattern dicari, jika ditemukan ketidaksesuaian karakter, pergeseran dilakukan sehingga kemunculan terakhir x di P bersesuaian dengan $T[j]$.
4. Jika tidak terdapat kemunculan terakhir, maka geser pattern sehingga $P[0]$ bersesuaian dengan $T[j+1]$.
5. Lakukanlah pencarian hingga pattern ditemukan atau teks yang dicari sudah sampai akhir.

3.3 Fitur Fungsional dan Arsitektur Aplikasi Desktop yang Dibangun

Fitur-fitur fungsional aplikasi sebagai berikut.

1. Aplikasi dapat memuat dataset SOCOFing, yaitu sebuah dataset yang berisi 6000 buah foto sidik jari (dengan format file .bmp).
2. Aplikasi dapat menyimpan biodata orang menggunakan basis data.
3. Aplikasi dapat menerima masukan berupa berkas foto/citra dengan format .bmp.
4. Aplikasi dapat menelusuri dataset SOCOFing untuk mencari foto sidik jari yang paling mirip dengan foto masukan.
5. Aplikasi menyediakan dua pilihan algoritme untuk mencocokkan gambar, yaitu algoritme Knuth-Morris-Pratt dan Boyer-Moore.
6. Aplikasi dapat mencari biodata orang yang mempunyai foto sidik jari tersebut, berdasarkan namanya yang sudah berubah menjadi bentuk *alay*.
7. Aplikasi dapat menampilkan hasil pencarian berupa foto sidik jari termirip, biodata orang terkait, tingkat kemiripan, serta waktu eksekusi.
8. Aplikasi dapat digunakan untuk melakukan pencarian berulang kali.

Arsitektur aplikasi bersifat sederhana. Aplikasi hanya terdiri dari sebuah halaman inti yang memuat keseluruhan antarmuka. Pada halaman ini, pengguna dapat mengakses semua fitur fungsional yang disediakan aplikasi yang bersifat interaktif. Aplikasi akan menerima masukan, menampilkan hasil pencarian, serta menampilkan informasi lainnya pada halaman ini.

Namun, sebelum aplikasi dapat berjalan, proses pemuatan dataset SOCOFing menggunakan waktu yang cukup lama sehingga dibuat sebuah halaman *loading screen* untuk memberitahu pengguna bahwa aplikasi dalam proses *launching*.

3.4 Contoh Ilustrasi Kasus

Akan dijelaskan contoh ilustrasi penyelesaian persoalan untuk uji kasus berikut. Diberikan masukan berupa gambar sidik jari. Ingin dicari siapa pemilik sidik jari ini, jika merujuk ke data pada basis data.

Berikut adalah gambar masukan.



1. Pertama, gambar akan dikonversi menjadi string binary. Anggap hasil konversinya adalah string P .
2. Akan diambil 64 karakter tengah dari string ini. Katakan hasilnya sebagai P' .
3. P' kemudian dikonversi menjadi bentuk ascii. Katakan hasilnya adalah P'' .
4. Hal yang sama telah dilakukan untuk semua gambar bmp pada basis data. Namun, untuk gambar pada basis data, binary tidak diambil 64 karakter tengah, melainkan diambil seluruhnya. Anggap string-string hasil konversi tersebut diletakkan pada sebuah array bernama S .
5. Untuk semua i , P'' akan dicocokkan dengan $S[i]$ hingga semua elemen ditelusuri, atau hingga P'' ditemukan pada salah satu $S[i]$.
6. Jika ditemukan, program akan meng-*query* basis data untuk gambar tersebut dan menemukan nama dalam bahasa *alay*, sebutlah N . Jika tidak, program akan mencari sidik jari yang paling mirip dengan ukuran *longest common subsequence*.
7. N akan dikonversi menjadi sebuah pola regex, kemudian program akan meng-*query* basis data untuk daftar semua nama. dicari nama yang memenuhi regex tersebut.
8. Program akan menampilkan biodata dengan nama yang memenuhi pola regex tersebut, beserta hasil pencarian lainnya.

3.5 Pemecahan Masalah Bonus (Enkripsi)

Data yang terdapat dalam KTP merupakan data yang privat dan tidak seharusnya mudah diakses orang yang tidak berwenang. Oleh karena itu, ada kebutuhan untuk melakukan enkripsi dari data yang ada pada basis data, agar apabila ada pihak luar yang berhasil mengakses data SQL, data yang didapat adalah data yang tidak bermakna.

Pada tugas besar ini sistem enkripsi yang digunakan adalah Okamoto-Uchiyama Cryptosystem. Sistem ini adalah sebuah sistem enkripsi kunci publik yang bekerja berdasarkan prinsip bahwa faktorisasi bilangan bulat adalah NP problem. Sehingga, meskipun pada kasus ini kunci publik adalah hasil perkalian dari kunci privat, kunci privat tetap tidak dapat dicari dengan mudah.

Proses generasi key pada sistem ini adalah sebagai berikut.

1. Pilih p dan q , yaitu bilangan prima berukuran besar yang akan menjadi private key
2. Public key dari sistem ini adalah (n, g, h) dengan ketentuan sebagai berikut :

$$\begin{aligned}
 n &= p^2 q \\
 g &\in \{2 \dots n - 1\}, g^{p-1} \not\equiv 1 \pmod{p^2} \\
 h &= g^n \pmod{n}
 \end{aligned}$$

Untuk melakukan enkripsi pesan $m < p$, dilakukan dengan cara berikut

1. Pilih bilangan bulat positif kurang dari $n - 1$
2. Hasil enkripsinya adalah $g^{mh} \pmod{n}$

Untuk melakukan dekripsi dari pesan c dilakukan dengan cara berikut

1. Hitung $a = L(c^{p-1} \pmod{p^2})$ dengan $L(x) = (x - 1)/p$
2. Hitung $b = L(g^{p-1} \pmod{p^2})$
3. Hitung $b^{-1} \pmod{p}$ dengan Extended Euclidean Algorithm
4. Hasil dekripsinya adalah $ab^{-1} \pmod{p}$

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

Pada subbab ini akan dijelaskan tentang struktur data, fungsi, dan prosedur yang dibangun pada aplikasi.

4.1.1 Struktur Data, Fungsi, dan Prosedur Controller GUI

Struktur data pada controller GUI berisi komponen-komponen antarmuka yang terlihat pada aplikasi, contohnya Button, TextBox, dan PictureBox.

```
Windows Form Designer generated code
private System.Windows.Forms.Button imageInput;
private System.Windows.Forms.OpenFileDialog openFileDialog1;
private System.Windows.Forms.TextBox TitleText;
private System.Windows.Forms.Button SearchButton;
private System.Windows.Forms.PictureBox OutputImage;
private System.Windows.Forms.RichTextBox richTextBox1;
private System.Windows.Forms.PictureBox InputImage;
private System.Windows.Forms.Button AlgToggle;
private System.Windows.Forms.RichTextBox SearchResultText;
private System.Windows.Forms.RichTextBox YourImageText;
private System.Windows.Forms.RichTextBox PerfCounterText;
private System.Windows.Forms.BindingSource form1BindingSource;
private System.Windows.Forms.RichTextBox InputErrorMsg;
```

Fungsi dan prosedur yang didefinisikan pada controller GUI yakni fungsi dan prosedur yang mengatur alur jalannya program dan alur interaksi pengguna dengan komponen antarmuka. Contohnya, `AlgToggle_Click` yang mengatur perilaku tombol *toggle* algoritme.

```
private void AlgToggle_Click(object sender, EventArgs e)
{
    if (this.alg == "kmp")
    {
        AlgToggle.Text = "Current Algorithm: BM\n(Click to toggle)";
        this.alg = "bm";
    }
    else
    {
        AlgToggle.Text = "Current Algorithm: KMP\n(Click to toggle)";
        this.alg = "kmp";
    }
}
```

Berikut adalah daftar fungsi dan prosedur pada controller halaman utama aplikasi secara lengkap dalam bentuk tabel.

<i>Signature</i> fungsi/prosedur	Kegunaan
<code>private async void init_stuff()</code>	Menampilkan loading screen dan menunggu pemuatan data sidik jari, kemudian menginisialisasi form halaman utama
<code>private void init_other_stuff(LoadingScreen ls)</code>	Memuat data sidik jari dan meng- <i>update</i> progress <i>loading screen</i> .
<code>private void ImageInputButton_Click(object sender, EventArgs e)</code>	Berisi pengaturan perilaku tombol input image ketika diklik
<code>private void SearchButton_Click(object sender, EventArgs e)</code>	Berisi pengaturan perilaku tombol search ketika diklik
<code>private void AlgToggle_Click(object sender, EventArgs e)</code>	Berisi pengaturan perilaku tombol toggle algoritme ketika diklik

4.1.2 Struktur Data, Fungsi, dan Prosedur Algoritme

Pada file Algoritme.cs, terdapat semua fungsi yang berkaitan dengan pemrosesan string, binary, ascii, dan regular expression. Berikut daftarnya dalam bentuk tabel.

<i>Signature</i> fungsi/prosedur	Kegunaan
<code>public static string bmpToBinary(string imagePath)</code>	Mengonversi isi file bmp menjadi string binary
<code>public static string binaryToAscii(string binaryString)</code>	Mengonversi string binary menjadi string ascii
<code>public static string get_middle_binary(string binaryString)</code>	Mendapatkan 64 karakter tengah dari string binary
<code>public static int[] badCharHeuristic(string str, int size)</code>	Mengonstruksi dan mengembalikan tabel <i>last occurrence</i> dari sebuah string

<code>public static bool bmSearch(string text, string pattern)</code>	Mencari string pattern pada string text menggunakan algoritme Boyer Moore
<code>public static int[] ComputeLPS(string pattern)</code>	Mengonstruksi dan mengembalikan LPS (<i>Longest Proper Suffix</i>) array
<code>public static bool kmpSearch(string text, string pattern)</code>	Mencari string pattern pada string text menggunakan algoritme Knuth Morris Pratt
<code>public static int longestCommonSS(string text1, string text2)</code>	Mengembalikan <i>longest common substring</i> dari dua string
<code>public static string getMiddle16(string text)</code>	Mengembalikan 64 karakter tengah dari sebuah string
<code>public static float calculateSimilarity(string text1, string text2)</code>	Menghitung similaritas dua string berdasarkan <i>longest common substring</i> -nya
<code>public static string replaceNumbersWithChars(string input)</code>	Menggantikan angka pada string menjadi karakter yang sesuai
<code>public static string convertToTitleCase(string input)</code>	Mengonversi string menjadi <i>title case</i> (Contohnya Seperti Ini)
<code>public static string addOptionalVowel(string input)</code>	Menambahkan huruf vokal yang opsional pada string
<code>public static string convertToRegexPattern(string input)</code>	Mengonversi string menjadi pola regular expression
<code>public static bool isMatch(string input, string pattern)</code>	Mengecek apakah sebuah string dapat diterima sebuah pola regular expression

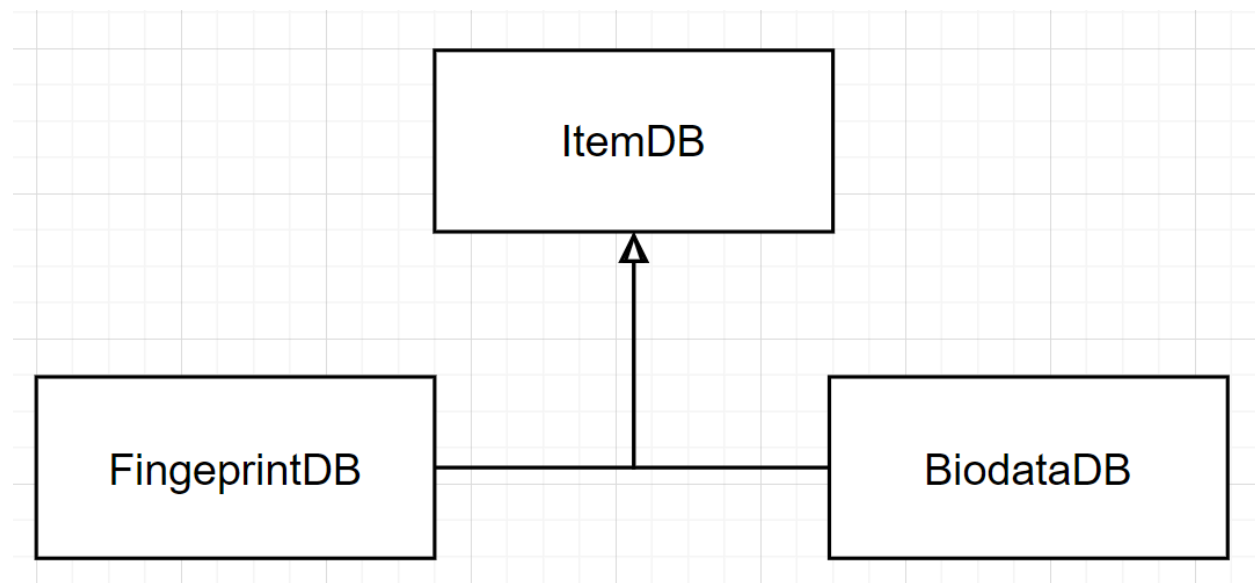
4.1.3 Struktur Data, Fungsi, dan Prosedur Interaksi Basis Data

Dalam interaksi dengan Basis Data MySQL digunakan library MySQL yang penggunaannya terdapat pada file ItemDB.cs, FingerPrintDB.cs, dan BiodataDB.cs, berikut penjelasan mengenai kelas yang digunakan.

Nama Kelas	Kegunaan
------------	----------

ItemDB	Membuat dan menyimpan koneksi ke basis data
BiodataDB : ItemDB	Kelas yang memberikan fungsionalitas untuk berinteraksi dengan tabel biodata
FingerprintDB : ItemDB	Kelas yang memberikan fungsionalitas untuk berinteraksi dengan tabel sidik_jari

Untuk memperjelas, hubungan ketiga kelas ini dapat digambarkan dengan diagram UML sebagai berikut dengan panah berarti hubungan IsA



Pada ItemDB sendiri terdapat fungsi dan prosedur sebagai berikut

<i>Signature</i> fungsi/prosedur	Kegunaan
protected static void Initialize()	Menginisialisasi koneksi dengan basis data berupa host, port, database, dan password user yang digunakan
protected static bool OpenConnection()	Digunakan ketika ingin mengakses basis data
protected static bool CloseConnection()	Digunakan ketika telah selesai mengakses basis data

Pada BiodataDB terdapat fungsi dan prosedur sebagai berikut

<i>Signature</i> fungsi/prosedur	Kegunaan
<code>public static void Insert(Biodata b)</code>	Mengenkripsi Biodata b dan menjalankan query untuk menambahkan data biodata tersebut ke basis data
<code>public static Biodata Find(string nama)</code>	Mengembalikan objek biodata yang memiliki nama tertentu pada basis data
<code>public static List<Biodata> All()</code>	Membaca seluruh data biodata dari basis data dan mendekripsinya kemudian mengembalikannya dalam bentuk objek Biodata
<code>public static List<string> AllNames()</code>	Membaca seluruh data biodata dari basis data dan mendekripsinya kemudian mengembalikan list nama yang ada pada Biodata
<code>public static void Clear()</code>	Mengosongkan Tabel Biodata pada basis data

Pada FingerprintDB terdapat fungsi dan prosedur sebagai berikut

<i>Signature</i> fungsi/prosedur	Kegunaan
<code>public static void Insert(string berkas_citra, string nama)</code>	Mengenkripsi berkas_citra dan nama kemudian menjalankan query untuk menambahkan data tersebut ke basis data
<code>public static String Find(string berkas_citra)</code>	Melakukan pencarian nama yang teridentifikasi oleh suatu berkas citra pada basis data
<code>public static Dictionary<string, string> All()</code>	Mengembalikan seluruh data yang ada pada tabel sidik_jari yang telah didekripsi
<code>public static void Clear()</code>	Mengosongkan Tabel sidik_jari pada basis data

4.1.4 Struktur Data, Fungsi, dan Prosedur Enkripsi

Pada file Encryption.cs, terdapat semua fungsi yang berkaitan dengan enkripsi dan dekripsi data menggunakan sistem Okamoto-Uchiyama.

<i>Signature</i> fungsi/prosedur	Kegunaan
----------------------------------	----------

<code>public Tuple<BigInteger, BigInteger, BigInteger> getPublicKey()</code>	Mengembalikan Tuple(n, g, h) yaitu public key dari skema enkripsi ini
<code>public BigInteger Encrypt(BigInteger m)</code>	Mengembalikan hasil enkripsi untuk sebuah BigInteger m dengan ukuran yang relatif kecil dalam bentuk BigInteger
<code>public byte[] EncryptBytes(byte[] bytes)</code>	Mengembalikan hasil enkripsi untuk sebuah array of bytes dengan ukuran relatif kecil dalam bentuk array of bytes
<code>public byte[] EncryptLongBytes(byte[] bytes)</code>	Mengembalikan hasil enkripsi untuk sebuah array of bytes tanpa terbatas ukuran dalam bentuk array of bytes
<code>public BigInteger Decrypt(BigInteger c)</code>	Mengembalikan hasil dekripsi untuk sebuah BigInteger c dengan ukuran yang relatif kecil dalam bentuk BigInteger
<code>public byte[] DecryptBytes(byte[] bytes)</code>	Mengembalikan hasil dekripsi untuk sebuah array of bytes dengan ukuran relatif kecil dalam bentuk array of bytes
<code>public byte[] DecryptLongBytes(byte[] bytes)</code>	Mengembalikan hasil dekripsi untuk sebuah array of bytes tanpa terbatas ukuran dalam bentuk array of bytes
<code>public String EncryptString(String m)</code>	Mengembalikan string hasil enkripsi sebuah message m
<code>public String DecryptString(String m)</code>	Mengembalikan string hasil dekripsi sebuah code c
<code>public String EncryptDateTime(DateTime d)</code>	Mengembalikan string hasil enkripsi sebuah DateTime d
<code>public DateTime DecryptDateTime(String c)</code>	Mengembalikan DateTime hasil dekripsi sebuah String c

4.2 Penjelasan Tata Cara Penggunaan Program

Berikut adalah langkah-langkah penggunaan program.

1. Jalankan program dan tunggu hingga *loading* selesai
2. Pada halaman utama, klik tombol “*Click Me!*” untuk memasukkan foto.

3. Akan muncul prompt untuk memilih foto. Anda hanya bisa memilih file dalam format bmp.
4. Anda bisa memilih algoritme yang ingin dipakai (*default*-nya KMP).
5. Klik tombol search dan Anda akan mendapatkan hasil pencarian.
6. Jika tidak ada gambar dengan kemiripan di atas 60%, tidak akan dimunculkan hasil apa-apa selain keterangan tersebut.
7. Waktu eksekusi dapat dilihat di kanan bawah.
8. Anda bisa melakukan pencarian lagi, cukup ulangi langkah-langkah di atas (tidak perlu menjalankan ulang program).

4.3 Hasil Pengujian

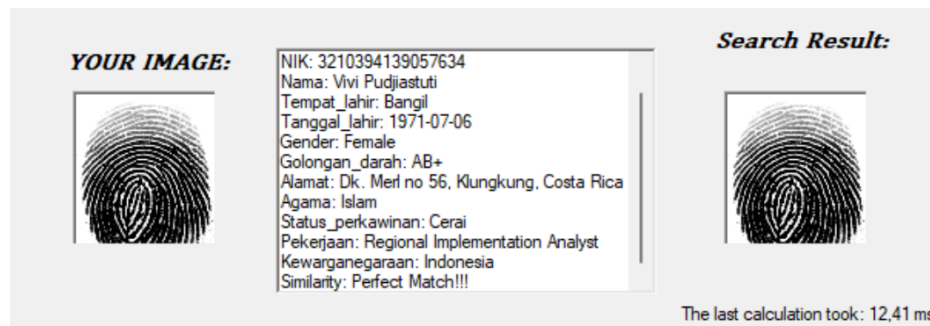
Berikut adalah hasil pengujian beberapa kasus, dimana data sidik jari di basis data adalah gambar sidik jari real.

Skenario input gambar real :

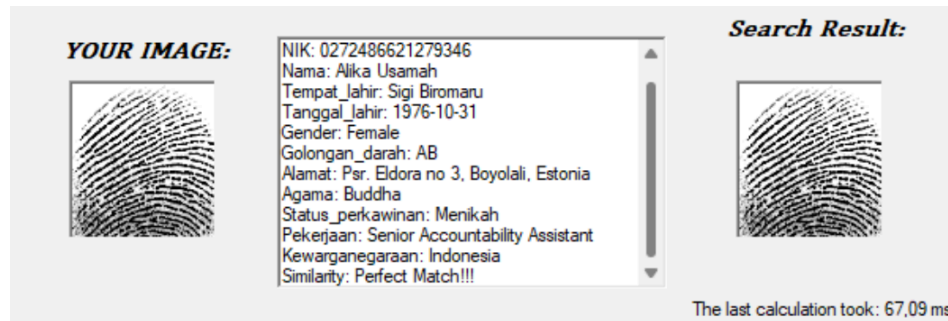
KMP



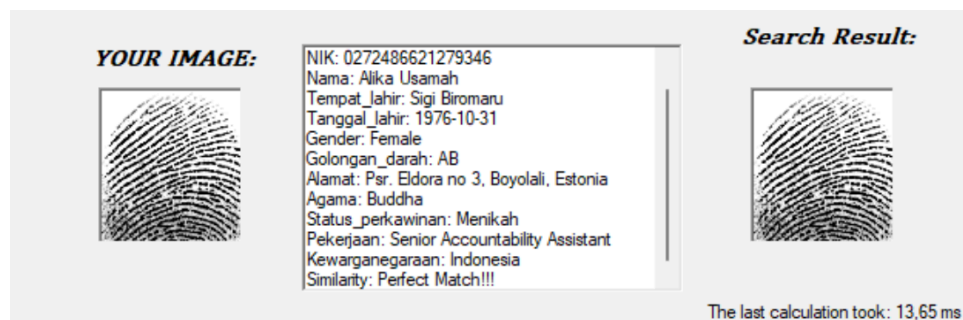
BM :



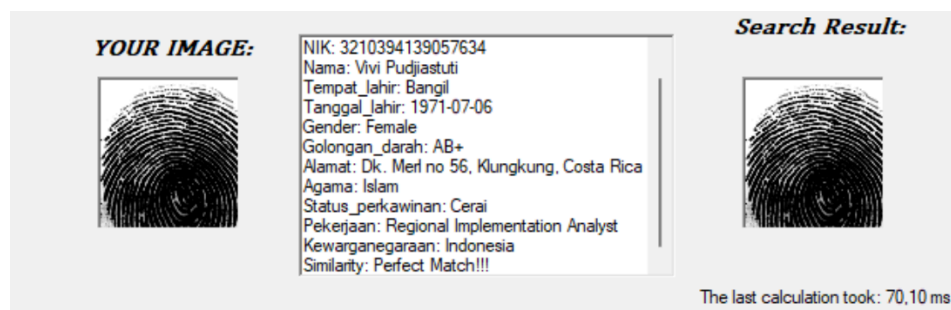
KMP :



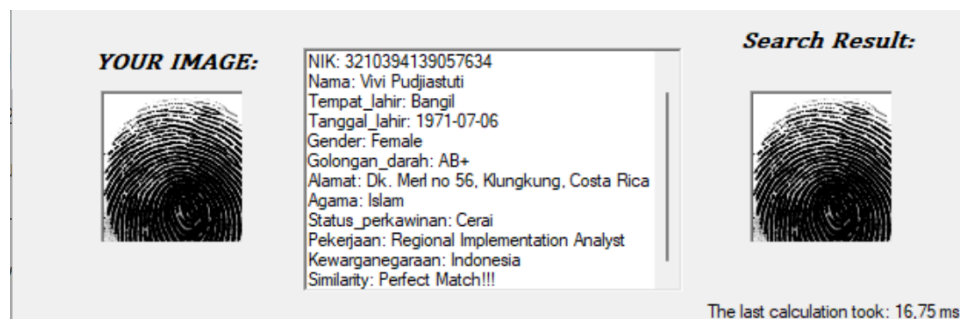
BM :



KMP :




BM :




Dapat terlihat bahwa dengan menggunakan dataset real, pencarian dengan KMP dan BM menghasilkan hasil yang sama, dengan pencarian dengan BM menghasilkan solusi lebih cepat.

Skenario input gambar altered-easy :

KMP :

YOUR IMAGE:


NIK: 3210394139057634
Nama: Vivi Pudjiastuti
Tempat_lahir: Bangil
Tanggal_lahir: 1971-07-06
Gender: Female
Golongan_darah: AB+
Alamat: Dk. Merl no 56, Klungkung, Costa Rica
Agama: Islam
Status_perkawinan: Cerai
Pekerjaan: Regional Implementation Analyst
Kewarganegaraan: Indonesia
Similarity: 91,00%

Search Result:


The last calculation took: 258,83 ms

BM :

YOUR IMAGE:


NIK: 3210394139057634
Nama: Vivi Pudjiastuti
Tempat_lahir: Bangil
Tanggal_lahir: 1971-07-06
Gender: Female
Golongan_darah: AB+
Alamat: Dk. Merl no 56, Klungkung, Costa Rica
Agama: Islam
Status_perkawinan: Cerai
Pekerjaan: Regional Implementation Analyst
Kewarganegaraan: Indonesia
Similarity: 91,00%

Search Result:


The last calculation took: 207,65 ms

KMP :

YOUR IMAGE:


NIK: 8657592619068910
Nama: Lega Setiawan
Tempat_lahir: Anyer
Tanggal_lahir: 1975-01-10
Gender: Male
Golongan_darah: A
Alamat: Jln. Rocio no 9, Kepanjen, Senegal
Agama: Islam
Status_perkawinan: Belum Menikah
Pekerjaan: Dynamic Branding Producer
Kewarganegaraan: Indonesia
Similarity: 92,00%


Search Result:


The last calculation took: 257,73 ms

BM :

YOUR IMAGE:



NIK: 8657592619068910
Nama: Lega Setiawan
Tempat_lahir: Anyer
Tanggal_lahir: 1975-01-10
Gender: Male
Golongan_darah: A
Alamat: Jln. Rocio no 9, Kepanjen, Senegal
Agama: Islam
Status_perkawinan: Belum Menikah
Pekerjaan: Dynamic Branding Producer
Kewarganegaraan: Indonesia
Similarity: 92,00%

Search Result:



The last calculation took: 203,79 ms

Skenario input gambar altered medium :

KMP :

YOUR IMAGE:


NIK: 3210394139057634
Nama: Vivi Pudjiastuti
Tempat_lahir: Bangil
Tanggal_lahir: 1971-07-06
Gender: Female
Golongan_darah: AB+
Alamat: Dk. Merl no 56, Klungkung, Costa Rica
Agama: Islam
Status_perkawinan: Cerai
Pekerjaan: Regional Implementation Analyst
Kewarganegaraan: Indonesia
Similarity: 76,00%

Search Result:

The last calculation took: 202,47 ms

BM :


YOUR IMAGE:


NIK: 3210394139057634
Nama: Vivi Pudjiastuti
Tempat_lahir: Bangil
Tanggal_lahir: 1971-07-06
Gender: Female
Golongan_darah: AB+
Alamat: Dk. Merl no 56, Klungkung, Costa Rica
Agama: Islam
Status_perkawinan: Cerai
Pekerjaan: Regional Implementation Analyst
Kewarganegaraan: Indonesia
Similarity: 76,00%


Search Result:

The last calculation took: 200,87 ms


KMP :

YOUR IMAGE:



Nama: Jaka Narpati
Tempat_lahir: Masamba
Tanggal_lahir: 1965-01-24
Gender: Male
Golongan_darah: A+
Alamat: Gg. Rogahn no 9, Cikampek,
Greenland
Agama: Buddha
Status_perkawinan: Cerai
Pekerjaan: Internal Accountability Liaison
Kewarganegaraan: Indonesia
Similarity: 86,00%

Search Result:

The last calculation took: 253,87 ms

BM :


YOUR IMAGE:


Nama: Jaka Narpati
Tempat_lahir: Masamba
Tanggal_lahir: 1965-01-24
Gender: Male
Golongan_darah: A+
Alamat: Gg. Rogahn no 9, Cikampek,
Greenland
Agama: Buddha
Status_perkawinan: Cerai
Pekerjaan: Internal Accountability Liaison
Kewarganegaraan: Indonesia
Similarity: 86,00%


Search Result:

The last calculation took: 200,98 ms

Skenario input gambar altered-hard :

KMP :


YOUR IMAGE:


Nama: Jarwadi Mandala
Tempat_Jahir: Sleman
Tanggal_Jahir: 2004-01-12
Gender: Male
Golongan_darah: B+
Alamat: Jr. Alexandrine no 89, Subang, Ghana
Agama: Konghucu
Status_perkawinan: Belum Menikah
Pekerjaan: Corporate Communications Assistant
Kewarganegaraan: Indonesia
Similarity: 74,00%


Search Result:


The last calculation took: 260,97 ms

BM :


YOUR IMAGE:


Nama: Jarwadi Mandala
Tempat_Jahir: Sleman
Tanggal_Jahir: 2004-01-12
Gender: Male
Golongan_darah: B+
Alamat: Jr. Alexandrine no 89, Subang, Ghana
Agama: Konghucu
Status_perkawinan: Belum Menikah
Pekerjaan: Corporate Communications Assistant
Kewarganegaraan: Indonesia
Similarity: 74,00%


Search Result:


The last calculation took: 207,43 ms

KMP :

YOUR IMAGE:


Most similar find:
NIK: 9237307665787957
Nama: Jarwa Marbun
Tempat_Jahir: Batang
Tanggal_Jahir: 1967-03-09
Gender: Male
Golongan_darah: B+
Alamat: Jln. Borer no 6, Sidenreng, Eritrea
Agama: Hindu
Status_perkawinan: Belum Menikah
Pekerjaan: Forward Factors Officer
Kewarganegaraan: Indonesia

Search Result:


The last calculation took: 269,84 ms

BM :

YOUR IMAGE:


NIK: 9237307665787957
Nama: Jarwa Marbun
Tempat_Jahir: Batang
Tanggal_Jahir: 1967-03-09
Gender: Male
Golongan_darah: B+
Alamat: Jln. Borer no 6, Sidenreng, Eritrea
Agama: Hindu
Status_perkawinan: Belum Menikah
Pekerjaan: Forward Factors Officer
Kewarganegaraan: Indonesia
Similarity: 71,00%

Search Result:


The last calculation took: 199,35 ms

Terlihat bahwa ketika gambar input merupakan gambar yang telah ter-alter, maka similarity yang ditampilkan berupa persen, bukan merupakan exact match. Hal ini karena jika gambar telah ter-alter, maka algoritme pertama akan mencari sidik jari yang exact match menggunakan

KMP/BM, jika tidak ditemukan maka akan dilakukan pencarian menggunakan LCS dan menghitung tingkat kemiripannya. Dalam hasil uji ini, baik input gambar real, altered-easy, altered-medium, atau altered-hard, seluruhnya menampilkan hasil sidik jari yang sesuai, tidak ada yang miss.

4.4 Analisis Hasil Pengujian

Dari hasil pengujian kami, didapat bahwa algoritme Boyer Moore dan Knuth-Morris Pratt sama-sama menghasilkan hasil yang akurat, tetapi algoritme BM selalu lebih cepat daripada algoritme KMP.

Algoritme BM dan KMP menghasilkan hasil yang akurat karena kedua algoritme tersebut memiliki tujuan yang sama, yaitu mencari pattern (teks yang lebih pendek) pada teks (teks yang lebih panjang) secara exact match, yang berarti seluruh pattern harus berada pada teks tanpa ada huruf yang berbeda. Dan dalam kasus input gambar real, pattern yang didapat pasti terdapat pada basis data dikarenakan gambar sidik jari yang dimasukkan kedalam basis data juga merupakan gambar-gambar dari sidik jari real.

Dalam mendapatkan pattern dari sidik jari input pengguna, kami menggunakan pattern berukuran 64 pixel, atau 64 digit binary, atau 8 digit ASCII yang berada di tengah-tengah sidik jari. Pemilihan pattern berukuran 64 pixel tersebut karena dari pengamatan kami, ukuran 64 pixel adalah ukuran yang pas, dimana ukuran pattern tersebut tidak terlalu kecil dan tidak terlalu besar (64 pixel pattern vs 10000 pixel teks).

Algoritme BM lebih cepat daripada KMP karena dalam algoritme BM, terdapat tabel Bad Character Heuristic (atau last occurrence index) yang dimana tabel ini memungkinkan lompatan yang jauh ketika melakukan perbandingan. Pada umumnya, algoritme BM baik digunakan jika ukuran alphabetnya besar, seperti contohnya alphabet inggris, dalam kasus ini bahkan ukuran alphabetnya jauh lebih besar daripada alphabet inggris karena terdapat 256 ASCII character. Sedangkan, algoritme KMP baik digunakan dalam perbandingan karakter dengan jumlah alphabet kecil, seperti binary ataupun perbandingan rantai DNA.

Algoritme BM memiliki kompleksitas waktu rata-rata $O(N + M)$ dimana N adalah ukuran teks dan M adalah ukuran pattern, untuk kasus terburuk, algoritme ini memiliki kompleksitas $O(MN + A)$, dimana A adalah ukuran alphabet, namun kasus terburuk ini sangat jarang terjadi, dan algoritme ini mempunyai kompleksitas ruang $O(A)$ untuk menyimpan tabel last occurrence. algoritme KMP memiliki kompleksitas waktu rata-rata $O(N + M)$ dan kompleksitas ruang $O(M)$ untuk menyimpan tabel fungsi pinggiran.

Di kasus gambar input bukan sidik jari real, algoritme KMP dan BM tidak dapat menemukan pattern yang exact match dengan teks, oleh karena itu digunakan algoritme Longest Common Subsequence (LCS) untuk menemukan gambar sidik jari paling mirip di basis data, untuk prosesnya sendiri sudah dijelaskan di bab 3.1. LCS memiliki kompleksitas waktu $O(NM)$, lebih lama daripada KMP ataupun BM. Oleh karena itu kami menggunakan algoritme ini hanya ketika tidak menemukan sidik jari yang exact match.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kami telah membuat sebuah perangkat lunak berupa aplikasi desktop dengan fitur sesuai pada subbab 3.3. Kami telah mengimplementasikan algoritme KMP dan BM untuk pencarian sidik jari yang exact match dan algoritme LCS untuk mencari sidik jari dengan tingkat kesamaan tertinggi pada proses pencocokan sidik jari.

5.2 Saran

Saran yang bisa kami berikan untuk pelaksanaan tugas besar ini adalah sebagai berikut.

1. Saran untuk pengerjaan tugas besar. Sebaiknya dalam pengerjaan tugas besar ini, mahasiswa memahami konsep algoritme *string matching* dengan baik, tidak hanya meng-copy paste kode dari internet.
2. Saran untuk desain tugas besar. Desain tugas besar ini sudah bagus, tetapi menurut kami ada baiknya jika implementasi *string matching* lebih dibebaskan dan tidak hanya terpaku pada algoritme yang diajarkan pada salindia kuliah (tetap menjadi spek wajib, tetapi mahasiswa dianjurkan untuk mengeksplorasi teknik dan algoritme lain).
3. Saran untuk penulisan spesifikasi tugas besar. Gunakan bentuk baku dari **algoritma**, yaitu **algoritme** (kecuali dalam penulisan judul karena merupakan unsur nama).

LAMPIRAN

Link Repository github : https://github.com/abdulrafiqh/Tubes3_TheMan

Link Video : <https://youtu.be/-oUwLTH1UyY?si=b0zMSk10ZfFTseNq>

DAFTAR PUSTAKA

- Munir, Rinaldi. 2024. "Pencocokan String (String/Pattern Matching)".
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>
- Munir, Rinaldi. 2024. "String Matching dengan Regular Expression".
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2022-2023/String-Matching-dengan-Regex-2019.pdf>
- Munir, Rinaldi. 2024. "Kriptografi Kunci Publik".
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2023-2024/17-Kriptografi-Kunci-Publik-2024.pdf>
- Knuth, D. E., Morris, J. H., & Pratt, V. R. (1977). "Fast pattern matching in strings." *SIAM Journal on Computing*, 6(2), 323-350. <https://doi.org/10.1137/0206024>
- Boyer, R. S., & Moore, J. S. (1977). "A fast string searching algorithm." *Communications of the ACM*, 20(10), 762-772. <https://doi.org/10.1145/359842.359859>
- Buchanan, Bill (2019). "Okamoto-Uchiyama Crypto" Medium.com
<https://medium.com/asecuritysite-when-bob-met-alice/okamoto-uchiyama-crypto-6bd7181fa5b3>