



**IMPLEMENTASI TEKNIK KOMPRESI DENGAN  
ALGORITMA SEQUITUR PADA PESAN TEKS**

Disusun dan Diajukan untuk Memenuhi Persyaratan Ujian Akhir Memperoleh  
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi  
Universitas Pembangunan Panca Budi  
Medan

**SKRIPSI**

OLEH:

NAMA : RAGIL ADITVA PRAYUGO  
NPM : 1614370169  
PROGRAM STUDI : SISTEM KOMPUTER

FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS PEMBANGUNAN PANCA BUDI  
MEDAN  
2020

LEMBAR PENGESAHAN

IMPLEMENTASI TEKNIK KOMPRESI DENGAN ALGORITMA  
SEQUITUR PADA PESAN TEKS

Dibuat Oleh:

NAMA : BAGIL ADITYA PRAYUGO  
NPM : 1614570169  
PROGRAM STUDI : SISTEM KOMPUTER

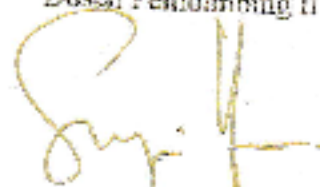
Skripsi Telah Ditetapkan oleh Dosen Pembimbing Skripsi  
pada Tanggal :

Dosen Pembimbing I



Dian Kurnia, S.Kom., M.Kom.

Dosen Pembimbing II



Supiyandi, S.Kom., M.Kom.

Mengetahui:

Dekan Fakultas Sains dan Teknologi



Hardean, S.T., M.T.

Ketua Program Studi Sistem Komputer

Eko Hariyanto, S.Kom., M.Kom.





**YAYASAN PROF. DR. H. KADIRUN YAHYA**  
**PERPUSTAKAAN UNIVERSITAS PEMBANGUNAN PANCA BUDI**  
Jl. Jend. Gatot Subroto KM. 4,5 Medan Sunggal, Kota Medan Kode Pos 20122

**SURAT BEBAS PUSTAKA**  
**NOMOR: 2355/PERP/BP/2020**

Perpustakaan Universitas Pembangunan Panca Budi menerangkan bahwa berdasarkan data pengguna perpustakaan nama saudara/i:

Nama : RAGIL ADITYA PRAYUGO  
NIM : 1614370169  
Kelas/Semester : Akhir  
Jurusan : SAINS & TEKNOLOGI  
Jurusan/Prodi : Sistem Komputer

namanya terhitung sejak tanggal 06 Juli 2020, dinyatakan tidak memiliki tanggungan dan atau pinjaman buku sekaligus tidak terdaftar sebagai anggota Perpustakaan Universitas Pembangunan Panca Budi Medan.

Medan, 06 Juli 2020  
Diketahui oleh,  
Kepala Perpustakaan,

  
Sugiarjo S. Sos. S.Pd.I



**KARTU BEBAS PRAKTIKUM**  
**Nomor. 1262/BL/LAKO/2020**

Perantara tangan dibawah ini Ka. Laboratorium Komputer dengan ini menerangkan bahwa :

Nama : RAGIL ADITYA PRAYUGO  
NIM : 1614370189  
Kelas/Semester : Akhir  
Jurusan : SAINS & TEKNOLOGI  
Konsentrasi/Prodi : Sistem Komputer

Adapun telah menyelesaikan urusan administrasi di Laboratorium Komputer Universitas Pembangunan Panca Budi Medan.

Medan, 01 Oktober 2020  
Ka. Laboratorium

  
Fachrud Wadly, S. Kom., M. Kom.





YAYASAN PROF. DR. H. KADRUN YAHYA  
UNIVERSITAS PEMBANGUNAN PANCA BUDI

Jl. Jend. Gatot Subroto KM 4.5 PO. BOX 1099 Telp. 061-30105057 Fax. (061) 451-4808  
MEDAN - INDONESIA  
Website : www.pancabudi.ac.id - Email : admin@pancabudi.ac.id

### LEMBAR BUKTI BIMBINGAN SKRIPSI

Mahasiswa : RAGIL ADITYA PRAYUGO  
NPM : 1814370169  
Jurusan Studi : Sistem Komputer  
Pendidikan : Strata Satu  
Pembimbing : Dian Kurnia, S.Kom., M.Kom.  
Materi : Implementasi Teknik Kompresi Dengan Algoritma Sequitur Pada Pesan Teks

Tanggal	Pembahasan Materi
2020	Perbaiki tata letak gambar UML dan activity diagram pada BAB 3, beberapa tujuan yang salah pada BAB 1, Lanjut Bab 4
2020	Perbaiki bab 3 pada rancangan penelitian sesuaikan dengan panduan skripsi dan susunan sub bab 4 sesuaikan dengan panduan skripsi
2020	Lengkapi berkas Seminar Hasil, ACC Seminar Hasil
2020	Lengkapi berkas sidang meja hijau, ACC Sidang
Desember 2020	Lengkapi berkas Jلد Lux, ACC Jلد

Medan, 01 Oktober 2020  
Dosen Pembimbing

Dian Kurnia, S.Kom., M.Kom



YAYASAN PROF. DR. H. KADIRUN YAHYA  
UNIVERSITAS PEMBANGUNAN PANCA BUDI

Jl. Jend. Gatot Subroto KM 4,5 P.O. BOX 1099 Telp. 061-30108057 Fax. (061) 4514808  
MEDAN - INDONESIA  
Website : [www.pancabudi.ac.id](http://www.pancabudi.ac.id) - Email : [admin@pancabudi.ac.id](mailto:admin@pancabudi.ac.id)

### LEMBAR BUKTI BIMBINGAN SKRIPSI

Nama : RAGIL ADITYA PRAYUGO  
NIM : 1614370169  
Jurusan : Sistem Komputer  
Pendidikan : Sarjana Satu  
Pembimbing : Supiyandi, S.Kom., M.Kom  
Judul : Inkrementasi Teknik Kompresi Dengan Algoritma Heuristik Pada Pesan Teks0

Tanggal	Pembahasan Materi
	- Pada BAB I satu susuaikan Bahasa Asing dengan mencentak Miring Teks nya - Pada BAB II beri no urut pada persamaan di rumus , apabil persamaan.
	Revisi BAB III , Berikan penjelasan untuk knangka Peristiwa yang memili Fase 1 s.d Fase 4 , pada tabel
	Perbaiki bab 3 pada rangkangan penelitian sesuaikan dengan panduan skripsi dan susunan sub bab 4 sesuaikan dengan panduan skripsi
	ACC Seminar Hasil
	Lengkapi berkas sidang mois ijel, ACC Sidang
10/10/2020	ACC JIEd

Medan, 01 Oktober 2020  
Dosen Pembimbing

Supiyandi, S.Kom., M.Kom



# UNIVERSITAS PEMBANGUNAN PANCA BUDI FAKULTAS SAINS & TEKNOLOGI

Jl. Jend. Gatot Subroto Km 4,5 Medan Fax. 061 8458077 PO. BOX : 1099 MEDAN

PROGRAM STUDI TEKNIK ELEKTRO	(TERAKREDITASI)
PROGRAM STUDI ARSITEKTUR	(TERAKREDITASI)
PROGRAM STUDI SISTEM KOMPUTER	(TERAKREDITASI)
PROGRAM STUDI TEKNIK KOMPUTER	(TERAKREDITASI)
PROGRAM STUDI AGROTEKNOLOGI	(TERAKREDITASI)
PROGRAM STUDI PETERNAKAN	(TERAKREDITASI)

## PERMOHONAN JUDUL TESIS / SKRIPSI / TUGAS AKHIR\*

Saya yang bertanda tangan di bawah ini :

Nama Lengkap	: RAGIL ADITYA PRAYUGO
Tempat/Tgl. Lahir	: TANJUNGPINANG - 30 Juli 1998
Nomor Pokok Mahasiswa	: 1614320164
Program Studi	: Sistem Komputer
Konentrasi	: Rekayasa Perangkat Lunak
Jumlah Kredit yang telah dicapai	: 135 SKS - IPK 2.10
Nomor Hp	: 087361444421

Dengan ini mengajukan judul sesuai bidang ilmu sebagai berikut

No.	Judul
1.	Implementasi Teknik Kompresi Dengan Algoritma LZW pada Pesan Teks

Catatan : Disetujui Dosen Pembimbing I dan Pembimbing II

\*Cara Yang Tidak

Sherry Pramono, SE., MM

Medan, 02 Mei 2020  
  
 (Ragil Aditya Prayugo)

Tanggal _____ Disetujui oleh Dosen Pembimbing I  (Handi S. MT)	Tanggal _____ Disetujui oleh Ka. Prodi Sistem Komputer  (Handi S. MT)
--	---

Tanggal _____ Disetujui oleh Dosen Pembimbing I  (Handi S. MT)	Tanggal _____ Disetujui oleh Dosen Pembimbing II  (Handi S. MT)
--	---

## SURAT KETERANGAN PLAGIAT CHECKER

Dengan ini saya Ke.LPMU UNPAB menerangkan bahwa surat ini adalah bukti pengesahan dari LPMU sebagai pengesah proses plagiat checker Tugas Akhir/ Skripsi/Tesis selama masa pandemi *Covid-19* sesuai dengan edaran rektor Nomor : 7594/13/R/2020 Tentang Pemberitahuan Perpanjangan PBM Online.

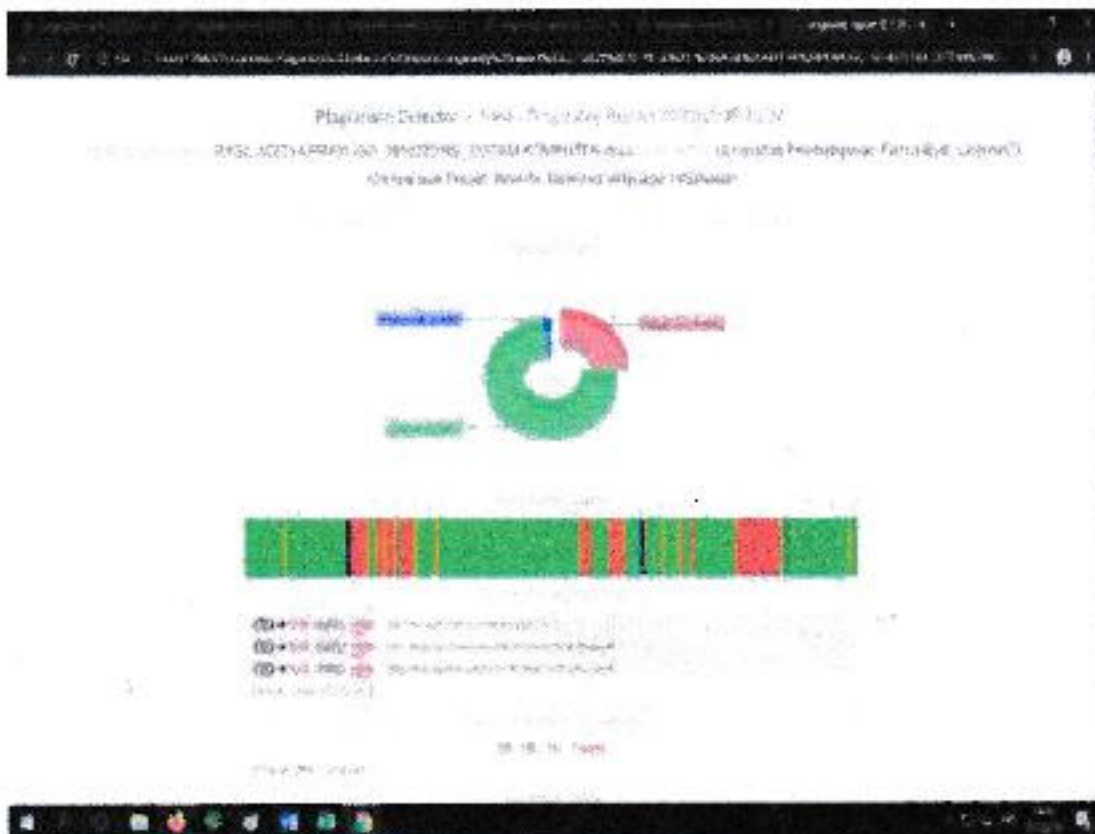
Demikian disampaikan.

NB: Segala pernyalagunaan/pelanggaran atas surat ini akan di proses sesuai ketentuan yang berlaku UNPAB.

Ke.LPMU



Cahyo Pranono, SE.,MM





Hal: Permohonan Meja Hijau

Medan, 25 Juli 2020  
 Kepada Yth: Bapak/Ibu Dekan  
 Fakultas SAINS & TEKNOLOGI  
 UINPA Medan  
 Di -  
 Tempat

Dengan hormat, saya yang bertanda tangan di bawah ini:

Nama : RAGEL ADITYA PRAYUGO  
 Tempat/Tgl. Lahir : TANJUNG ANOM - 10 Juni 1998  
 Nama Orang Tua : SUWARNO  
 N. P. M : 1614370149  
 Fakultas : SAINS & TEKNOLOGI  
 Program Studi : Sistem Komputer  
 No. HP : 08136144422  
 Alamat : Jalan Besar Tanjung Anom No 214

Datang bermohon kepada Bapak/Ibu untuk dapat diterima mengikuti Ujian Meja Hijau dengan judul Implementasi Teknik Kompresi Dengan Algoritma Sequitur Pada Pesan Teks. Selanjutnya saya menyatakan:

1. Melampirkan KKW yang telah disahkan oleh Ka. Prodi dan Dekan
2. Tidak akan menuntut ujian perbaikan nilai/mata kuliah untuk perbaikan indeks prestasi (IP), dan mohon diterbitkan ijazahnya setelah lulus ujian meja hijau.
3. Telah tercapai keterangan bebas jastaka
4. Tertampirl surat keterangan bebas laboratorium
5. Tertampirl pas photo untuk Ijazah ukuran 4x6 = 5 lembar dan 3x4 = 5 lembar Hitam Putih
6. Tertampirl foto copy STTB SUTA dilegalisir 1 (satu) lembar dan bagi mahasiswa yang lanjutan D3 ke S1 lampirkan ijazah dan transkripnya sebanyak 1 lembar
7. Tertampirl pelunasan kwitansi pembayaran uang kuliah berkaki dan insuda sebanyak 1 lembar
8. Serpias sudah diproses lux 2 exemplar (1 untuk perpustakaan, 1 untuk mahasiswa) dan 3000 kertas jeruk 5 exemplar untuk penguji (bentuk dan warna penulisan diserahkan berdasarkan ketentuan fakultas yang berlaku) dan lembar persetujuan sudah di tandatangan dosen pembimbing, prodi dan dekan
9. Soft Copy Serpias disimpan di CD sebanyak 2 disc (sesuai dengan Judul Skripsinya)
10. Tertampirl surat keterangan BKROL (pada saat pengambilan Ijazah)
11. Setelah menyelesaikan persyaratan point-point diatas berkas di masukan ke dalam MAP
12. Bersedia menanggung biaya-biaya yang dibebankan untuk memproses pelaksanaan ujian dimaksud, dengan perhatian sbb:

1. [100] Ujian Meja Hijau	Rp.	0
2. [170] Administrasi Wisuda	Rp.	1.500.000
3. [202] Bebas Pustaka	Rp.	100.000
4. [221] Bebas LAB	Rp.	5.000
<b>Total Biaya</b>	<b>Rp.</b>	<b>1.605.000</b>

Periode Wisuda Ke : **65**

Ukuran Toga : **L**

Diketahui/Disetujui oleh:



HANDANI S. M.  
 Dekan Fakultas SAINS & TEKNOLOGI

Normal saya



RAGEL ADITYA PRAYUGO  
 1614370149

## SURAT PERNYATAAN

Saya Yang Bertanda Tangan Dibawah Ini :

Nama : RAGIL ADITYA PRAYUGO  
 N. P. M : 1814370169  
 Tempat/Tgl. Lahir : TANJUNG ANOM / 30 Juni 1998  
 Alamat : Jalan Besar Tanjung Anom No 214  
 No. HP : 081361444422  
 Nama Orang Tua : SUMARNO/EMFYATI  
 Fakultas : SAINS & TEKNOLOGI  
 Program Studi : Sistem Komputer  
 Judul : Implementasi Teknik Kompresi Dengan Algoritma Squirtur Pada Pesan Teks

Bersama dengan surat ini menyatakan dengan sebenar - benarnya bahwa data yang bertera diatas adalah sudah benar sesuai dengan ijazah pada pendidikan terakhir yang saya peroleh. Maka dengan ini saya tidak akan menyalahin pernyataan kepada UINPAD. Apabila ada kesalahan data pada ijazah saya.

Demiikianlah surat pernyataan ini saya buat dengan sebenar - benarnya, tanpa ada paksaan dari pihak manapun dan dibuat dalam keadaan sadar. Jika terjadi kesalahan, maka saya bersedia bertanggung jawab atas kesalahan saya.

Mendes, 15 Juli 2020

Saya Yang Bertanda Pernyataan

STAMP  
 000  
 RAGIL ADITYA PRAYUGO  
 1814370169

## SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Ragil Aditya Prayugo  
NPM : 1614370169  
Program Studi : Sistem Komputer  
Jenjang : S-1 (Strata Satu)  
Judul Skripsi : Implementasi Teknik Kompresi Dengan  
Algoritma Sequitur Pada Pesan Teks

Dengan ini menyatakan bahwa:

1. Skripsi ini merupakan hasil karya tulis saya sendiri dan bukan merupakan hasil karya orang lain.
2. Memberikan izin hak bebas Royalti Non-Eksklusif kepada Unpub untuk menyimpan, mengalih-media/formatkan, mengelola, mendistribusikan, dan mempublikasikan karya skripsi saya melalui internet atau media lain bagi kepentingan akademis.

Pernyataan ini saya buat dengan penuh tanggung jawab dan saya bersedia menerima konsekuensi apa pun sesuai dengan aturan yang berlaku apabila dikemudian hari diketahui bahwa pernyataan ini tidak benar.

Medan, 16 Desember 2020



(Ragil Aditya Prayugo)

## SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Ragil Aditya Prayugo  
NPM : 1614370169  
Program Studi : Sistem Komputer  
Jenjang : S-1 (Strata Satu)  
Judul Skripsi : Implementasi Teknik Kompresi Dengan  
Algoritma Sequitur Pada Pesan Teks

Dengan ini menyatakan bahwa:

1. Skripsi ini merupakan hasil karya tulis saya sendiri dan bukan merupakan hasil karya orang lain.
2. Memberikan izin hak bebas Royalti Non-Eksklusif kepada Unpub untuk menyimpan, mengalih-media/formatkan, mengelola, mendistribusikan, dan mempublikasikan karya skripsi saya melalui internet atau media lain bagi kepentingan akademis.

Pernyataan ini saya buat dengan penuh tanggung jawab dan saya bersedia menerima konsekuensi apa pun sesuai dengan aturan yang berlaku apabila dikemudian hari diketahui bahwa pernyataan ini tidak benar.

Medan, 16 Desember 2020



(Ragil Aditya Prayugo)

## **ABSTRAK**

**RAGIL ADITYA PRAYUGO**

**Implementasi Teknik Kompresi Dengan Algoritma Sequitur Pada Pesan  
Teks  
2020**

Data merupakan informasi yang mengandung makna tertentu. Setiap informasi dikemas dalam suatu file yang disebut sebagai data digital. Data ini tidak memiliki fisik tetapi dapat menempati ruang penyimpanan. Seiring perkembangan teknologi, kebutuhan ruang penyimpanan semakin bertambah. Banyaknya data yang akan disimpan akan mempengaruhi kapasitas penyimpanan. Kompresi sangat dibutuhkan dalam meminimalisir penggunaan ruang penyimpanan. Algoritma Sequitur adalah salah satu algoritma kompresi yang dapat melakukan kompresi data. Algoritma ini akan bekerja dengan mencari suku kata yang identik. Suku kata yang sama akan digantikan menjadi karakter baru yang lebih singkat sehingga mengurangi penggunaan data. Algoritma ini dapat melakukan kompresi data lebih dari 30% dari jumlah penggunaan ruang penyimpanan.

**Kata Kunci:** algoritma, kompresi, dekompresi, Sequitur

## KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Kuasa, karena dengan berkat dan rahmat-Nya penulis masih diberikan kesempatan untuk menyelesaikan skripsi ini sebagaimana mestinya. Skripsi ini berjudul "**Implementasi Teknik Kompresi Dengan Algoritma Sequitur Pada Pesan Teks**". Dalam kesempatan ini, penulis mengucapkan rasa terima kasih yang tak terhingga kepada pihak-pihak yang telah membantu dalam penyelesaian skripsi ini. Penulis ingin mengucapkan terima kasih kepada :

1. Orang tua saya yang selalu memberikan semangat, dukungan dan motivasi dalam penyusunan skripsi ini.
2. Bapak Dr. H. Muhammad Isa Indrawan, S.E., M.M., selaku Rektor Universitas Pembangunan Panca Budi Medan.
3. Bapak Ir. Bhakti Alamsyah, M.T., Ph.D., selaku Rektor I, Universitas Pembangunan Panca Budi Medan.
4. Bapak Hamdani, S.T., M.T., selaku Dekan Fakultas Sains dan Teknologi Universitas Pembangunan Panca Budi Medan.
5. Bapak Eko Hariyanto, S.Kom., M.Kom., selaku Ketua Program Studi Sistem Komputer Universitas Pembangunan Panca Budi Medan.
6. Bapak Dian Kurnia, S.Kom., M.Kom., selaku Dosen Pembimbing I yang telah memberikan arahan dan membimbing dalam penyelesaian skripsi ini.
7. Bapak Supiyandi, S.Kom., M.Kom., selaku Dosen Pembimbing II yang telah memberikan ilmu pengetahuan, serta bimbingan dalam penyelesaian skripsi ini.
8. Dosen-dosen pada Program Studi Sistem Komputer Fakultas Sains dan Teknologi Universitas Pembangunan Panca Budi Medan.
9. Staff dan karyawan pada Universitas Pembangunan Panca Budi Medan.
10. Seluruh teman-teman penulis dari program studi Sistem Komputer, Fakultas Sains dan Teknologi, Universitas Pembangunan Panca Budi, Medan

Penulis juga menyadari bahwa penyusunan skripsi ini belum mendapatkan kesempurnaan dalam segi penulisan ataupun isi. Hal ini disebabkan pengetahuan penulis yang sangat terbatas. Penulis sangat mengharapkan adanya kritik dan saran dari pembaca untuk dapat memperbaiki isi skripsi.

Medan, 12 September 2020  
Penulis

Ragil Aditya Prayugo  
1614370169

## DAFTAR ISI

<b>KATA PENGANTAR</b> .....	<b>i</b>
<b>DAFTAR ISI</b> .....	<b>ii</b>
<b>DAFTAR GAMBAR</b> .....	<b>iv</b>
<b>DAFTAR TABEL</b> .....	<b>v</b>
<b>BAB I PENDAHULUAN</b> .....	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan Penelitian .....	3
1.5 Manfaat Penelitian .....	3
<b>BAB II LANDASAN TEORI</b> .....	<b>4</b>
2.1 Data .....	4
2.1.1 Bagaimana Data Disimpan .....	5
2.1.2 Jenis data .....	6
2.1.3 Pengelolaan dan Penggunaan Data.....	7
2.2 Media Penyimpanan.....	8
2.2.1 Penyimpanan Magnetik .....	9
2.2.2 Penyimpanan Optical .....	9
2.2.3 Penyimpanan Awan.....	10
2.3 Keamanan Data .....	11
2.3.1 Pentingnya Keamanan Data .....	12
2.3.2 Solusi Keamanan Data .....	13
2.3.3 Kerahasiaan .....	14
2.3.4 Integritas .....	15
2.3.5 Ketersediaan .....	16
2.3.6 Kontrol Akses .....	16
2.4 Algoritma .....	16
2.4.1 Desain Konseptual.....	18
2.4.2 Tugas Algoritma.....	19
2.4.3 Rekayasa Algoritma .....	20
2.5 Kompresi Data .....	20
2.6 Sequitur .....	24
2.7 Unified Modelling Language (UML) .....	27
2.7.1 Use Case Diagram .....	28
2.7.2 Activity Diagram .....	31
2.7.3 Sequence Diagram.....	32
2.8 Flowchart .....	33
<b>BAB III METODE PENELITIAN</b> .....	<b>37</b>
3.1 Tahapan Penelitian .....	37
3.2 Perancangan Penelitian .....	41

3.2.1	Use Case Diagram .....	42
3.2.2	Activity Diagram .....	43
3.2.3	Flowchart Kompres .....	45
3.2.4	Flowchart Dekompres .....	46
3.3	Desain Interface .....	47
3.3.1	Menu Utama .....	47
3.3.2	Menu Sequitur .....	48
3.3.3	Menu Info .....	49
3.3.4	Menu About.....	50
<b>BAB IV HASIL DAN PEMBAHASAN .....</b>		<b>51</b>
4.1	Spesifikasi Sistem .....	51
4.1.1	Spesifikasi Perangkat Keras .....	52
4.1.2	Spesifikasi Perangkat Lunak .....	52
4.2	Hasil Antarmuka .....	53
4.2.1	Halaman Menu Utama.....	53
4.2.2	Halaman Info .....	54
4.2.3	Halaman About.....	55
4.2.4	Halaman Sequitur .....	56
4.2.5	Hasil Kompresi .....	57
4.2.6	Hasil Dekompresi .....	58
4.3	Test Perhitungan.....	59
<b>BAB V PENUTUP.....</b>		<b>63</b>
5.1	Kesimpulan .....	63
5.2	Saran.....	63

## **DAFTAR PUSTAKA**



## DAFTAR GAMBAR

Gambar 2.4 Use-case Diagram ATM.....	29
Gambar 3.1 Kerangka Penelitian .....	38
Gambar 3.2 Use Case Diagram .....	44
Gambar 3.3 Activity Diagram.....	45
Gambar 3.4 Flowchart proses kompresi.....	46
Gambar 3.5 Flowchart proses dekompresi.....	47
Gambar 3.6 Tampilan Menu Utama.....	48
Gambar 3.7 Tampilan Menu Sequitur.....	49
Gambar 3.8 Tampilan Menu Info.....	50
Gambar 3.9 Tampilan Menu About .....	51
Gambar 4.1 Halaman Menu Utama .....	55
Gambar 4.2 Halaman Info.....	56
Gambar 4.3 Halaman About .....	57
Gambar 4.4 Halaman Sequitur.....	58
Gambar 4.5 Hasil kompresi algoritma Sequitur.....	59
Gambar 4.6 Hasil dekompresi algoritma Sequitur .....	60

## DAFTAR TABEL

Tabel 2.1 Simbol Use Case Diagram .....	30
Tabel 2.2 Simbol Activity Diagram .....	32
Tabel 2.3 Simbol Sequence Diagram.....	33
Tabel 2.4 Simbol Flowchart.....	35
Tabel 4.1 Spesifikasi perangkat keras .....	53
Tabel 4.2 Spesifikasi perangkat lunak .....	53

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Pesatnya perkembangan zaman sekarang membuat penggunaan data semakin besar. Data yang disimpan pada ruang penyimpanan kadang tidak mencukupi. Data membutuhkan suatu tempat penyimpanan yang besar. Banyak media penyimpanan yang menawarkan segala fitur dan fasilitas dalam menyimpan data. Tetapi, semakin besar kapasitas yang diinginkan, semakin mahal harganya. Media penyimpanan yang ada seperti *flashdisk* sudah mendukung penyimpanan dalam jumlah yang besar. Tetapi bagaimana cara untuk menyimpan data dalam suatu media yang tidak memiliki kapasitas yang besar. Kompresi data adalah teknik yang baik dalam memampatkan informasi menjadi lebih kecil dan ramping agar dapat disimpan dalam media yang tidak begitu besar (Supiyandi & Frida, 2018).

Dalam ilmu komputer dikenal istilah yang bernama kompresi data. Kegiatan kompresi data dilakukan bagi seseorang yang ingin memperoleh data dengan ukuran yang lebih kecil dari data aslinya. Kompresi data ini ternyata terdiri dari beberapa jenis. Ada banyak algoritma untuk melakukan kompresi data. Penelitian ini membahas salah satu algoritma kompresi. Algoritma yang dibahas adalah algoritma *Sequitur*. Algoritma ini bekerja dengan cara membaca pasangan huruf atau kata. Setiap huruf dan kata yang berulang akan diperkecil dengan suatu karakter lain, sehingga penggunaan kata tersebut menjadi lebih ramping.

Kompresi data adalah sebuah teknik pada ilmu komputer untuk mengecilkan ukuran data. Beberapa teori menyebutkan kompresi merupakan pemampatan data. Data yang ada dimampatkan menjadi lebih kecil dari ukuran sebelumnya dengan tujuan menghemat ruang penyimpanan. Apabila kompresi data dilakukan, otomatis ruang penyimpanan yang dibutuhkan menjadi lebih sedikit dan efisien.

Untuk mendukung pembuktian proses kompresi dan dekompresi, maka akan diciptakan suatu aplikasi yang akan dibuat oleh penulis adalah dengan menggunakan Visual Studio 2010 dengan berdasarkan judul yang dibawa. Penulis tertarik untuk memilih judul “**Implementasi Teknik Kompresi Dengan Algoritma *Sequitur* Pada Pesan Teks**”.

## **1.2 Rumusan Masalah**

Adapun rumusan masalah yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Bagaimana melakukan proses kompresi data dengan menggunakan algoritma *Sequitur*?
2. Bagaimana menentukan karakter yang memiliki pasangan pada teks?
3. Bagaimana mengembalikan data dari *file* terkompresi?

## **1.3 Batasan Masalah**

Adapun batasan masalah yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Pesan yang akan dimampatkan berupa pesan teks.
2. Batas karakter yang digunakan adalah sebanyak 1000 karakter.
3. Bahasa pemrograman yang digunakan adalah Microsoft Visual Basic.Net 2010.
4. Program aplikasi berbasis desktop dan tidak online.

#### **1.4 Tujuan Penelitian**

Adapun tujuan penelitian yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Untuk melakukan proses kompresi data dengan menggunakan algoritma *Sequitur*.
2. Untuk menentukan karakter yang memiliki pasangan pada teks.
3. Untuk mengembalikan data dari *file* terkompresi.

#### **1.5 Manfaat Penelitian**

Adapun manfaat penelitian yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Dapat mengamankan pesan yang akan dikirim ke orang lain.
2. Menghemat ruang penyimpanan.
3. Mempercepat proses *transfer* data.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Data**

Data, dalam konteks komputasi, mengacu pada bagian informasi digital yang berbeda. Data biasanya diformat dengan cara tertentu dan dapat ada dalam berbagai bentuk, seperti angka, teks, dll. Ketika digunakan dalam konteks media transmisi, data merujuk ke informasi dalam format digital biner. Data adalah istilah luas dalam teknologi komputer, tetapi sering digunakan untuk mengidentifikasi dan memisahkan informasi dari bit belaka. Dalam telekomunikasi, data sering merujuk pada informasi digital, bukan analog. Tidak seperti transmisi analog, yang memerlukan koneksi garis keras selama durasi transmisi, data digital dikirim dalam paket (Sun et al., 2014).

Dalam komputasi, data adalah informasi yang telah diterjemahkan ke dalam bentuk yang efisien untuk pergerakan atau pemrosesan. Relatif terhadap komputer dan media transmisi saat ini, data adalah informasi yang diubah menjadi bentuk digital biner. Data dapat diterima untuk digunakan sebagai subjek tunggal atau subjek jamak. Data mentah adalah istilah yang digunakan untuk menggambarkan data dalam format digital paling dasar.

Konsep data dalam konteks komputasi berakar pada karya Claude Shannon, seorang ahli matematika Amerika yang dikenal sebagai bapak teori informasi. Dia mengantarkan konsep digital biner berdasarkan penerapan logika Boolean dua nilai ke sirkuit elektronik. *Format* digit biner mendasari CPU, memori *semikonduktor*

dan *disk drive*, serta banyak perangkat periferan yang umum dalam komputasi saat ini. Input komputer awal untuk kontrol dan data berupa kartu *punch*, diikuti oleh pita magnetik dan *hard disk*.

Pada awalnya, pentingnya data dalam komputasi bisnis menjadi jelas dengan popularitas istilah "pemrosesan data" dan "pemrosesan data elektronik," yang, untuk beberapa waktu, datang untuk mencakup keseluruhan dari apa yang sekarang dikenal sebagai teknologi informasi. Selama sejarah komputasi perusahaan, spesialisasi terjadi, dan profesi data yang berbeda muncul seiring dengan pertumbuhan pemrosesan data perusahaan.

### **2.1.1 Bagaimana Data Disimpan**

Komputer mewakili data, termasuk video, gambar, suara dan teks, sebagai nilai *biner* menggunakan pola hanya dua angka: 1 dan 0. Sedikit adalah unit data terkecil dan hanya mewakili nilai tunggal. Satu *byte* terdiri dari delapan *digit biner*. Penyimpanan dan memori diukur dalam megabita dan *gigabita*.

Unit-unit pengukuran data terus bertambah seiring dengan meningkatnya jumlah data yang dikumpulkan dan disimpan. Istilah "*brontobyte*" yang relatif baru, misalnya, adalah penyimpanan data yang setara dengan 10 hingga 27 byte. Data dapat disimpan dalam *format file*, seperti pada sistem *mainframe* menggunakan *ISAM* dan *VSAM*. *Format file* lain untuk penyimpanan, konversi, dan pemrosesan data termasuk nilai yang dipisah koma. *Format* ini terus menemukan kegunaan di berbagai jenis mesin, bahkan ketika pendekatan yang lebih berorientasi data terstruktur memperoleh pijakan dalam komputasi perusahaan. Spesialisasi yang

lebih besar dikembangkan sebagai basis data, sistem manajemen basis data, dan kemudian teknologi basis data relasional muncul untuk mengatur informasi (Zhang et al., 2009).

### 2.1.2 Jenis data

Pertumbuhan web dan telepon pintar selama dekade terakhir menyebabkan peningkatan dalam penciptaan data digital. Data sekarang termasuk informasi teks, audio dan video, serta catatan aktivitas log dan web. Banyak dari itu adalah data yang tidak terstruktur.

Istilah big data telah digunakan untuk menggambarkan data dalam kisaran *petabyte* atau lebih besar. Tulisan singkat menggambarkan data besar dengan 3V - *volume*, variasi, dan kecepatan. Ketika *e-commerce* berbasis *web* telah menyebar, model bisnis berbasis data besar telah berevolusi yang memperlakukan data sebagai aset. Tren semacam itu juga telah menimbulkan keasyikan yang lebih besar dengan penggunaan sosial data dan *privasi* data.

Data memiliki makna di luar penggunaannya dalam aplikasi komputasi yang berorientasi pada pemrosesan data. Misalnya, dalam interkoneksi komponen elektronik dan komunikasi jaringan, istilah data sering dibedakan dari "informasi kontrol," "*bit* kontrol," dan istilah serupa untuk mengidentifikasi konten utama dari unit transmisi. Selain itu, dalam sains, istilah data digunakan untuk menggambarkan kumpulan fakta. Itu juga terjadi di bidang-bidang seperti keuangan, pemasaran, *demografi* dan kesehatan.



### 2.1.3 Pengelolaan dan Penggunaan Data

Dengan semakin banyaknya data dalam organisasi, penekanan tambahan telah ditempatkan pada memastikan kualitas data dengan mengurangi duplikasi dan menjamin yang paling akurat, catatan saat ini digunakan. Banyak langkah yang terlibat dengan manajemen data modern termasuk pembersihan data, serta mengekstrak, mengubah dan memuat (*ETL*) proses untuk mengintegrasikan data. Data untuk diproses telah dilengkapi dengan metadata, kadang-kadang disebut sebagai "data tentang data," yang membantu *administrator* dan pengguna memahami database dan data lainnya.

Analisis yang menggabungkan data terstruktur dan tidak terstruktur menjadi bermanfaat, karena organisasi berupaya memanfaatkan informasi tersebut. Sistem untuk analitik semacam itu semakin berupaya untuk kinerja waktu-nyata, sehingga mereka dibangun untuk menangani data yang masuk yang dikonsumsi dengan tingkat konsumsi tinggi, dan untuk memproses aliran data untuk penggunaan langsung dalam operasi.

Seiring waktu, gagasan basis data untuk operasi dan transaksi telah diperluas ke basis data untuk pelaporan dan analitik data prediktif. Contoh utama adalah gudang data, yang dioptimalkan untuk memproses pertanyaan tentang operasi untuk analisis bisnis dan pemimpin bisnis. Meningkatnya penekanan pada menemukan pola dan memprediksi hasil bisnis telah mengarah pada pengembangan teknik penambangan data (Barone et al., 2017).

## 2.2 Media Penyimpanan

Sistem pada Komputer memiliki 4 komponen utama dari perangkat kerasnya yaitu *Central Processing Unit (CPU)*, *Primary Storage/Memori Utama*, *Secondary Storage/Memori Sekunder*, dan *Input-output Device*. Ke empat komponen itu harus ada untuk menjalankan suatu perangkat komputer agar berjalan dengan baik. Salah satu perangkat keras yang mengalami perubahan yang sangat pesat adalah terletak pada Media Penyimpanan Data. Peran media penyimpanan data pada perangkat komputer sangat penting, karena mereka yang mengatur mengenai berjalannya sebuah proses dan menyimpan data dengan aman. Banyak jenis atau macam-macam media penyimpanan data pada komputer salah satu media penyimpanan seperti yang sering kita kenal adalah *Hard disk* yang biasa dipakai komputer dan Kartu Memori (*memory card*) yang biasa dipakai pada ponsel.

*Computer Data Storage (Penyimpanan Data Komputer)* adalah media yang digunakan dengan fungsi untuk menyimpan berbagai macam data digital yang tersedia pada perangkat komputer dengan waktu tertentu sehingga dapat dibaca dan dibuka kembali untuk diproses ulang pada perangkat. Untuk saat ini Media penyimpanan Komputer terbagi menjadi 3 kategori, yaitu Media penyimpanan Magnetik (*Magnetic Disk*), Media Penyimpanan Optical (*Optical Disk*), dan Media Penyimpanan Awan (*Cloud Storage*). Dan selanjutnya akan saya bahas ketiga kategori tersebut secara detail, jenis-jenis dan juga contoh dari setiap kategori tersebut.

### 2.2.1 Penyimpanan Magnetik

Penyimpanan Magnetik merupakan media penyimpanan yang termasuk ke dalam penyimpanan sekunder (secondary storage) yang paling banyak dipakai pada sistem komputer modern. Kelebihan dan Kekurangan dari penyimpanan magnetik antara lain:

1. Kelebihan : Kapasitas penyimpanan pada media ini lebih besar dari media penyimpanan lainnya bahkan sudah mencapai *Petabyte* dan Kecepatan akses datanya tinggi.
2. Kekurangan : Harganya lebih mahal jika dibandingkan dengan media penyimpanan lainnya.

Pada saat *disk* digunakan, motor *drive* berputar dengan kecepatan yang sangat tinggi. Ada sebuah *read-write* head yang ditempatkan di atas permukaan piringan tersebut. Permukaan disk terbagi atas beberapa track yang masih terbagi lagi menjadi beberapa sektor. Cakram *fixed-head* memiliki satu *head* untuk tiap-tiap *track*, sedangkan cakram *moving-head* (atau sering dikenal dengan nama cakram keras) hanya memiliki satu *head* yang harus dipindah-pindahkan untuk mengakses dari satu *track* ke *track* yang lainnya.

### 2.2.2 Penyimpanan Optical

Penyimpanan *optical* adalah media yang menyimpan data komputer yang dapat ditulis dan dibaca dengan menggunakan laser bertenaga rendah. Kelebihan dan Kekurangan dari penyimpanan *optical* antara lain:

1. Kelebihan : Beratnya lebih ringan dari beberapa media penyimpanan *Magnetic Disk*.
2. Kekurangan : Kapasitas memorinya lebih kecil dari *Magnetic Disk* dan Jika tergores maka resikonya data tidak akan terbaca.

Media penyimpanan tersebut berputar dengan sangat kencang (putaran tersebut mempengaruhi kecepatan *transfer* data) dengan membaca data melalui *optik* yang berada pada perangkat pembacanya.

### **2.2.3 Penyimpanan Awan**

Penyimpanan Awan merupakan media yang masih tergolong baru, media ini bersifat *online* dan tidak menggunakan kapasitas data memori pada perangkat karena mereka menggunakan penyimpanan yang terdapat pada Internet. Kelebihan dan Kekurangan dari penyimpanan awan antara lain:

1. Kelebihan : Tidak memerlukan perangkat untuk menyimpan data.
2. Kekurangan : Sering terjadi kesalahan pada Server dengan resiko data akan hilang dan juga dikenakan akses koneksi data.

Untuk dapat menyimpan data pada media ini kita diharuskan untuk mengunggah *file* tersebut dan untuk mengambil data kita harus mengunduh *file* tersebut.

### 2.3 Keamanan Data

Keamanan data adalah seperangkat standar dan teknologi yang melindungi data dari kehancuran, modifikasi, atau pengungkapan yang disengaja atau tidak disengaja. Keamanan data dapat diterapkan dengan menggunakan berbagai teknik dan teknologi, termasuk kontrol administratif, keamanan fisik, kontrol logis, standar organisasi, dan teknik perlindungan lainnya yang membatasi akses ke pengguna atau proses yang tidak sah atau berbahaya (Rao & Selvamani, 2015).

Keamanan data mengacu pada langkah-langkah *privasi* digital pelindung yang diterapkan untuk mencegah akses tidak sah ke komputer, database, dan situs *web*. Keamanan data juga melindungi data dari korupsi. Keamanan data adalah aspek penting dari TI untuk organisasi dari berbagai ukuran dan tipe. Keamanan data juga dikenal sebagai keamanan informasi atau keamanan komputer.

Contoh teknologi keamanan data termasuk *backup*, masking data dan penghapusan data. Ukuran teknologi keamanan data utama adalah *enkripsi*, di mana data digital, perangkat lunak / perangkat keras, dan *hard drive* *dienkripsi* dan karenanya tidak dapat dibaca oleh pengguna dan peretas yang tidak sah. Salah satu metode yang paling umum dijumpai dalam mempraktikkan keamanan data adalah penggunaan otentikasi. Dengan otentikasi, pengguna harus memberikan kata sandi, kode, data *biometrik*, atau bentuk data lainnya untuk memverifikasi identitas sebelum akses ke sistem atau data diberikan. Keamanan data juga sangat penting untuk catatan perawatan kesehatan, sehingga pendukung kesehatan dan praktisi medis di AS dan negara-negara lain berupaya menerapkan privasi rekam medis

elektronik dengan menciptakan kesadaran tentang hak-hak pasien terkait dengan pelepasan data ke laboratorium, dokter, rumah sakit dan fasilitas medis lainnya.

### **2.3.1 Pentingnya Keamanan Data**

Semua bisnis saat ini menangani data hingga taraf tertentu. Dari raksasa perbankan yang menangani data pribadi dan keuangan dalam *volume* besar hingga bisnis satu orang yang menyimpan detail kontak pelanggannya di ponsel, data berperan di perusahaan baik besar maupun kecil.

Tujuan utama keamanan data adalah untuk melindungi data yang dikumpulkan, disimpan, diterima, atau ditransmisikan oleh suatu organisasi. Kepatuhan juga merupakan pertimbangan utama. Tidak masalah perangkat, teknologi, atau proses mana yang digunakan untuk mengelola, menyimpan, atau mengumpulkan data, itu harus dilindungi. Pelanggaran data dapat menyebabkan kasus litigasi dan denda yang sangat besar, belum lagi kerusakan reputasi organisasi. Pentingnya melindungi data dari ancaman keamanan lebih penting saat ini daripada sebelumnya.

Keamanan data mengacu pada proses melindungi data dari akses yang tidak sah dan korupsi data sepanjang siklus hidupnya. Keamanan data termasuk enkripsi data, tokenization, dan praktik manajemen kunci yang melindungi data di semua aplikasi dan platform. Organisasi di seluruh dunia banyak berinvestasi dalam kemampuan pertahanan *cyber* teknologi informasi untuk melindungi aset penting mereka. Apakah suatu perusahaan perlu melindungi merek, modal intelektual, dan informasi pelanggan atau menyediakan kontrol untuk infrastruktur penting, sarana

untuk mendeteksi insiden dan merespons melindungi kepentingan organisasi memiliki tiga elemen umum: orang, proses, dan teknologi.

### 2.3.2 Solusi Keamanan Data

Data membutuhkan enkripsi dalam mengamankan informasi yang ada dalam data tersebut. Dengan enkripsi data canggih, tokenization, dan manajemen utama untuk melindungi data di seluruh aplikasi, transaksi, penyimpanan, dan platform big data, Teknik ini menyederhanakan perlindungan data sensitif bahkan dalam kasus penggunaan yang paling kompleks sekalipun. Beberapa model keamanan data antara lain:

1. Keamanan akses *cloud - Platform* perlindungan yang memungkinkan Anda untuk pindah ke *cloud* dengan aman sambil melindungi data dalam aplikasi *cloud*.
2. Enkripsi data - Solusi keamanan data-sentris dan tokenisasi yang melindungi data di lingkungan perusahaan, *cloud*, seluler, dan data besar.
3. Modul keamanan perangkat keras - Modul keamanan perangkat keras yang menjaga data keuangan dan memenuhi persyaratan keamanan dan kepatuhan industri.
4. Manajemen kunci - Solusi yang melindungi data dan memungkinkan kepatuhan regulasi industri.
5. *Enterprise Data Protection* - Solusi yang menyediakan pendekatan data-centric *end-to-end* untuk perlindungan data perusahaan.

6. Keamanan Pembayaran - Solusi menyediakan enkripsi dan tokenisasi point-to-point lengkap untuk transaksi pembayaran ritel, memungkinkan pengurangan lingkup PCI.
7. *Big Data, Hadoop*, dan perlindungan data *IofT* - Solusi yang melindungi data sensitif di Danau Data - termasuk Hadoop, Teradata, *Micro Focus Vertica*, dan *platform Big Data* lainnya.
8. Keamanan Aplikasi Seluler - Melindungi data sensitif di aplikasi seluler asli sembari menjaga data dari ujung ke ujung.
9. Keamanan Peramban *Web* - Melindungi data sensitif yang diambil di peramban, dari titik pelanggan memasukkan pemegang kartu atau data pribadi dan menjaganya agar tetap terlindungi melalui ekosistem ke tujuan tuan rumah tepercaya.
10. *eMail Security* - Solusi yang menyediakan enkripsi ujung ke ujung untuk email dan olahpesan seluler, menjaga informasi pribadi dan informasi kesehatan pribadi tetap aman dan pribadi.

### **2.3.3 Kerahasiaan**

Kerahasiaan mengacu pada melindungi informasi agar tidak diakses oleh pihak yang tidak berwenang. Dengan kata lain, hanya orang yang diberi wewenang untuk melakukannya yang dapat memperoleh akses ke data sensitif. Bayangkan catatan bank harus dapat diakses, tentu saja, dan karyawan di bank yang membantu dalam menjalankan transaksi harus dapat mengaksesnya, tetapi tidak ada orang lain yang seharusnya. Kegagalan untuk menjaga kerahasiaan berarti bahwa seseorang



yang seharusnya tidak memiliki akses telah berhasil mendapatkannya, melalui perilaku yang disengaja atau karena kecelakaan. Kegagalan kerahasiaan seperti itu, umumnya dikenal sebagai pelanggaran, biasanya tidak dapat diperbaiki. Setelah rahasia itu terungkap, tidak ada cara untuk mengetahuinya. Jika catatan bank diposting di situs *web* publik, semua orang dapat mengetahui nomor rekening bank, saldo, dll., Informasi itu tidak dapat dihapus dari pikiran, kertas, komputer, dan tempat lain mereka. Hampir semua insiden keamanan utama yang dilaporkan di media saat ini melibatkan kerugian besar kerahasiaan. Jadi, secara ringkas, pelanggaran kerahasiaan berarti bahwa seseorang memperoleh akses ke informasi yang seharusnya tidak memiliki akses ke sana.

#### **2.3.4 Integritas**

*Integritas* mengacu pada memastikan keaslian informasi — bahwa informasi tidak diubah, dan bahwa sumber informasi itu asli. Bayangkan jika seseorang memiliki situs *web* dan Anda menjual produk di situs itu. Sekarang bayangkan penyerang dapat berbelanja di situs *web* dan dengan jahat mengubah harga produk Anda sehingga mereka dapat membeli apa pun dengan harga berapa pun yang mereka pilih. Itu akan menjadi kegagalan *integritas* karena informasi dalam hal ini, harga suatu produk telah diubah dan perubahan ini tidak dapat digagalkan. Contoh lain dari kegagalan integritas adalah ketika seseorang mencoba terhubung ke situs *web* dan penyerang jahat antara Anda dan situs *web* mengalihkan lalu lintas ke situs web yang berbeda. Dalam hal ini, situs yang dituju tidak asli.

### 2.3.5 Ketersediaan

Ketersediaan berarti informasi dapat diakses oleh pengguna yang berwenang. Jika penyerang tidak dapat mengkompromikan dua elemen pertama dari keamanan informasi (lihat di atas) mereka dapat mencoba melakukan serangan seperti penolakan layanan yang akan menurunkan server, membuat situs *web* tidak tersedia untuk pengguna yang sah karena kurangnya ketersediaan.

### 2.3.6 Kontrol Akses

Kesalahan terbesar yang bisa dilakukan oleh perancang aplikasi adalah mengabaikan kontrol akses sebagai bagian dari fungsionalitas yang diperlukan. Jarang bahwa setiap pengguna atau sistem yang berinteraksi dengan suatu aplikasi harus memiliki hak yang sama di seluruh aplikasi itu. Beberapa pengguna mungkin memerlukan akses ke data tertentu dan bukan yang lain; beberapa sistem harus atau tidak dapat mengakses aplikasi. Akses ke komponen, fungsi, atau modul tertentu dalam aplikasi juga harus dikontrol. Kontrol akses juga penting untuk kepatuhan audit dan peraturan. Beberapa cara umum mengelola kontrol akses adalah:

1. Baca, tulis, dan jalankan hak istimewa: *File*
2. Kontrol akses berbasis peran: *administrator*, pengguna
3. Alamat IP akses berbasis *host*, nama mesin
4. Objek kode kontrol akses tingkat objek, banyak pembaca / penulis tunggal

## 2.4 Algoritma

Untuk membuat komputer melakukan apa pun, seseorang harus menulis program komputer. Untuk menulis program komputer, seseorang harus memberi

tahu komputer, langkah demi langkah, persis apa yang seseorang inginkan. Komputer kemudian "mengeksekusi" program, mengikuti setiap langkah secara mekanis, untuk mencapai tujuan akhir. Ketika seseorang memberi tahu komputer apa yang harus dilakukan, seseorang juga harus memilih bagaimana melakukannya. Di situlah algoritma komputer masuk. Algoritma adalah teknik dasar yang digunakan untuk menyelesaikan pekerjaan (Gurevich, 2012). Mari kita ikuti contoh untuk membantu mendapatkan pemahaman tentang konsep algoritma. Katakanlah seseorang memiliki seorang teman yang tiba di bandara, dan teman seseorang perlu pergi dari bandara ke rumah. Berikut adalah empat algoritma berbeda yang mungkin akan diberikan kepada orang lain untuk sampai ke rumah:

Dalam pemrograman komputer, seringkali ada banyak cara berbeda - algoritma - untuk menyelesaikan tugas yang diberikan. Setiap algoritma memiliki kelebihan dan kekurangan dalam situasi yang berbeda. Penyortiran adalah satu tempat di mana banyak penelitian telah dilakukan karena komputer menghabiskan banyak daftar penyortiran waktu. Berikut adalah lima algoritma berbeda yang digunakan dalam penyortiran:

1. *Bin Sort*
2. *Merge Sort*
3. *Bubble Sort*
4. *Shell Short*
5. *Quick Sort*

Jika ada sejuta nilai integer antara 1 dan 10 dan perlu diurutkan, jenis *bin sort* adalah algoritma yang tepat untuk digunakan. Jika Anda memiliki sejuta judul buku, quicksort mungkin merupakan algoritma terbaik. Dengan mengetahui kekuatan dan kelemahan dari berbagai algoritma, Anda memilih yang terbaik untuk tugas yang ada.

#### **2.4.1 Desain Konseptual**

Algoritma adalah serangkaian instruksi, sering disebut sebagai "proses," yang harus diikuti ketika memecahkan masalah tertentu. Meskipun secara teknis tidak dibatasi oleh definisi, kata itu hampir selalu terkait dengan komputer, karena algoritma yang diproses komputer dapat mengatasi masalah yang jauh lebih besar daripada manusia, jauh lebih cepat. Karena komputasi modern menggunakan algoritma jauh lebih sering daripada pada titik lain dalam sejarah manusia, bidang telah tumbuh di sekitar desain, analisis, dan penyempurnaan. Bidang desain algoritma membutuhkan latar belakang matematika yang kuat, dengan gelar ilmu komputer yang sangat dicari kualifikasi. Ini menawarkan semakin banyak pilihan karir yang sangat dikompensasi, karena kebutuhan akan lebih banyak (dan juga lebih canggih) algoritma terus meningkat.

Pada tingkat yang paling sederhana, algoritma pada dasarnya hanya seperangkat instruksi yang diperlukan untuk menyelesaikan tugas. Pengembangan algoritma, meskipun umumnya tidak disebut demikian, telah menjadi kebiasaan yang populer dan pengejaran profesional untuk semua catatan sejarah. Jauh sebelum fajar era komputer modern, orang menetapkan rutinitas yang telah

ditentukan untuk bagaimana mereka akan melakukan tugas sehari-hari, sering menuliskan daftar langkah-langkah yang harus diambil untuk mencapai tujuan penting, mengurangi risiko melupakan sesuatu yang penting. Ini, pada dasarnya, adalah apa itu algoritma. *Desainer* mengambil pendekatan yang mirip dengan pengembangan algoritma untuk tujuan komputasi: pertama, mereka melihat masalah. Kemudian, mereka menguraikan langkah-langkah yang akan diperlukan untuk menyelesaikannya. Akhirnya, mereka mengembangkan serangkaian operasi matematika untuk mencapai langkah-langkah tersebut.

#### **2.4.2 Tugas Algoritma**

Tugas sederhana dapat diselesaikan dengan algoritma yang dihasilkan dengan beberapa menit, atau paling banyak pekerjaan pagi. Tingkat kompleksitas menjalankan tantangan yang panjang, namun, sampai pada masalah yang sangat rumit sehingga mereka telah menghalangi matematikawan yang tak terhitung jumlahnya selama bertahun-tahun - atau bahkan berabad-abad. Komputer modern menghadapi masalah pada tingkat ini di bidang-bidang seperti keamanan dunia maya, serta penanganan data besar - penyortiran set data yang efisien dan menyeluruh sedemikian besar sehingga bahkan komputer standar tidak dapat memprosesnya secara tepat waktu. Contoh data besar mungkin termasuk "setiap artikel di *Wikipedia*," "setiap halaman *web* yang diindeks dan diarsipkan akan kembali ke tahun 1998," atau "enam bulan terakhir pembelian online yang dilakukan di Amerika."

### 2.4.3 Rekayasa Algoritma

Ketika desain algoritma baru diterapkan dalam istilah praktis, disiplin terkait dikenal sebagai rekayasa algoritma. Kedua fungsi tersebut sering dilakukan oleh orang yang sama, meskipun organisasi yang lebih besar (seperti *Amazon* dan *Google*) mempekerjakan *desainer* dan *insinyur* khusus, mengingat tingkat kebutuhan mereka akan algoritma baru dan khusus. Seperti proses desain, rekayasa algoritma sering kali melibatkan akreditasi *sains* komputer, dengan latar belakang yang kuat dalam matematika: di mana mereka ada sebagai profesi yang terpisah dan terspesialisasi, insinyur algoritma mengambil ide-ide konseptual dari *desainer* dan proses kreatif dari mereka yang akan dipahami oleh komputer. Dengan kemajuan teknologi digital yang mantap, para insinyur yang berdedikasi akan terus menjadi semakin umum.

## 2.5 Kompresi Data

Kompresi data, juga disebut pemadatan, proses mengurangi jumlah data yang diperlukan untuk penyimpanan atau transmisi informasi yang diberikan, biasanya dengan menggunakan teknik pengkodean. Kompresi mendahului teknologi digital, yang telah digunakan dalam Kode Morse, yang menetapkan kode terpendek ke karakter yang paling umum, dan dalam telepon, yang memotong frekuensi tinggi dalam transmisi suara. Saat ini, ketika gambar digital yang tidak terkompresi mungkin memerlukan 20 *megabita*, kompresi data penting dalam menyimpan informasi secara digital pada *disk* komputer dan dalam mentransmisikannya melalui jaringan komunikasi (Marlina et al., 2017).

Informasi dikodekan secara digital sebagai pola 0s dan 1s, atau bit (digit *biner*). *Alfabet* empat huruf (a, e, r, t) akan membutuhkan dua bit per karakter jika semua karakter memiliki kemungkinan yang sama. Semua huruf dalam kalimat "Seekor tikus makan tart saat minum teh," dengan demikian dapat dikodekan dengan  $2 \times 18 = 36$  bit. Karena a paling sering dalam teks ini, dengan t yang paling umum kedua, menetapkan kode biner panjang *variabel* — a: 0, t: 10, r: 110, e: 111 — akan menghasilkan pesan terkompresi yang hanya 32 bit. Pengkodean ini memiliki properti penting bahwa tidak ada kode yang merupakan awalan dari yang lain. Artinya, tidak ada bit tambahan yang diperlukan untuk memisahkan kode huruf: 010111 menerjemahkan secara tidak ambigu seperti makan.

Kompresi data dapat berupa *lossless* (tepat) atau *lossy* (tidak tepat). Kompresi *lossless* dapat dibalik untuk menghasilkan data asli, sementara kompresi *lossy* kehilangan detail atau menyebabkan kesalahan kecil saat pembalikan. Kompresi *lossless* diperlukan untuk teks, di mana setiap karakter penting, sementara kompresi *lossy* dapat diterima untuk gambar atau suara (keterbatasan spektrum frekuensi di telepon menjadi contoh kompresi *lossy*). Tiga program kompresi yang paling umum untuk data umum adalah *Zip* (pada komputer yang menggunakan sistem operasi *Windows*), *StuffIt* (pada komputer *Apple*), dan *gzip* (pada komputer yang menjalankan *UNIX*); semua menggunakan kompresi *lossless*. Format umum untuk mengompresi gambar statis, terutama untuk tampilan melalui Internet, adalah *GIF* (format pertukaran grafik), yang juga *lossless* kecuali bahwa gambarnya dibatasi hingga 256 warna. Rentang warna yang lebih besar dapat digunakan dengan format standar *JPEG* (kelompok ahli fotografi gabungan), yang

menggunakan teknik *lossless* dan *lossy*, seperti halnya berbagai standar *MPEG* (kelompok ahli gambar bergerak) untuk video (Hemmendinger, 2019).

Agar program kompresi dapat berfungsi, mereka harus memiliki model data yang menjelaskan distribusi karakter, kata, atau elemen lain, seperti frekuensi di mana karakter individu muncul dalam bahasa *Inggris*. Model yang diperbaiki seperti contoh sederhana dari alfabet empat karakter, di atas, mungkin tidak mengkarakterisasi satu teks dengan baik, terutama jika teks tersebut berisi data tabular atau menggunakan kosakata khusus. Dalam kasus ini, model adaptif, yang berasal dari teks itu sendiri, mungkin lebih unggul. Model adaptif memperkirakan distribusi karakter atau kata berdasarkan apa yang telah mereka proses sejauh ini. Sifat penting dari pemodelan adaptif adalah bahwa jika program kompresi dan dekompresi menggunakan aturan yang sama persis untuk membentuk model dan tabel kode yang sama yang mereka tetapkan untuk elemen-elemennya, maka model itu sendiri tidak perlu dikirim ke program dekompresi. Misalnya, jika program kompresi memberikan kode yang tersedia berikutnya untuk ketika dilihat untuk ketiga kalinya, dekompresi akan mengikuti aturan yang sama dan mengharapkan kode itu untuk setelah kejadian kedua.

Pengodean dapat bekerja dengan simbol individu atau dengan kata-kata. Kode *Huffman* menggunakan model statis dan membuat kode seperti yang diilustrasikan sebelumnya dalam alfabet empat huruf (Suherman & Siahaan, 2016). Pengkodean aritmatika mengkodekan string simbol sebagai rentang bilangan real dan mencapai kode yang hampir optimal. Ini lebih lambat daripada pengkodean *Huffman* tetapi cocok untuk model adaptif. *Run-length encoding (RLE)* baik untuk



data berulang, menggantinya dengan hitungan dan satu salinan item yang diulang. Metode kamus adaptif membuat tabel string dan kemudian mengganti kemunculannya dengan kode yang lebih pendek. Algoritma *Lempel-Ziv*, ditemukan oleh ilmuwan komputer Israel Abraham Lempel dan Jacob Ziv, menggunakan teks itu sendiri sebagai kamus, menggantikan kemunculan string yang kemudian dengan angka yang menunjukkan di mana itu terjadi sebelum dan panjangnya. *Zip* dan *gzip* menggunakan variasi dari algoritma *Lempel-Ziv*.

Kompresi *lossy* memperluas teknik ini dengan menghapus detail. Secara khusus, gambar digital terdiri dari piksel yang mewakili skala abu-abu atau informasi warna. Ketika sebuah piksel hanya berbeda sedikit dari tetangganya, nilainya dapat diganti oleh tetangganya, setelah itu gambar "dihaluskan" dapat dikompres menggunakan *RLE*. Sementara memuluskan sebagian besar gambar akan sangat jelas, perubahan itu jauh kurang terlihat ketika tersebar di bagian-bagian kecil yang tersebar. Metode yang paling umum menggunakan *diskrit cosinus transform*, rumus matematika yang terkait dengan transformasi *Fourier*, yang memecah gambar menjadi bagian-bagian terpisah dari berbagai tingkat kepentingan yang berbeda untuk kualitas gambar. Teknik ini, serta teknik fraktal, dapat mencapai rasio kompresi yang sangat baik. Sementara kinerja kompresi *lossless* diukur dengan tingkat kompresinya, kompresi *lossy* juga dievaluasi berdasarkan kesalahan yang diperkenalkannya. Ada metode matematika untuk menghitung kesalahan, tetapi ukuran kesalahan juga tergantung pada bagaimana data akan digunakan: membuang nada frekuensi tinggi menghasilkan sedikit kerugian untuk

rekaman yang diucapkan, misalnya, tetapi degradasi yang tidak dapat diterima untuk musik.

Gambar video dapat dikompresi dengan hanya menyimpan sedikit perbedaan antara frame yang berurutan. *MPEG-1* adalah umum dalam mengompresi video untuk *CD-ROM*; itu juga merupakan dasar untuk format MP3 yang digunakan untuk kompres musik. *MPEG-2* adalah format kualitas "siaran" yang lebih tinggi yang digunakan untuk *DVD* (lihat compact disc: *DVD*) dan beberapa perangkat jaringan televisi. *MPEG-4* dirancang untuk aplikasi "bandwidth rendah" dan umum untuk menyiarkan video melalui *World Wide Web* (*WWW*). (*MPEG-3* dimasukkan ke dalam *MPEG-2*.) Kompresi video dapat mencapai rasio kompresi mendekati 20 banding 1 dengan distorsi minimal.

Ada *trade-off* antara waktu dan memori yang dibutuhkan algoritma kompresi dan kompresi yang dicapai. Teks bahasa Inggris umumnya dapat dikompresi menjadi setengah atau sepertiga dari ukuran aslinya. Gambar seringkali dapat dikompresi oleh faktor 10 hingga 20 atau lebih. Meskipun pertumbuhan kapasitas penyimpanan komputer dan kecepatan jaringan, kompresi data tetap menjadi alat penting untuk menyimpan dan mengirimkan koleksi data yang semakin besar.

## 2.6 *Sequitur*

*Sequitur* (atau *algoritma Nevill-Manning*) adalah algoritma rekursif yang dikembangkan oleh Craig Nevill-Manning dan Ian H. Witten pada tahun 1997 yang menyimpulkan struktur *hierarkis* (tata bahasa bebas konteks) dari urutan simbol

diskrit. Algoritma beroperasi dalam ruang dan waktu *linier*. Dapat digunakan dalam aplikasi perangkat lunak kompresi data. *Sequitur* adalah metode untuk menyimpulkan hierarki komposisi dari string. Mendeteksi pengulangan dan faktor itu keluar dari string dengan membentuk aturan dalam tata bahasa. Aturan dapat terdiri dari non-terminal, sehingga menimbulkan hirarki. Ini berguna untuk mengenali struktur leksikal dalam string, dan unggul dalam urutan yang sangat panjang.

Algoritma *sequitur* membangun tata bahasa dengan mengganti *frase* berulang dalam urutan yang diberikan dengan aturan baru dan karenanya menghasilkan representasi singkat dari urutan. Misalnya, jika urutannya

$S \rightarrow \text{abcab}$ , algoritma akan menghasilkan

$S \rightarrow \text{AcA}$ ,  $A \rightarrow \text{ab}$ .

Saat memindai urutan input, algoritma mengikuti dua kendala untuk menghasilkan tata bahasa secara efisien: digram *uniqueness* dan *utility rule*. Setiap kali simbol baru dipindai dari urutan, itu akan ditambahkan dengan simbol yang dipindai terakhir untuk membentuk digram baru. Jika digram ini telah terbentuk sebelumnya maka aturan baru dibuat untuk menggantikan kedua kemunculan digram. Oleh karena itu, memastikan bahwa tidak ada digram terjadi lebih dari sekali dalam tata bahasa. Misalnya, dalam urutan  $S \rightarrow \text{abaaba}$ , ketika empat simbol pertama sudah dipindai, digram yang terbentuk adalah  $\text{ab}$ ,  $\text{ba}$ ,  $\text{aa}$ . Ketika simbol kelima dibaca, digram baru  $\text{'ab'}$  terbentuk yang sudah ada. Oleh karena itu, kedua

contoh 'ab' digantikan oleh aturan baru (katakanlah, A) di S. Sekarang, tata bahasa menjadi  $S \rightarrow AaAa$ ,  $A \rightarrow ab$ , dan proses berlanjut sampai tidak ada digram yang berulang dalam tata bahasa.

Batasan ini memastikan bahwa semua aturan digunakan lebih dari sekali di sisi kanan semua produksi tata bahasa, yaitu, jika aturan terjadi sekali saja, harus dihapus dari tata bahasa dan kemunculannya harus diganti dengan simbol dari yang itu dibuat. Misalnya, dalam contoh di atas, jika seseorang memindai simbol terakhir dan menerapkan keunikan digram untuk 'Aa', maka tata bahasa akan menghasilkan  $S \rightarrow BB$ ,  $A \rightarrow ab$ ,  $B \rightarrow Aa$ . Sekarang, aturan 'A' hanya muncul sekali dalam tata bahasa di  $B \rightarrow Aa$ . Oleh karena itu, A dihapus dan akhirnya tata bahasa menjadi

$$S \rightarrow BB, B \rightarrow aba.$$

Batasan ini membantu mengurangi jumlah aturan dalam tata bahasa. Algoritma bekerja dengan memindai urutan simbol terminal dan membangun daftar semua pasangan simbol yang telah dibaca. Kapan pun kemunculan kedua pasangan ditemukan, kedua kejadian tersebut diganti secara berurutan dengan simbol nonterminal yang ditemukan, daftar pasangan simbol disesuaikan agar sesuai dengan urutan yang baru, dan pemindaian berlanjut. Jika simbol nonterminal pasangan digunakan hanya dalam definisi simbol yang baru saja dibuat, simbol yang digunakan diganti dengan definisinya dan simbol tersebut dihapus dari simbol nonterminal yang ditentukan. Setelah pemindaian selesai, urutan yang ditransformasikan dapat ditafsirkan sebagai aturan tingkat atas dalam tata bahasa

untuk urutan asli. Definisi aturan untuk simbol nonterminal yang dikandungnya dapat ditemukan dalam daftar pasangan simbol. Definisi aturan itu sendiri dapat mengandung simbol nonterminal tambahan yang definisi aturannya juga dapat dibaca dari tempat lain dalam daftar pasangan simbol.

## 2.7 *Unified Modelling Language (UML)*

*Unified Modeling Language (UML)* adalah bahasa pemodelan standar yang memungkinkan pengembang menentukan, memvisualisasikan, membuat, dan mendokumentasikan artefak sistem perangkat lunak (Technopedia, 2019). Dengan demikian, *UML* membuat artefak ini dapat diskalakan, aman, dan kuat dalam eksekusi. *UML* adalah aspek penting yang terlibat dalam pengembangan perangkat lunak berorientasi objek. Ini menggunakan notasi grafis untuk membuat model visual dari sistem perangkat lunak. Arsitektur *UML* didasarkan pada fasilitas meta-objek, yang mendefinisikan dasar untuk membuat bahasa pemodelan. Mereka cukup tepat untuk menghasilkan seluruh aplikasi. *UML* yang sepenuhnya dapat dieksekusi dapat digunakan untuk berbagai platform menggunakan teknologi yang berbeda dan dapat digunakan dengan semua proses sepanjang siklus pengembangan perangkat lunak. *UML* dirancang untuk memungkinkan pengguna mengembangkan bahasa pemodelan visual yang ekspresif, siap pakai. Selain itu, mendukung konsep pengembangan tingkat tinggi seperti kerangka kerja, pola, dan kolaborasi (Wasserkrug et al., 2019).

Penggunaan model ini bertujuan untuk mengidentifikasi bagian-bagian yang termasuk dalam lingkup sistem yang dibahas dan bagaimana hubungan antara

sistem dengan subsistem maupun sistem lain diluarnya (Sukmawati & Priyadi, 2019).

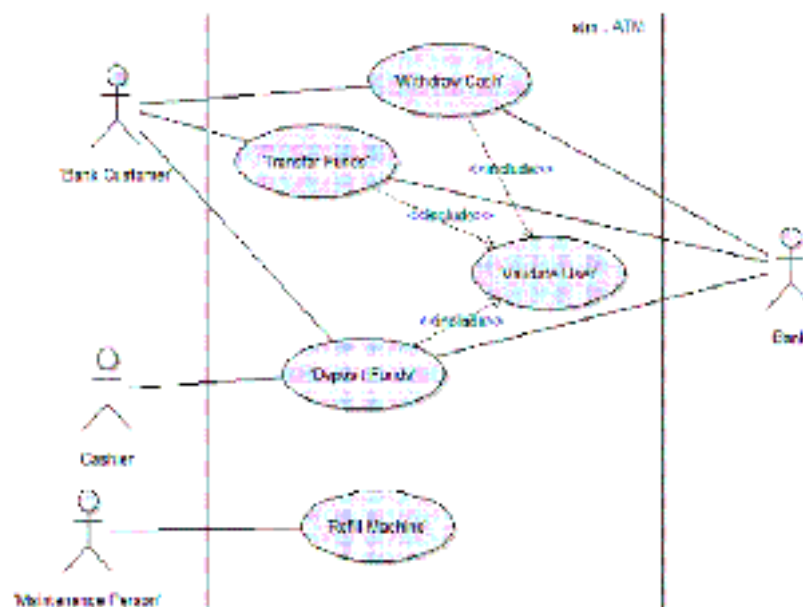
### **2.7.1 Use Case Diagram**

*Use Case Diagram* adalah model tentang bagaimana berbagai jenis pengguna berinteraksi dengan sistem untuk memecahkan masalah. Dengan demikian, ini menggambarkan tujuan pengguna, interaksi antara pengguna dan sistem, dan perilaku sistem yang diperlukan dalam memenuhi tujuan-tujuan ini. Model *use-case* terdiri dari sejumlah elemen model. Elemen model yang paling penting adalah kasus penggunaan, aktor dan hubungan di antara mereka. Diagram *use-case* digunakan untuk menggambarkan secara grafis subset dari model untuk menyederhanakan komunikasi. Biasanya akan ada beberapa diagram kasus penggunaan yang terkait dengan model yang diberikan, masing-masing menunjukkan subset elemen model yang relevan untuk tujuan tertentu. Elemen model yang sama dapat ditampilkan pada beberapa diagram *use-case*, tetapi setiap instance harus konsisten. Jika alat digunakan untuk mempertahankan model *use-case*, kendala konsistensi ini otomatis sehingga setiap perubahan pada elemen model (mengubah nama misalnya) akan secara otomatis tercermin dalam setiap diagram *use-case* yang menunjukkan elemen itu (UTM, 2019).

Model *use-case* dapat berisi paket yang digunakan untuk menyusun model untuk menyederhanakan analisis, komunikasi, navigasi, pengembangan, pemeliharaan, dan perencanaan. Faktanya, sebagian besar model *use-case* adalah tekstual, dengan teks yang ditangkap dalam Spesifikasi *Use-Case* yang terkait

dengan setiap elemen model *use-case*. Spesifikasi ini menjelaskan alur peristiwa *use case*. Model *use-case* berfungsi sebagai utas pemersatu sepanjang pengembangan sistem. Ini digunakan sebagai spesifikasi utama dari persyaratan fungsional untuk sistem, sebagai dasar untuk analisis dan desain, sebagai *input* untuk perencanaan iterasi, sebagai dasar mendefinisikan kasus uji dan sebagai dasar untuk dokumentasi pengguna. (Kurniawan, 2018).

*Use case diagram* merupakan suatu diagram yang berisi *use case*, *actor*, serta *relationship* diantaranya. *Use Case Diagram* dapat digunakan untuk kebutuhan apa saja yang diperlukan dalam suatu sistem, sehingga sistem dapat digambarkan dengan jelas bagaimana proses dari sistem tersebut, bagaimana cara aktor menggunakan sistem, serta apa saja yang dapat dilakukan pada suatu sistem.




**Gambar 2.1 Use-case Diagram ATM**





Sumber: (Uml-diagrams.org, 2019)

Gambar di atas adalah contoh dari penggunaan *use-case* diagram pada mesin ATM. *Use-case* memiliki beberapa simbol untuk menyatakan kegiatan dari *use-case* tersebut. Adapun simbol dari *use case* adalah sebagai berikut:

**Tabel 2.1 Simbol Use Case Diagram**

No	Gambar	Nama	Keterangan
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
3		<i>Generalization</i>	Hubungan dimana objek anak berbagi perilaku dan struktur data dari objek yang ada di atasnya.
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.



6		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Sumber: (Kurniawan, 2018)






### 2.7.2 Activity Diagram

*Activity Diagram* (Diagram Aktifitas) menggambarkan berbagai alir aktifitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir

berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir (Ladjamudin, 2017).

*Activity diagram* menurut adalah salah satu cara untuk memodelkan *event-event* yang terjadi dalam suatu *use case*. Diagram ini juga dapat digantikan dengan sejumlah teks.

**Tabel 2.2 Simbol Activity Diagram**

No	Gambar	Nama	Keterangan
1		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		<i>Initial Node</i>	Bagaimana objek dibentuk /diawali.
4		<i>Activity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
5		<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran



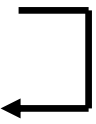
Sumber: (Kurniawan, 2018)

### 2.7.3 Sequence Diagram

*Sequence Diagram* digunakan untuk menunjukkan aliran fungsionalitas dalam *use case* yang disusun berdasarkan urutan waktu. *Sequence diagram*

menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudah, *sequence* diagram adalah gambaran tahap demi tahap yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram*. Berikut komponen-komponen yang ada pada *sequence* diagram.

**Tabel 2.3 Simbol Sequence Diagram**

SIMBOL	NAMA	KETERANGAN
	Objek	Menggambarkan objek/orang yang berintraksi di dalam sistem
	Stimulus	Menggambarkan pengiriman pesan
	Self Stimulus	Menyatakan suatu objek mengirimkan pesan untuk menjalankan operasi yang ada pada objek lain.

Sumber: (Kurniawan, 2018)

## 2.8 *Flowchart*

*Flowchart* digunakan dalam mendesain dan mendokumentasikan proses atau program sederhana. Seperti jenis diagram lainnya, diagram membantu memvisualisasikan apa yang sedang terjadi dan dengan demikian membantu memahami suatu proses, dan mungkin juga menemukan fitur-fitur yang kurang jelas dalam proses tersebut, seperti kekurangan dan hambatan. Ada berbagai jenis diagram alur: masing-masing jenis memiliki set kotak dan notasi sendiri. Dua jenis kotak yang paling umum dalam diagram alur adalah:

1. langkah pemrosesan, biasanya disebut aktivitas dan dilambangkan sebagai kotak persegi panjang.
2. keputusan biasanya dilambangkan sebagai berlian.

Diagram alir digambarkan sebagai "lintas fungsional" ketika bagan dibagi menjadi bagian *vertikal* atau *horizontal* yang berbeda, untuk menggambarkan kontrol unit organisasi yang berbeda. Simbol yang muncul di bagian tertentu berada dalam kendali unit organisasi itu. *Flowchart* lintas fungsional memungkinkan penulis untuk menemukan tanggung jawab untuk melakukan suatu tindakan atau membuat keputusan dengan benar, dan untuk menunjukkan tanggung jawab masing-masing unit organisasi untuk bagian berbeda dari satu proses tunggal.



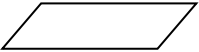
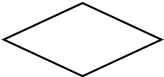
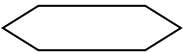
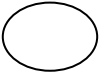

Diagram alir menggambarkan aspek-aspek tertentu dari proses dan biasanya dilengkapi dengan jenis diagram lainnya. Misalnya, Kaoru Ishikawa, mendefinisikan diagram alir sebagai salah satu dari tujuh alat dasar kendali mutu, di sebelah *histogram*, diagram *Pareto*, lembar periksa, diagram kontrol, diagram sebab-akibat, dan diagram sebaran. Demikian pula, di *UML*, notasi pemodelan konsep standar yang digunakan dalam pengembangan perangkat lunak, diagram aktivitas, yang merupakan jenis diagram alur, hanyalah salah satu dari banyak jenis diagram yang berbeda.

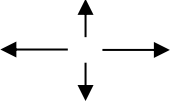


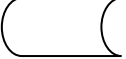

Diagram *Nassi-Shneiderman* dan *Drakon-chart* adalah notasi alternatif untuk aliran proses. Nama alternatif umum termasuk diagram alir, diagram alur proses, diagram alur fungsional, peta proses, diagram proses, diagram proses fungsional, model proses bisnis, model proses, diagram alir proses, diagram alur

kerja, diagram alir bisnis. Istilah "diagram alur" dan "diagram alir" digunakan secara bergantian (Nakatsu, 2019).

Struktur grafik yang mendasari diagram alur adalah grafik aliran, yang mengabstraksi jenis simpul, isinya, dan informasi tambahan lainnya. Adapun simbol-simbol *flowchart* lihat pada tabel sebagai berikut :

**Tabel 2.4 Simbol Flowchart**

NO	SIMBOL	FUNGSI
1.		Terminal, untuk memulai atau mengakhiri suatu program
2.		Proses, suatu simbol yang menunjukkan setiap pengolahan yang dilakukan.
3.		Input-Output, untuk memasukkan menunjukkan hasil dari suatu proses
4.		Decision, suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan
5.		Preparation, suatu symbol yang menyediakan tempat pengolahan
6.		Connector, suatu prosedur penghubung yang akan masuk atau keluar melalui symbol ini dalam lembar yang sama
7.		Off-Page Connector, merupakan symbol masuk atau keluarannya suatu prosedur pada lembaran kertas lainnya

8.		Arus/Flow, dari pada prosedur yang dapat dilakukan atas ke bawah dari bawah ke atas, ke atas dari kiri ke kanan ataupun dari kanan ke kiri
9.		Predefined Process, untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur
10.		Simbol untuk output, yang ditunjukkan ke suatu device, seperti printer, dan sebagainya
11		Penyimpanan file secara sementara
12		Menunjukkan input / Output Hardisk (media penyimpanan)

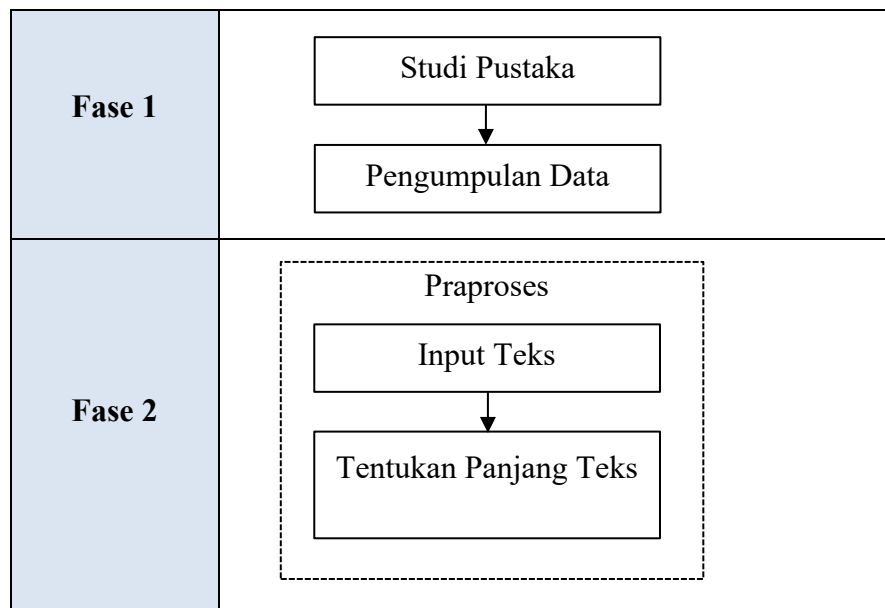
Sumber: (Kurniawan, 2018)

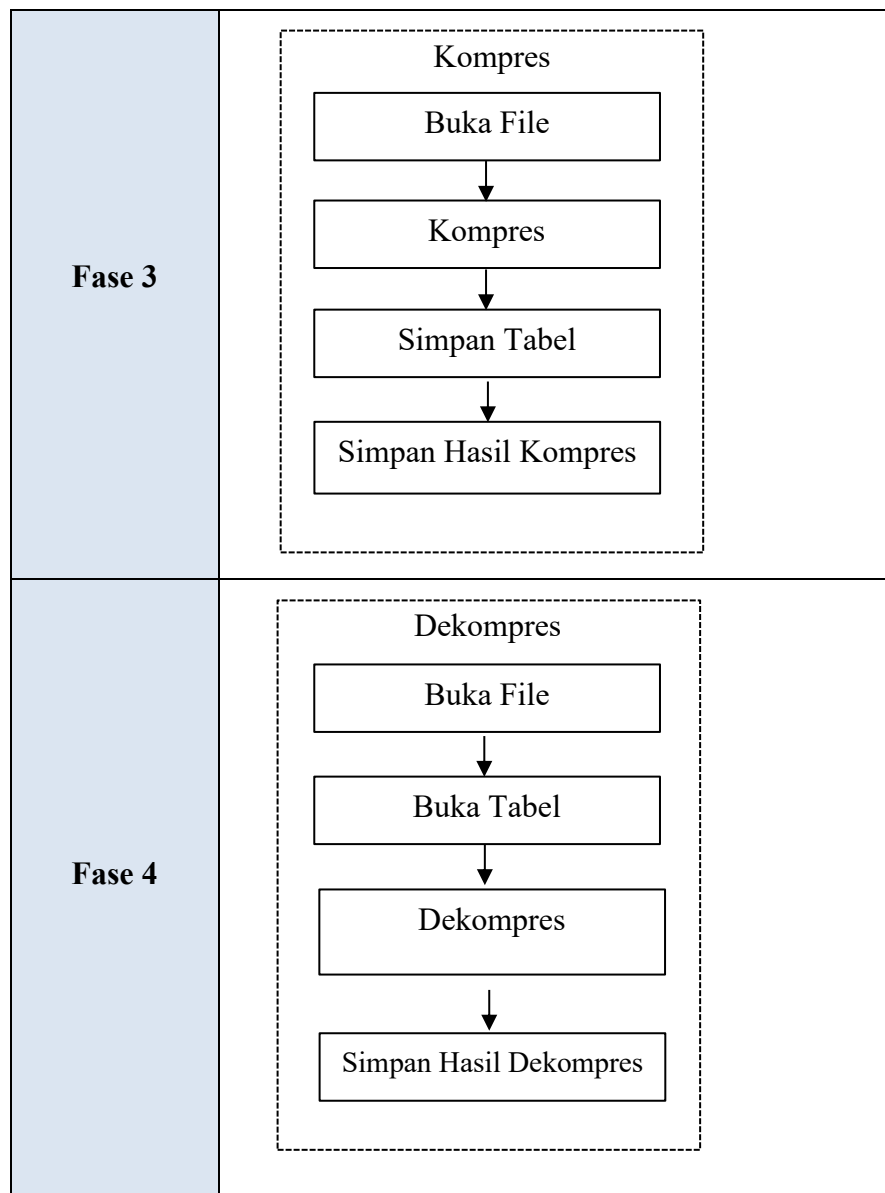
## BAB III

### METODE PENELITIAN

#### 3.1 Tahapan Penelitian

Dalam melakukan penelitian yang berhubungan dengan kompresi data, ada beberapa tahapan penelitian yang dilakukan oleh penulis agar sistem yang dibuat tidak mengalami kegagalan. Tahapan penelitian ini terdiri dari empat *fase* yang terpisah dimana setiap *fase* memiliki fungsi dan kegunaannya masing-masing. Gambar 3.1 adalah tahapan penelitian yang penulis lakukan.





**Gambar 3.1 Kerangka Penelitian**



Berikut adalah penjelasan dari Tahapan-Tahapann dari Fase 1 sebagai berikut :

1. Studi Pustaka

Studi pustaka ialah sumber dari perpustakaan buku dan jurnal yang berkaitan dengan Judul Ini.

2. Pengumpulan Data

Teknik pengumpulan data menggunakan wawancara dengan narasumber ataupun dosen pembimbing yang berkaitan dengan judul Implementasi Teknik Kompresi Dengan *Algoritma Squitur* Pada Pesan Teks.

Berikut adalah penjelasan dari Tahapan-Tahapann dari Fase 2 sebagai berikut :

1. Praproses

Untuk terciptanya aplikasi Implementasi Teknik Kompresi Dengan *Algoritma Squitur* Pada Pesan Teks ini ialah menggunakan *Microsoft Visual Studio 2010*

2. Input Teks

Untuk input teks nya ialah menggunakan kalimat LIKA LIKU LAKI LAKI TAK LAKU LAKU.

3. Tentukan Panjang Teks

Panjang teks untuk Kalimat LIKA LIKU LAKI LAKI TAK LAKU LAKU maksimal panjang nya 1000 Karakter.

Berikut adalah penjelasan dari Tahapan-Tahapann dari Fase 3 sebagai berikut :

1. Kompres

Kompres adalah untuk menghasil kan hasil yang dari plainteks, Setelah kalimat yang dari plainteks itu sudah di kompres maka hasil nya dapat dilihat dari riwayat.

2. Lalu simpan hasil kompres.

Berikut adalah penjelasan dari Tahapan-Tahapann dari Fase 4 sebagai berikut :

1. Dekompres

Dekompres adalah hasil dari yang sudah di kompreskan dari plainteks.

Berikut adalah tahapan penelitian yang dilakukan:

1. *Studi Literatur*

*Studi literatur* dilakukan untuk mendapatkan teori-teori tentang ilmu kompres data terutama algoritma *Sequitur*. Studi ini dapat dilakukan melalui buku dan informasi yang ada di internet.

2. Analisa

Bagian ini menjelaskan proses analisa permasalahan dan bagaimana permasalahan dapat diselesaikan dengan baik. Analisa akan memeriksa kebenaran dari rancangan yang akan dibuat.

3. Pembahasan

Bagian ini membahas tentang formula yang digunakan oleh pengguna mengenai algoritma *Sequitur* untuk melakukan proses kompres terhadap pesan teks.

4. Implementasi dan pengujian

Bagian ini dilakukan pengujian kebenaran *output* yang dihasilkan oleh program aplikasi Microsoft Visual Basic.Net 2010. Hasil akan diujicoba berdasarkan algoritma *Sequitur* yang telah ditetapkan dalam melakukan perhitungan.

### 3.2 Pengumpulan Data

Penelitian ini menggunakan data-data yang berupa teks-teks yang *diinputkan* langsung melalui teks sehingga data-data yang digunakan dapat berupa data apa saja yang berbentuk teks. Tujuan penggunaan data untuk melakukan

kompresi sehingga data tersebut lebih kecil dari sebelumnya. Data yang digunakan adalah data yang memiliki banyak kemiripan suku kata sehingga rasio kompresi dapat ditingkatkan

### **3.3 Analisa Sistem Yang Berjalan**

Pengiriman informasi atau data yang tidak menggunakan teknik kompresi akan sangat mempengaruhi *bandwidth* dan kecepatan *transfer*. Teknik konvensional dilakukan tidak menggunakan kompresi sehingga memperlambat proses *transfer* data dari pengirim ke penerima sehingga diperlukan suatu teknik yang baik dalam melakukan perkecilan jumlah data.

### **3.4 Rancangan Penelitian**

Perancangan penelitian adalah bagaimana suatu penelitian dimodelkan dalam suatu alur atau *diagram*. Banyak cara yang dapat dilakukan untuk merancang penelitian agar lebih terarah dan terstruktur. Perancangan ini membutuhkan ketelitian yang tinggi agar tidak menyalahi aturan yang ada. Hal ini bertujuan agar program aplikasi yang telah dibuat dapat bekerja secara efisien dan efektif. Desain penelitian dapat digambarkan dengan bentuk *Unified Modelling Language (UML)*. Pada *UML*, setiap arah penelitian tergambar dengan jelas dan terstruktur. Hal ini dapat memudahkan peneliti dalam menghasilkan *output* yang benar.

Perancangan penelitian yang menghasilkan batas kesalahan terkecil dalam penelitian disebut sebagai hasil perancangan yang terbaik. Perancangan penelitian

berikut ini berfungsi untuk mendefinisikan setiap tahapan untuk melengkapi kegiatan kerja pengguna terhadap rancangan penelitian yang akan dilaksanakan.

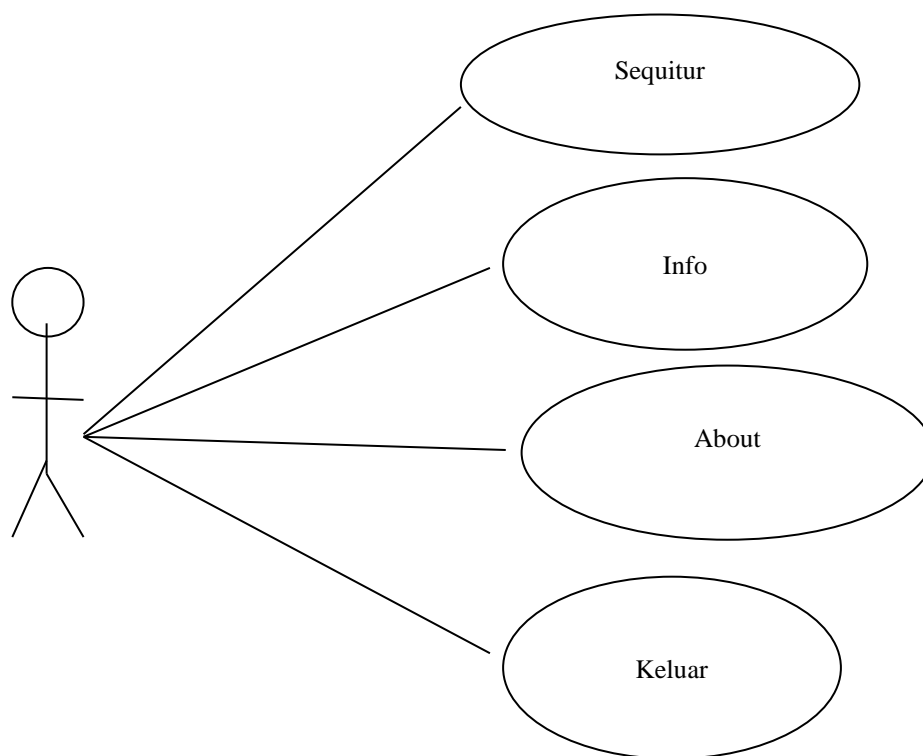
### **3.4.1 Rancangan UML**

#### **1. Use Case Diagram**

*Use Case diagram* didefinisikan sebagai diagram yang menangkap fungsionalitas dan persyaratan sistem dalam *UML*. *Use-cases* adalah konsep inti dari pemodelan bahasa *Unified Modeling*. *Use Case* terdiri dari *use case*, orang, atau berbagai hal yang menggunakan fitur yang disebut sebagai aktor dan elemen yang bertanggung jawab untuk mengimplementasikan *use case*. *Use case diagram* menangkap perilaku dinamis dari sistem *live*. Ini memodelkan bagaimana entitas eksternal berinteraksi dengan sistem untuk membuatnya bekerja. *Use case diagram* bertanggung jawab untuk memvisualisasikan hal-hal eksternal yang berinteraksi dengan bagian dari sistem.

*Use case* digunakan untuk mewakili fungsionalitas tingkat tinggi dan bagaimana pengguna akan menangani sistem. Sebuah *use case* mewakili fungsionalitas yang berbeda dari suatu sistem, komponen, paket, atau kelas.

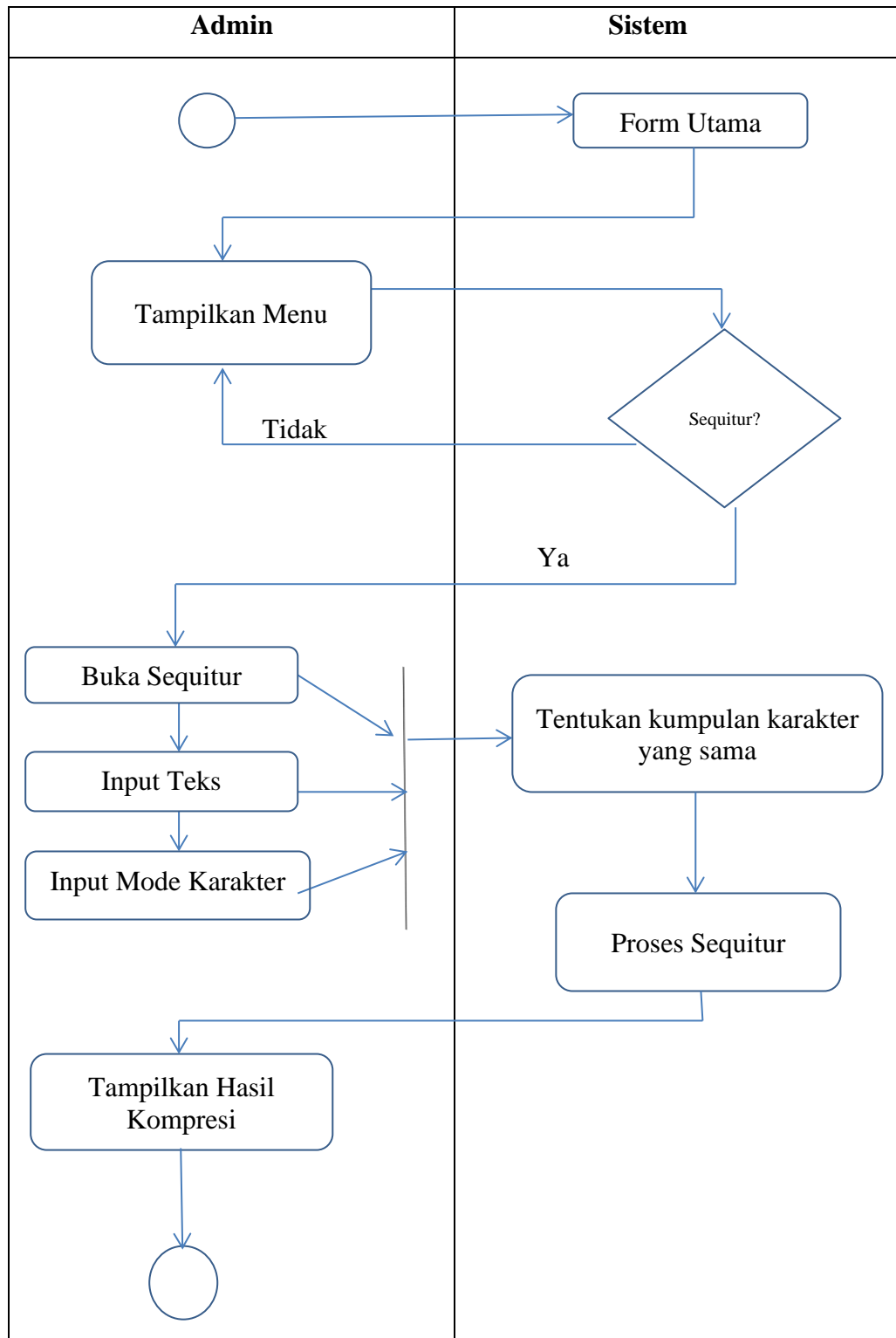
*Use Case* adalah penjelasan fungsi dari sebuah sistem dari segi pengguna. *Use Case* bekerja dengan cara menjelaskan interaksi antar *User* (pengguna) dengan sistemnya sendiri melalui sebuah bagan bagaimana suatu sistem dipakai. Gambar 3.2 adalah perancangan *Use Case* pada penelitian ini.



**Gambar 3.2 Use Case Diagram**

## **2. Activity Diagram**

*Activity diagram* menggambarkan perilaku alur kerja aktual suatu sistem dalam Teknologi Informasi. Diagram ini menggambarkan keadaan aktual dari suatu sistem dengan menunjukkan semua urutan kegiatan yang dilakukan. Juga, diagram ini dapat menunjukkan aktivitas yang kondisional atau *paralel*. Gambar 3.3 menjelaskan *Activity diagram* tersebut.

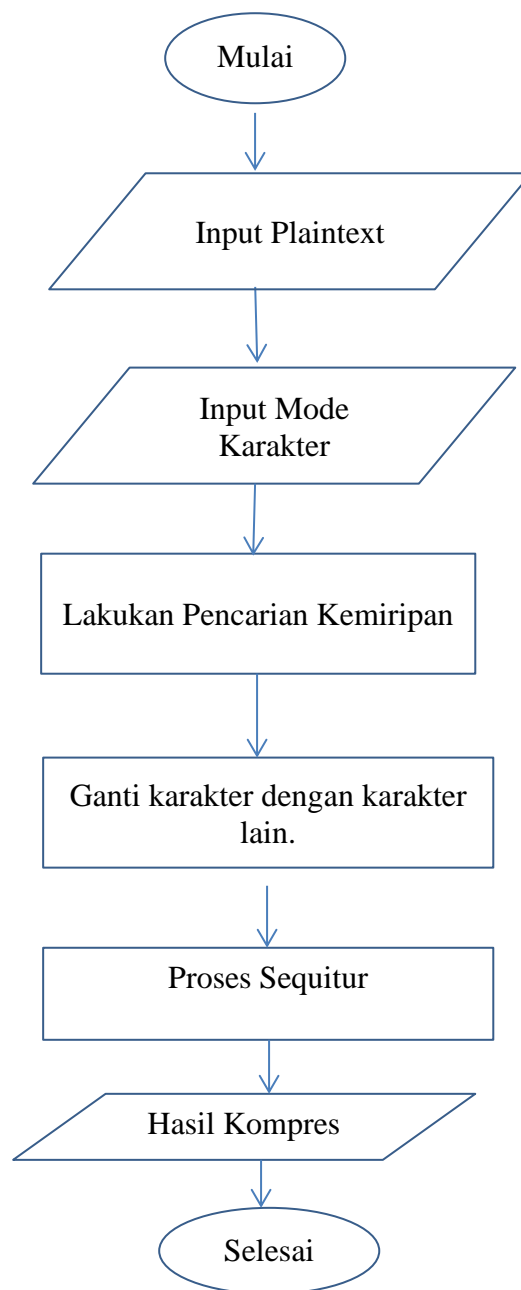


**Gambar 3.3 Activity Diagram**

### 3.4.2 Rancangan *Flowchart*

#### 1. *Flowchart* Kompres

*Flowchart* kompres akan menerangkan cara kerja algoritma *Sequitur* dalam mengkompres teks. *Flowchart* tersebut dapat dilihat pada gambar 3.4.

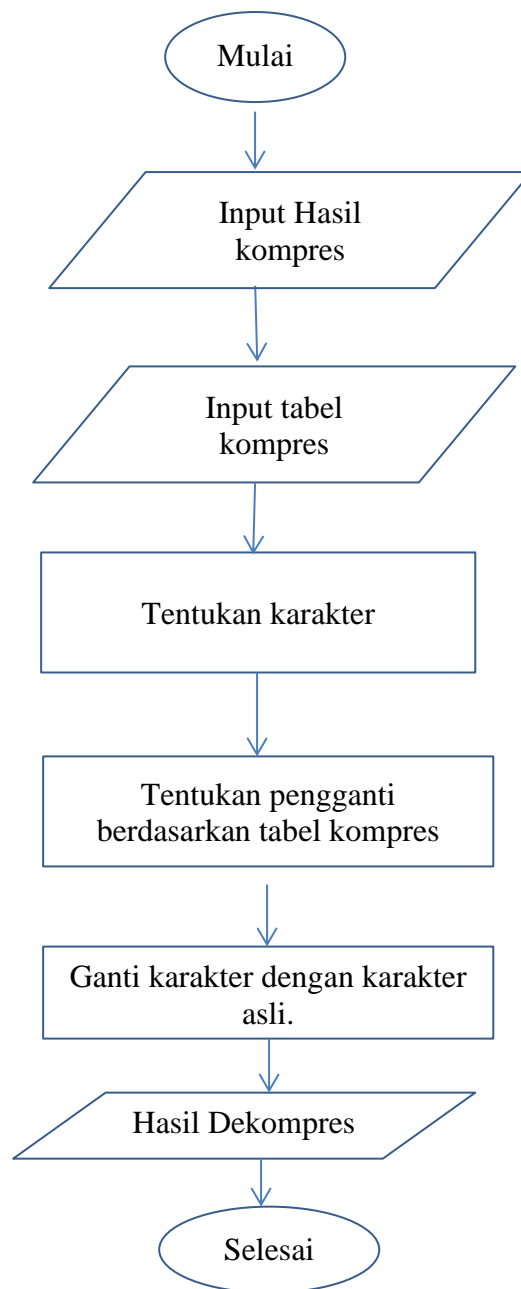


**Gambar 3.4** Flowchart proses kompresi



## 2. Flowchart Dekompres

*Flowchart* dekompres akan menjelaskan cara kerja algoritma Sequitur dalam mengekstrak kembali hasil kompres. *Flowchart* dekompres dapat dilihat pada gambar 3.5.



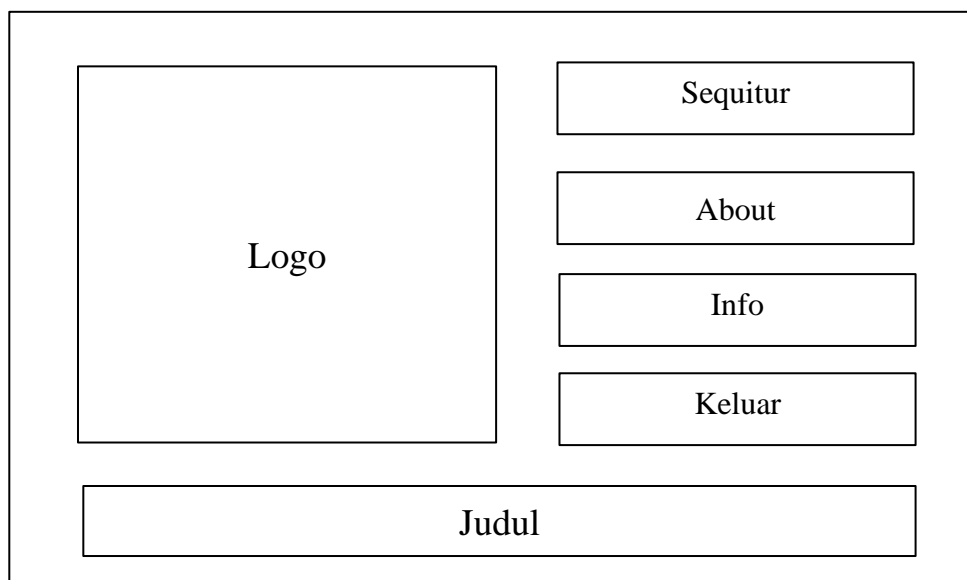
**Gambar 3.5** Flowchart proses dekompresi

### 3.4.3 Rancangan Tampilan Antarmuka

Rancangan tampilan antarmuka adalah perkiraan tampilan pada program aplikasi yang akan dibuat kemudian. Desain ini menunjukkan model tata letak objek-objek yang berpengaruh pada proses kompresi dan dekompresi. Program aplikasi akan dibuat menggunakan bahasa pemrograman Microsoft Visual Basic.Net 2010. Perancangan desain interface memiliki beberapa *sub desain* yang berhubungan antara satu sama lain.

#### 1. Menu Utama

Menu utama adalah tampilan yang pertama sekali ditampilkan pada saat program aplikasi dijalankan. Gambar 3.6 adalah hasil perancangan menu utama yang memiliki beberapa komponen lainnya.



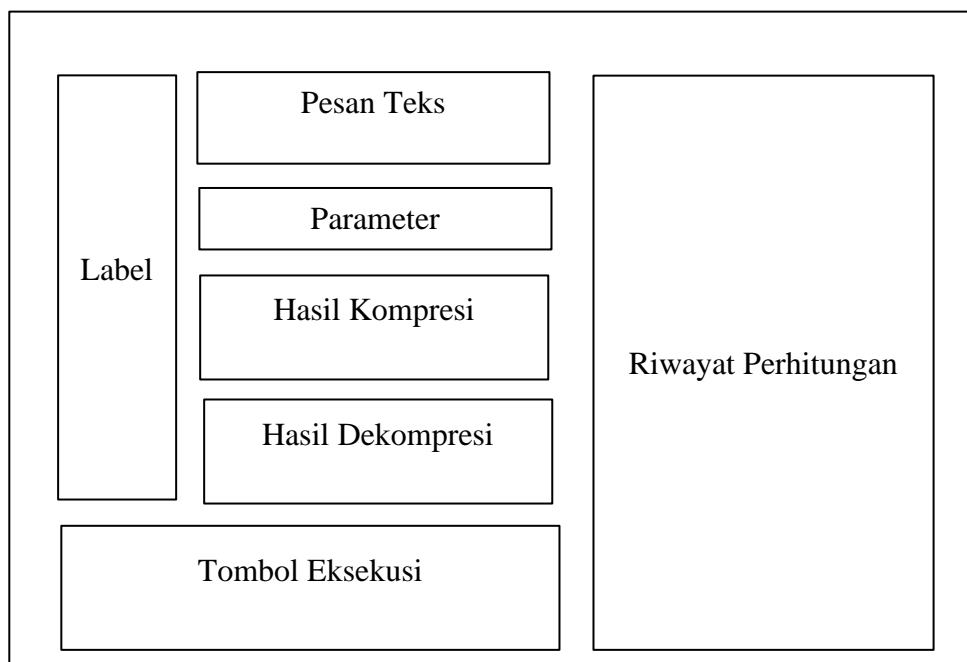
**Gambar 3.6 Tampilan Menu Utama**

Tampilan ini memiliki berapa sub-menu antara lain:

1. Logo : untuk menampilkan logo institusi
2. Sequitur : untuk proses kompresi
3. About : untuk menampilkan biodata penulis
4. Info : untuk menampilkan info
5. Keluar : untuk keluar dari aplikasi
6. Judul : untuk menampilkan judul tugas akhir

## 2. Menu Sequitur

Menu Sequitur adalah perancangan program aplikasi utama yang berfungsi melakukan kompresi dan dekompresi. Tampilan ini memiliki beberapa komponen yaitu textbox, button dan label. Gambar 3.6 adalah tampilan menu ini.



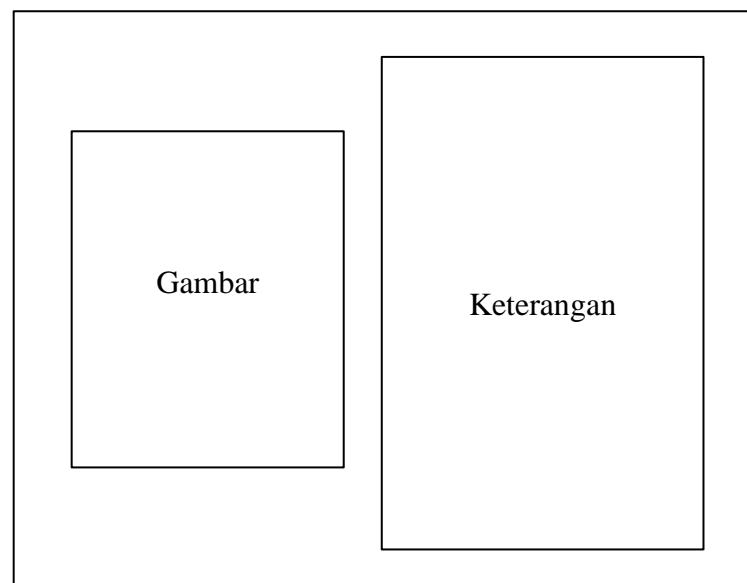
**Gambar 3.7 Tampilan Menu Sequitur**

Tampilan algoritma Sequitur memiliki beberapa bagian antara lain:

1. Label : untuk memberi keterangan objek
2. Pesan Teks : untuk memasukkan pesan yang akan dikompres
3. Parameter : variabel pengatur kompresi
4. Hasil Kompresi : hasil pemampatan pesan
5. Hasil Dekompresi : hasil ekstraksi pesan terkompresi
6. Tombol Eksekusi : tombol perintah untuk melakukan kompresi
7. Riwayat Perhitungan : proses perhitungan kompresi dan dekompresi

### 3. Menu Info

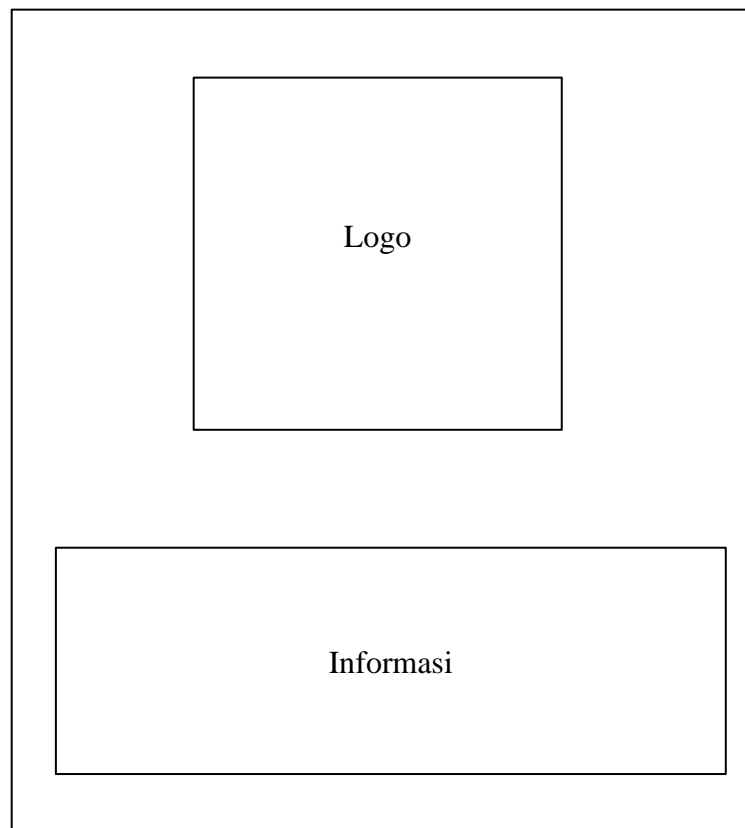
Menu ini menampilkan informasi tentang algoritma Sequitur. Tampilan ini terdiri dari objek gambar dan keterangan. Gambar 3.8 adalah hasil perancangan menu Info.



**Gambar 3.8 Tampilan Menu Info**

#### 4. Menu About

Menu ini menampilkan informasi penulis dan institusi dimana penulis melakukan penelitian. Tampilan ini terdiri dari logo Universitas Pembangunan Panca Budi dan biodata. Gambar 3.9 adalah hasil tampilan dari menu About.



**Gambar 3.9 Tampilan Menu About**

## **BAB IV**

### **HASIL DAN PEMBAHASAN**

Implementasi adalah pelaksanaan, pelaksanaan, atau praktik rencana, metode, atau desain, ide, model, spesifikasi, standar, atau kebijakan apa pun untuk melakukan sesuatu. Dengan demikian, implementasi adalah tindakan yang harus mengikuti pemikiran awal apa pun agar sesuatu benar-benar terjadi. Dalam konteks teknologi informasi, implementasi perangkat lunak atau perangkat keras mencakup semua proses punjaval yang terlibat dalam sesuatu yang beroperasi dengan baik di lingkungannya, termasuk menganalisis persyaratan, *instalasi*, *konfigurasi*, penyesuaian, berjalan, pengujian, integrasi sistem, pelatihan pengguna, pengiriman dan pembuatan yang diperlukan. perubahan.

#### **4.1 Spesifikasi Minimum *Hardware* dan *Software***

Spesifikasi sistem menjelaskan persyaratan operasional dan kinerja suatu sistem, seperti komputer. Ini dianggap sebagai dokumen tingkat tinggi yang menentukan fungsi global. Spesifikasi sistem membantu untuk menentukan pedoman operasional dan kinerja untuk suatu program aplikasi. Spesifikasi sistem dapat menguraikan bagaimana sistem diharapkan untuk melakukan, dan apa yang mungkin termasuk. Spesifikasi utama dapat mencakup definisi antarmuka, aturan desain dokumen, dan area fungsional. Spesifikasi dapat menentukan akses keamanan. Spesifikasi pada penelitian ini terdiri dari perangkat keras dan lunak.

#### 4.1.1 Spesifikasi Perangkat Keras

Penerapan algoritma Sequitur untuk melaksanakan kompresi dan dekompresi membutuhkan perangkat keras untuk menjalankan program aplikasi. Hal ini sebagai sarana pendukung utama. Tabel 4.1 adalah spesifikasi perangkat keras yang digunakan pada penelitian ini.

**Tabel 4.1 Spesifikasi perangkat keras**

No.	Komponen	Spesifikasi
1	Processor	Intel Core i3 2.2 GHz
2	RAM	4096 MB
3	Storage	500 GB
4	Display	14 inch

#### 4.1.2 Spesifikasi Perangkat Lunak

Tahap spesifikasi perangkat lunak memiliki tujuan, *deskripsi* kebutuhan dan persiapan validasi aplikasi perangkat lunak. *Deskripsi* kebutuhan memunculkan *file* spesifikasi aplikasi perangkat lunak. Kebutuhan akan perangkat lunak sebagai sarana *non-fisik* sangat mendukung hasil keluaran. Tabel 4.2 adalah spesifikasi perangkat lunak yang digunakan pada penelitian ini.

**Tabel 4.2 Spesifikasi perangkat lunak**

No.	Komponen	Spesifikasi
1	Sistem Operasi	Windows 10 64 Bit
2	IDE Pemrograman	Microsoft Visual Basic.NET 2010
3	Tangkap Gambar	Snipping Tool
4	Data Editor	Microsoft Excel

## **4.2 Hasil Antarmuka**

Hasil desain merupakan hasil pekerjaan untuk menerapkan perancangan menjadi sesuatu yang dapat digunakan. Proses ini terdiri dari langkah-langkah dalam menampilkan menu-menu yang berhubungan dengan program aplikasi. Agar proses implementasi berhasil, setiap menu harus memiliki hubungan yang sinkron dalam menjalankan perintah yang diberikan pada program aplikasi tersebut. Penelitian ini menggunakan program aplikasi yang telah dirancang dengan baik dan memiliki hasil antarmuka yang cukup ramah digunakan sehingga mempermudah pengguna dalam melakukan navigasi pada program aplikasi tersebut. Ada beberapa bagian yang akan dilakukan atau ditampilkan pada hasil antarmuka. Antar muka memiliki bagian utama dan bagian lainnya dimana bagian utama adalah program yang akan bekerja dalam melaksanakan kompresi dan dekompresi,

### **4.2.1 Halaman Menu Utama**

Halaman menu utama menampilkan serangkaian tombol-tombol untuk melakukan navigasi kepada menu-menu yang lain. Pada menu utama, terdapat empat buah tombol yaitu:

1. Tombol Sequitur
2. Tombol About
3. Tombol Info
4. Tombol Keluar



Masing-masing tombol memiliki fungsi dan kegunaannya masing-masing.

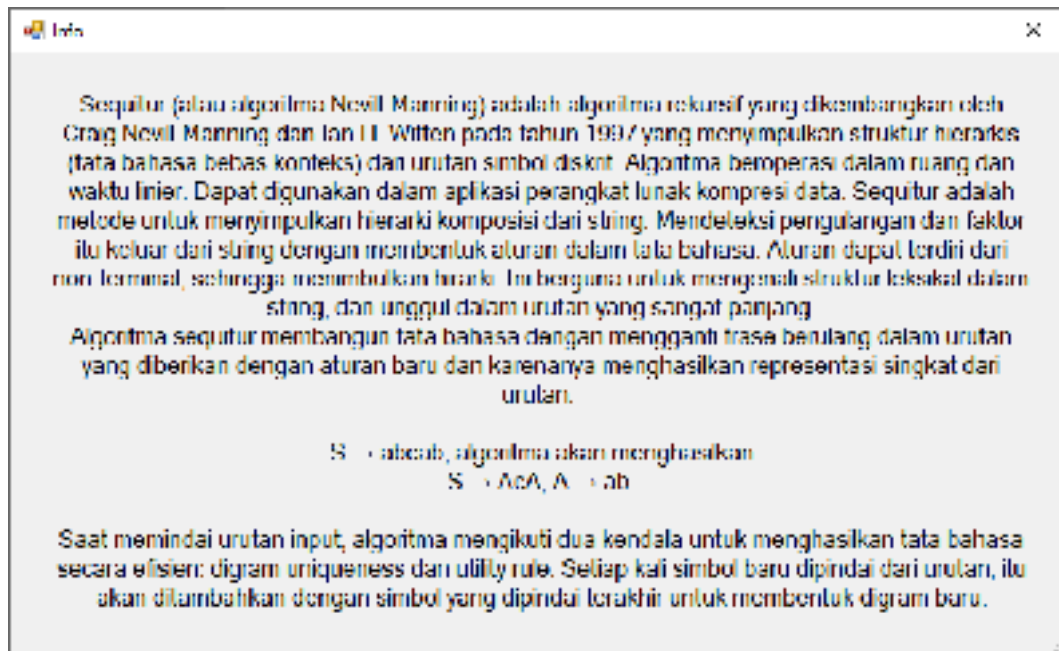
Gambar 4.1 adalah hasil tampilan menu utama.



**Gambar 4.1 Halaman Menu Utama**

#### **4.2.2 Halaman Info**

Halaman info adalah menu yang menampilkan informasi singkat tentang sejarah algoritma *Sequitur* dan penggunaan algoritma *Sequitur* tersebut. Gambar 4.2 adalah hasil tampilan dari halaman info.



**Gambar 4.2 Halaman Info**

### 4.2.3 Halaman About

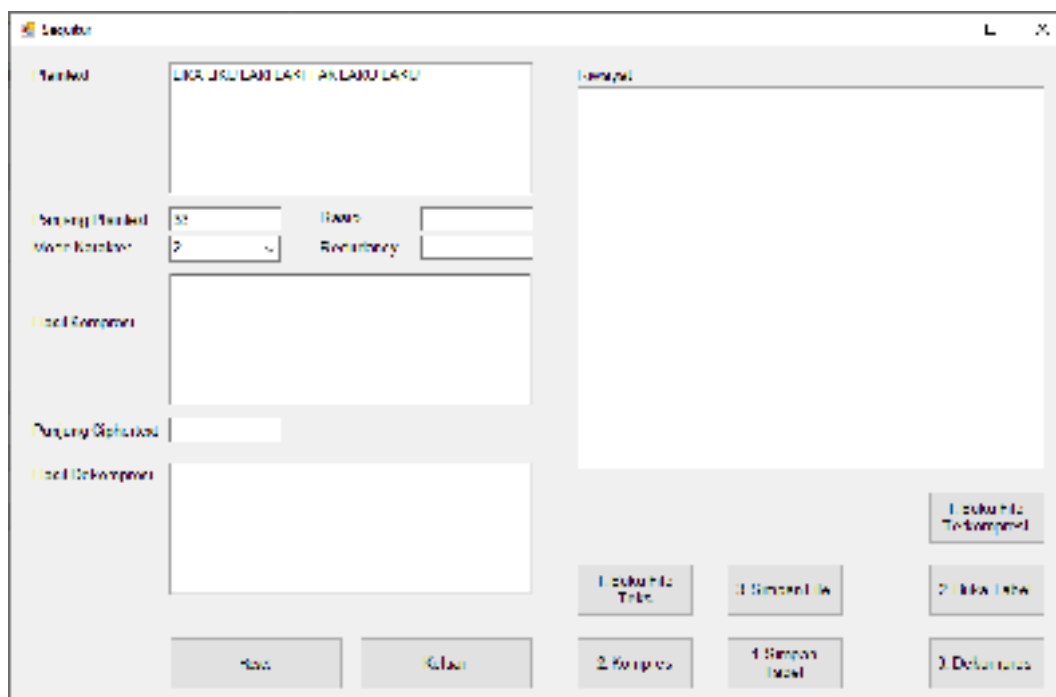
Halaman about menampilkan status dari penulis dalam lingkungan universitas. Form ini memiliki sebuah objek label dan *picturebox*. Gambar 4.3 adalah tampilan dari halaman About.



**Gambar 4.3 Halaman About**

#### **4.2.4 Halaman Sequitur**

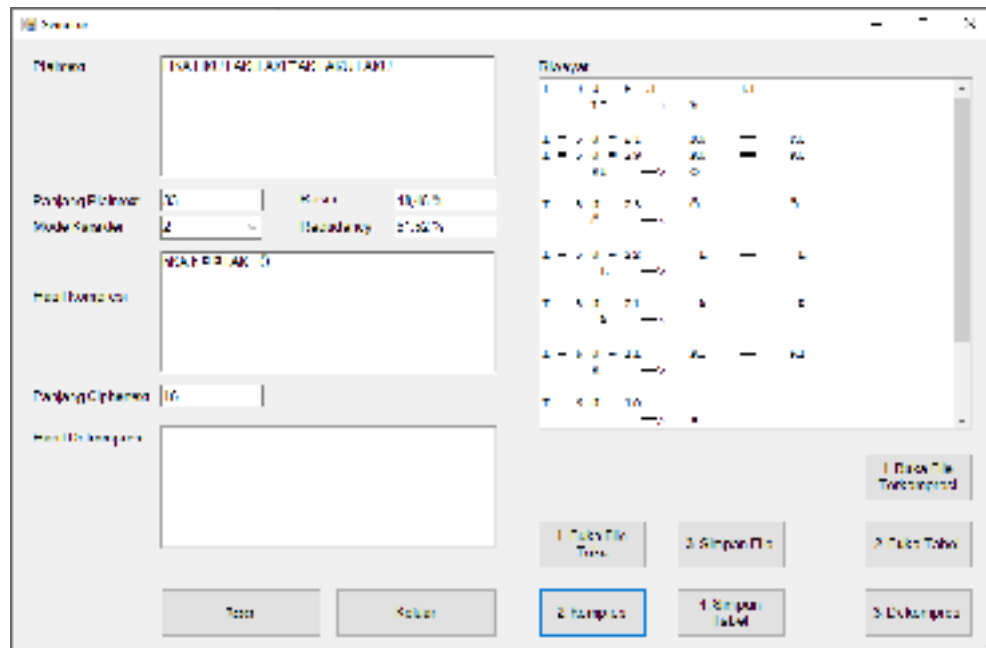
Halaman ini adalah bagian utama dari program aplikasi yang telah diciptakan. Halaman ini berupa proses *kompresi* dan *dekompresi* pada pesan teks yang akan di proses. Bagian ini terdiri dari tiga buah *textbox* penting yang menampung *plaintext*, hasil kompresi dan hasil dekompresi. *Input* dilakukan pada *plaintext* sementara proses kompresi dan dekompresi akan memberikan *output* pada *textbox* hasil kompresi dan hasil dekompresi. Gambar 4.4 adalah hasil tampilan dari halaman *Sequitur*.



**Gambar 4.4 Halaman Sequitur**

#### 4.2.5 Hasil Kompresi

Bagian ini berisi tentang hasil proses kompresi dari *plaintext* yang dimasukkan pada *textbox plaintext*. Selain *plaintext*, ada input lain yang menentukan pencarian kesamaan karakter yaitu mode karakter. *Mode* karakter berfungsi untuk menentukan apakah karakter yang akan diperiksa berjumlah 2, 3, atau dalam jumlah tertentu. *Mode* ini akan menentukan seberapa besar rasio dari kompresi tersebut. *Rasio* merupakan *persentase* kompresi sementara *Redudancy* adalah jumlah *persentase* karakter yang terbuang. Gambar 4.5 adalah tampilan dari hasil perhitungan dan proses kompresi menggunakan algoritma *Sequitur* dengan *Plaintext* = LIKA LIKU LAKI LAKI TAK LAKU LAKU dan Mode Karakter = 2.



Gambar 4.5 Hasil kompresi algoritma Sequitur

### Perhitungan Manual Kompresi

$I = 0 \quad J = 5 \quad LI \quad == \quad LI$   
 $LI \quad ==> \quad h$

$I = 5 \quad J = 24 \quad KU \quad == \quad KU$   
 $I = 5 \quad J = 29 \quad KU \quad == \quad KU$   
 $KU \quad ==> \quad \ddot{O}$

$I = 5 \quad J = 23 \quad \ddot{O} \quad == \quad \ddot{O}$   
 $\ddot{O} \quad ==> \quad \text{``}$

$I = 5 \quad J = 22 \quad \text{``}L \quad == \quad \text{``}L$   
 $\text{``}L \quad ==> \quad \bullet$

$I = 5 \quad J = 21 \quad \bullet A \quad == \quad \bullet A$   
 $\bullet A \quad ==> \quad \text{``}$

$I = 6 \quad J = 11 \quad KI \quad == \quad KI$   
 $KI \quad ==> \quad \bullet$

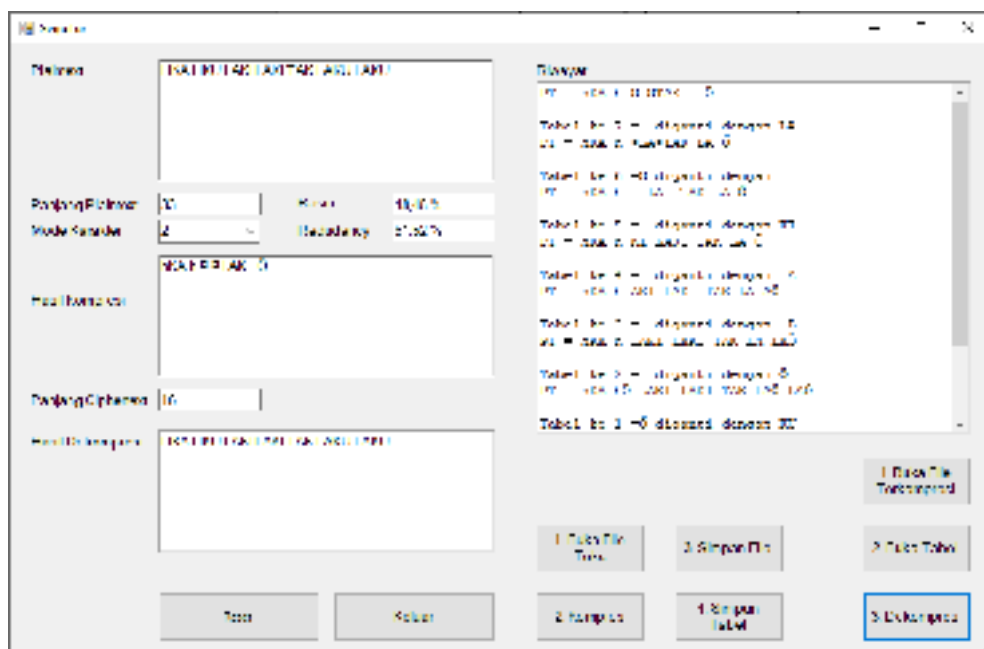
$I = 6 \quad J = 10 \quad \bullet \quad == \quad \bullet$   
 $\bullet \quad ==> \quad \alpha$

$I = 7 \quad J = 14 \quad LA \quad == \quad LA$   
 $LA \quad ==> \quad \bullet$

Hasil kompresi adalah hKA h $\text{``}\alpha\bullet\alpha$ TAK  $\bullet\text{``}\ddot{O}$

#### 4.2.6 Hasil Dekompresi

Setelah proses kompresi, hasil kompresi merupakan teks yang tidak dapat terbaca lagi, tetapi teks ini memiliki jumlah karakter yang lebih sedikit dari teks aslinya. Teks ini disebut dengan ciphertext. Ciphertext adalah teks hasil konversi yang sudah tidak dapat dipahami lagi tetapi masih dapat dibaca. Tetapi untuk dapat dibaca dan dipahami kembali, ciphertext atau hasil kompresi tersebut tersebut harus dikembalikan kembali ke plaintext dengan proses dekomposisi. Dekompresi harus menggunakan mode karakter dan tabel data yang sama sehingga dapat menghasilkan karakter yang sama persis dengan seperti sebelum dilakukan kompresi. Gambar 4.6 adalah tampilan dari hasil perhitungan proses dekomposisi pada karakter sebelumnya yang dihasilkan pada gambar 4.5.



Gambar 4.6 Hasil dekomposisi algoritma Sequitur

### Perhitungan Manual Dekompresi

PT = hKA h"α•αTAK •"Ö

Tabel ke 7 =• diganti dengan LA

PT = hKA h"αLAαTAK LA"Ö

Tabel ke 6 =α diganti dengan •

PT = hKA h"• LA• TAK LA"Ö

Tabel ke 5 =• diganti dengan KI

PT = hKA h"KI LAKI TAK LA"Ö

Tabel ke 4 =" diganti dengan •A

PT = hKA h•AKI LAKI TAK LA•AÖ

Tabel ke 3 =• diganti dengan "L

PT = hKA h"LAKI LAKI TAK LA"LAÖ

Tabel ke 2 =" diganti dengan Ö

PT = hKA hÖ LAKI LAKI TAK LAÖ LAÖ

Tabel ke 1 =Ö diganti dengan KU

PT = hKA hKU LAKI LAKI TAK LAKU LAKU

Tabel ke 0 =h diganti dengan LI

PT = LIKA LIKU LAKI LAKI TAK LAKU LAKU

Hasil dekomposisi adalah LIKA LIKU LAKI LAKI TAK LAKU LAKU

### **4.3 Test Perhitungan**

Tes perhitungan dirancang untuk mengukur kemampuan program aplikasi *sequitur* dalam melakukan kompres dan *dekompres* pada teks tertentu. *Mode* karakter akan menghasilkan hasil kompresi yang berbeda karena *mode* karakter menentukan seberapa banyak karakter yang akan dicek kemiripannya dalam suatu deretan *plaintext*. Pengujian ini dilakukan untuk melihat seberapa besar *rasio* dan *redundancy* dari proses kompresi pada program aplikasi yang diciptakan dan apakah sesuai dengan perhitungan yang dilakukan secara manual. Proses yang dilakukan terdiri dari dua proses, yaitu proses kompresi dan proses dekomposisi . Berikut ini adalah penjelasan dan perhitungan lengkap proses *kompresi* dan *dekomposisi* pada

algoritma Sequitur dengan memberikan sebuah *plaintext* dan dua buah mode karakter.

### Pengujian Pertama

*Plaintext* = LIKA LIKU LAKI LAKI TAK LAKU LAKU

Mode Karakter = 2

### Hasil Kompresi

I = 0 J = 5	LI	==	LI
	LI ==>		h
I = 5 J = 24	KU	==	KU
I = 5 J = 29	KU	==	KU
	KU ==>		Ö
I = 5 J = 23	Ö	==	Ö
	Ö ==>		"
I = 5 J = 22	"L	==	"L
	"L ==>		•
I = 5 J = 21	•A	==	•A
	•A ==>		"
I = 6 J = 11	KI	==	KI
	KI ==>		•
I = 6 J = 10	•	==	•
	• ==>		α
I = 7 J = 14	LA	==	LA
	LA ==>		•

Jumlah Sama : 9

Hasil kompresi adalah \_KA\_ \_üHüTAK H\_Ø

### Hasil Dekompresi

PT = hKA h"α•αTAK •"Ö

Tabel ke 7 =• diganti dengan LA



PT = hKA h"LA" TAK LA"

Tabel ke 6 =x diganti dengan •  
PT = hKA h"• LA• TAK LA"

Tabel ke 5 =• diganti dengan KI  
PT = hKA h"KI LAKI TAK LA"

Tabel ke 4 =" diganti dengan •A  
PT = hKA h•AKI LAKI TAK LA•A

Tabel ke 3 =• diganti dengan "L  
PT = hKA h"LAKI LAKI TAK LA"L

Tabel ke 2 =" diganti dengan "L  
PT = hKA h" LAKI LAKI TAK LA" LA"

Tabel ke 1 =L diganti dengan KU  
PT = hKA hKU LAKI LAKI TAK LAKU LAKU

Tabel ke 0 =h diganti dengan LI  
PT = LIKA LIKU LAKI LAKI TAK LAKU LAKU

Hasil dekompresi adalah LIKA LIKU LAKI LAKI TAK LAKU LAKU

## Pengujian Kedua

Plaintext = LIKA LIKU LAKI LAKI TAK LAKU LAKU

Mode Karakter = 3

## Hasil Kompresi

I = 0 J = 5 LIK == LIK  
LIK ==> -

I = 4 J = 23 U L == U L  
U L ==> ;

I = 4 J = 21 ;AK == ;AK  
;AK ==> '

I = 6 J = 15 LA == LA  
LA ==> ;

Jumlah Sama : 4

Hasil kompresi adalah `_A_'I;KI TAK;K'U`

### Hasil Dekompresi

PT = `_A_'I;KI TAK;K'U`

Tabel ke 3 = ; diganti dengan LA  
 PT = `_A_'I LAKI TAK LAK'U`

Tabel ke 2 = ' diganti dengan ;AK  
 PT = `_A_';AKI LAKI TAK LAK;AKU`

Tabel ke 1 = ; diganti dengan U L  
 PT = `_A_U LAKI LAKI TAK LAKU LAKU`

Tabel ke 0 = \_ diganti dengan LIK  
 PT = `LIKA LIKU LAKI LAKI TAK LAKU LAKU`

Hasil dekompresi adalah `LIKA LIKU LAKI LAKI TAK LAKU LAKU`

## **BAB V**

### **PENUTUP**

#### **5.1 Kesimpulan**

Penulis dapat menarik beberapa kesimpulan berdasarkan hasil pengujian yang dilakukan setelah melakukan penelitian. Adapun kesimpulan yang diperoleh adalah antara lain:

1. Algoritma *Sequitur* dapat dilakukan untuk mengecilkan kapasitas pesan.
2. Algoritma *Sequitur* bekerja dengan mencari kumpulan pasangan karakter yang ada sehingga dapat digantikan dengan karakter tunggal.
3. Hasil kompresi akan memiliki *rasio* yang besar apabila pada pesan terdapat banyak kumpulan karakter yang sama dan memiliki *rasio* yang kecil apabila pada pesan terdapat sedikit kumpulan karakter yang sama.
4. Mode karakter menentukan besar *rasio* dan *redundancy* karena *mode* karakter menentukan seberapa banyak karakter yang akan diuji kemiripan dalam suatu deretan *plaintext*.

#### **5.2 Saran**

Penelitian juga memiliki kekurangan. Terdapat beberapa saran yang dapat penulis kemukakan untuk meningkatkan kualitas penelitian ini. Adapun saran tersebut adalah antara lain:

1. Sebaiknya program aplikasi dapat digunakan secara daring dan berbasis mobile.

2. Hendaknya algoritma *Sequitur* dapat dikombinasikan dengan algoritma lain sehingga dapat bekerja tanpa harus menggunakan kumpulan karakter yang sama.

## DAFTAR PUSTAKA

- Barone, L., Williams, J., & Micklos, D. (2017). Unmet needs for analyzing biological big data: A survey of 704 NSF principal investigators. *PLOS Computational Biology*, 13(10), e1005755. <https://doi.org/10.1371/journal.pcbi.1005755>
- Gurevich, Y. (2012). *What Is an Algorithm?* (pp. 31–42). [https://doi.org/10.1007/978-3-642-27660-6\\_3](https://doi.org/10.1007/978-3-642-27660-6_3)
- Hemendinger, D. (2019). *Data compression*. Britannica. <https://www.britannica.com/technology/data-compression>
- Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 5(1), 77. <https://doi.org/10.25126/jtiik.201851610>
- Ladjamudin, A.-B. bin. (2017). *Analisis dan Desain Sistem Informasi*. Graha Ilmu.
- Marlina, L., Siahaan, A. P. U., Kurniawan, H., & Sulistianingsih, I. (2017). Data Compression Using Elias Delta Code. *International Journal of Recent Trends in Engineering and Research*, 3(8), 210–217. <https://doi.org/10.23883/IJRTER.2017.3406.TEGS6>
- Nakatsu, R. T. (2019). *Reasoning with Diagrams : Decision-Making and Problem-Solving with Diagrams*. John Wiley & Sons.
- Rao, R. V., & Selvamani, K. (2015). Data Security Challenges and Its Solutions in Cloud Computing. *Procedia Computer Science*, 48, 204–209. <https://doi.org/10.1016/j.procs.2015.04.171>
- Suherman, & Siahaan, A. P. U. (2016). Huffman Text Compression Technique. *International Journal of Computer Science and Enginee Ring*, 3(8), 103–108.
- Sukmawati, R., & Priyadi, Y. (2019). Perancangan Proses Bisnis Menggunakan UML Berdasarkan Fit/Gap Analysis Pada Modul Inventory Odoo. *INTENSIF: Jurnal Ilmiah Penelitian Dan Penerapan Teknologi Sistem Informasi*, 3(2), 104. <https://doi.org/10.29407/intensif.v3i2.12697>
- Sun, Y., Zhang, J., Xiong, Y., & Zhu, G. (2014). Data Security and Privacy in Cloud Computing. *International Journal of Distributed Sensor Networks*, 10(7), 190903. <https://doi.org/10.1155/2014/190903>
- Supiyandi, & Frida, O. (2018). Analisis Perbandingan Pemampatan Data Teks Dengan Menggunakan Metode Huffman dan Half-Byte. *Jurnal Ilmu*

Technopedia. (2019). *Unified Modeling Language (UML)*. Technopedia.  
<https://www.techopedia.com/definition/3243/unified-modeling-language-uml>

Uml-diagrams.org. (2019). *Use case diagrams are UML diagrams describing units of useful functionality (use cases) performed by a system in collaboration with external users (actors)*. <https://www.uml-diagrams.org/use-case-diagrams.html>

UTM. (2019). *Concept: Use-Case Model*. Univesidad Technologica de La Mixteca.  
[http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use\\_case\\_model\\_CD178AF9.html](http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use_case_model_CD178AF9.html)

Wasserkrug, S., Dalvi, N., Munson, E. V., Gogolla, M., Sirangelo, C., Fischer-Hübner, S., Ives, Z., Velegarakis, Y., Bevan, N., Jensen, C. S., & Snodgrass, R. T. (2019). Unified Modeling Language. In *Encyclopedia of Database Systems* (pp. 3232–3239). Springer US. [https://doi.org/10.1007/978-0-387-39940-9\\_440](https://doi.org/10.1007/978-0-387-39940-9_440)

Zhang, D., Tsotras, V. J., Levialdi, S., Grinstein, G., Berry, D. A., Gouet-Brunet, V., Kosch, H., Döller, M., Döller, M., Kosch, H., Maier, P., Bhattacharya, A., Ljosa, V., Nack, F., Bartolini, I., Gouet-Brunet, V., Mei, T., Rui, Y., Crucianu, M., ... Pitoura, E. (2009). Indexed Sequential Access Method. In *Encyclopedia of Database Systems* (pp. 1435–1438). Springer US. [https://doi.org/10.1007/978-0-387-39940-9\\_738](https://doi.org/10.1007/978-0-387-39940-9_738)

Badawi, A. (2018). Evaluasi Pengaruh Modifikasi Three Pass Protocol Terhadap Transmisi Kunci Enkripsi.

Batubara, Supina. "Analisis perbandingan metode fuzzy mamdani dan fuzzy sugeno untuk penentuan kualitas cor beton instan." *IT Journal Research and Development* 2.1 (2017): 1-11.

Bahri, S. (2018). *Metodologi Penelitian Bisnis Lengkap Dengan Teknik Pengolahan Data SPSS*. Penerbit Andi (Anggota Ikapi). Percetakan Andi Offset. Yogyakarta.

Erika, Winda, Heni Rachmawati, and Ibnu Surya. "Enkripsi Teks Surat Elektronik (E-Mail) Berbasis Algoritma Rivest Shamir Adleman (RSA)." *Jurnal Aksara Komputer Terapan* 1.2 (2012).

Fitriani, W., Rahim, R., Oktaviana, B., & Siahaan, A. P. U. (2017). Vernam Encrypted Text in End of File Hiding Steganography Technique. *Int. J. Recent Trends Eng. Res*, 3(7), 214-219.

Hardinata, R. S. (2019). Audit Tata Kelola Teknologi Informasi menggunakan Cobit 5 (Studi Kasus: Universitas Pembangunan Panca Budi Medan). *Jurnal Teknik dan Informatika*, 6(1), 42-45.

Hariyanto, E., Lubis, S. A., & Sitorus, Z. (2017). Perancangan prototipe helm pengukur kualitas udara. *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)*, 1(1).

- Hariyanto, E., & Rahim, R. (2016). Arnold's cat map algorithm in digital image encryption. *International Journal of Science and Research (IJSR)*, 5(10), 1363-1365.
- Harumy, T. H. F., & Sulistianingsih, I. (2016). Sistem penunjang keputusan penentuan jabatan manager menggunakan metode mfep pada cv. Sapo durin. In *Seminar Nasional Teknologi Informasi dan Multimedia* (pp. 6-7).
- Iqbal, M., Siahaan, A. P. U., Purba, N. E., & Purwanto, D. (2017). Prim's Algorithm for Optimizing Fiber Optic Trajectory Planning. *Int. J. Sci. Res. Sci. Technol*, 3(6), 504-509.
- Marlina, L., Muslim, M., Siahaan, A. U., & Utama, P. (2016). Data Mining Classification Comparison (Naïve Bayes and C4. 5 Algorithms). *Int. J. Eng. Trends Technol*, 38(7), 380-383.
- Muttaqin, Muhammad. "ANALISA PEMANFAATAN SISTEM INFORMASI E-OFFICE PADA UNIVERSITAS PEMBANGUNAN PANCA BUDI MEDAN DENGAN MENGGUNAKAN METODE UTAUT." *Jurnal Teknik dan Informatika* 5.1 (2018): 40-43.
- Ramadhan, Z., Zarlis, M., Efendi, S., & Siahaan, A. P. U. (2018). Perbandingan Algoritma Prim dengan Algoritma Floyd-Warshall dalam Menentukan Rute Terpendek (Shortest Path Problem). *JURIKOM (Jurnal Riset Komputer)*, 5(2), 135-139.
- Rahim, R., Aryza, S., Wibowo, P., Harahap, A. K. Z., Suleman, A. R., Sihombing, E. E., ... & Agustina, I. (2018). Prototype file transfer protocol application for LAN and Wi-Fi communication. *Int. J. Eng. Technol.*, 7(2.13), 345-347.
- Wahyuni, Sri. "Implementasi Rapidminer Dalam Menganalisa Data Mahasiswa Drop Out." *Jurnal Abdi Ilmu* 10.2 (2018): 1899-1902.

## Listing Program

1.frmSequitur.vb

```
Imports System.IO
```

```
Public Class frmSequitur
```

```
    Public Log As String  
    Public Pjg_PT, Pjg_CT As Integer  
    Public PT, CT, PTB, CTB As String  
    Public PBytes(), CBytes() As Byte  
    Public Input, Output As String  
    Public JlhSama As Integer  
    Public Tabel As List(Of String)
```

```
    ' ===== FUNGSI UMUM  
    ===== '
```

```
    Public Function GenerateChar(ByVal PT As String) As Char
```

```
        Dim R As Char = ""  
        Dim Rand As Random = New Random
```

```
        Dim Ketemu As Boolean = True
```

```
        While Ketemu
```

```
            Ketemu = False  
            R = Convert.ToChar(Rand.Next(1, 255))  
            For i As Integer = 0 To PT.Count() - 1  
                If R = PT(i) Then  
                    Ketemu = True  
                    Exit For  
                End If  
            Next
```

```
        End While
```

```
        Return R
```

```
    End Function
```

```
    ' ===== SEQUITUR  
    ===== '
```

```
    Public Function Kompres(ByVal PT As String) As String
```

```
        Dim R As String = ""  
        Dim JlhSub As Integer  
        Dim Index As Integer = 1  
        Dim Ketemu As Boolean  
        Dim NT As Char
```

```
        Log = ""  
        JlhSub = cmbJH.Text  
        JlhSama = 0  
        R = PT  
        Tabel = New List(Of String)
```

```
        For i As Integer = 0 To R.Count - JlhSub  
            Ketemu = False
```



```

        For j As Integer = i + JlhSub To R.Count - JlhSub
            If R.Substring(i, JlhSub) = R.Substring(j, JlhSub) Then
                Log &= "I = " & i & vbTab & "J = " & j & vbTab &
R.Substring(i, JlhSub) & vbTab & "==" & vbTab & R.Substring(j, JlhSub) &
vbCrLf

                JlhSama += 1
                Ketemu = True

            End If

        Next

        If Ketemu Then
            NT = GenerateChar(R)
            Tabel.Add(NT & R.Substring(i, JlhSub))
            Log &= vbTab & R.Substring(i, JlhSub) & vbTab & "=>" &
vbTab & NT & vbCrLf & vbCrLf
            R = R.Replace(R.Substring(i, JlhSub), NT)
            Index += 1
            i = 0
            Log &= "hasil = " & R & vbCrLf & vbCrLf

        End If
    Next

    CT = R

    'Masukkan Ke CBytes
    ReDim CBytes(CT.Count() - 1)
    For i As Integer = 0 To CT.Count() - 1
        CBytes(i) = Convert.ToByte(CT(i))
    Next

    'Hitung Rasio dan Redudancy
    Pjg_PT = PT.Count()
    Pjg_CT = CT.Count()
    txtRasio.Text = Format(Pjg_CT / Pjg_PT * 100, "#,##") & " %"
    txtRed.Text = Format((Pjg_PT - Pjg_CT) / Pjg_PT * 100, "#,##") &
" %"

    txtLog.Text = Log

    Log += "Jumlah Sama : " & JlhSama & vbCrLf
    txtCT.Text = R
    txtPjg_CT.Text = R.Count().ToString()
    txtLog.Text = Log
    Return R
End Function

Public Function Dekompres(ByVal CT As String) As String
    Log = ""

    Pjg_CT = CT.Count()
    PT = CT.Trim()

    Log &= "PT = " & PT & vbCrLf & vbCrLf
    For i As Integer = Tabel.Count - 1 To 0 Step -1

        PT = PT.Replace(Tabel(i)(0), Tabel(i).Substring(1,
Tabel(i).Length - 1))

```

```

        Log &= "Tabel ke " & i & " =" & Tabel(i)(0) & " diganti dengan
" & Tabel(i).Substring(1, Tabel(i).Length - 1) & vbCrLf
        Log &= "PT = " & PT & vbCrLf & vbCrLf

    Next

    'Hitung Rasio dan Redudancy
    Pjg_PT = PT.Count()
    txtRasio.Text = Format(Pjg_CT / Pjg_PT * 100, "#,##") & " %"
    txtRed.Text = Format((Pjg_PT - Pjg_CT) / Pjg_PT * 100, "#,##") &
" %"
    txtDT.Text = PT
    txtPjg_PT.Text = Pjg_PT.ToString()
    txtLog.Text = Log

    Return CT
End Function

' ===== EVENT
HANDLER ===== '

Private Sub txtPT_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtPT.TextChanged
    txtPjg_PT.Text = txtPT.Text.Length
End Sub

Private Sub frmSequitur_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    txtPjg_PT.Text = txtPT.Text.Length
End Sub

Private Sub btnDekompres_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnDekompres.Click
    'CT = txtCT.Text
    Try
        Dekompres(CT)
    Catch ex As Exception

    End Try
End Sub

Private Sub btnKompres_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnKompres.Click
    PT = txtPT.Text
    If PT.Length > 1000 Then
        MessageBox.Show("Panjang karakter tidak boleh melebihi dari
1000 karakter")
    Exit Sub
    End If
    Try
        If cmbJH.Text <= 1 Then
            MessageBox.Show("Jumlah substring tidak boleh lebih kecil
dari 2", "Peringatan")
        Exit Sub
        End If
        Kompres(PT)
    Catch ex As Exception

```

```
        End Try
    End Sub

    Private Sub btnKeluar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnKeluar.Click
        Application.Exit()
    End Sub

    Private Sub cmbJH_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmbJH.SelectedIndexChanged

    End Sub
End Class
```