Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

Perancangan Aplikasi Kompresi File MP3 Dengan Menggunakan Algoritma Lempel Ziv Welch (LZW)

Tia Betsaida Situmorang

[1]Fakultas Ilmu Komputer & Teknologi Informasi, Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: tia03situmorang@gmail.com

Abstrak - File berbentuk suara banyak digunakan dalam kegiatan sehari-hari orang dalam mendukung aktifitasnya seperti penggunaan file suara dalam aplikasi pemutar musik baik secara daring maupun luring serta pada aplikasi *audio streaming*. Pemanfaatan teknik kompresi suara sangatlah di butuhkan mengingat panjangnya durasi suara dan banyak nya file suara dalam ruang penyimpanan sehingga membutuh ruang penyimpanan yang banyak pula. Teknik kompresi merupakan teknik dimana kapasitas secara fisik file suara dikurangi dengan mengusahakan tidak mengurangi kualitas suara yang terdapat dalam file audio. File suara dengan ekstensi MP3 merupakan ekstensi file suara yang banyak digunakan saat ini, yang memiliki kepanjangan MPEG *Audio Layer* III. Menangani permasalah ruangan penyimpanan dan sekaligus menghemat penggunaan paket data pada saat melakukan pengaksesan sekaligus transimis file suara, maka diterapkan metode kompresi. Metode kompresi yang digunakan adalah Algoritma Lempel Ziv Welch (LZW). Algoritma Lempel Ziv Welch adalah algoritma kompresi yang menjaga kualitas dan mampu mengidentifkasi dan mengganti pola berulang dalam data. Penelitian ini diharapkan bisa dimanfaatkan untuk pengurangan ukuran data suatu file gambar dan video. Pada tahap awal yang akan dianalisa penulis ialah dengan mencari file Mp3 yang akan dikompres kemudian diubah menjadi nilai hexadecimal melalui aplikasi HxD. Setelah didapat nilai hexadecimal file Mp3 maka akan dilakukan proses kompresi dan diperoleh file hasil kompresi. Hasil kompresi file audio dengan algoritma LZW diperoleh nilai Compression Ratio 16%.

Kata Kunci: Perancangan, Aplikasi, Kompresi, File, MP3, Lempel Ziv Welch

Abstract - Sound files are widely used in people's daily activities to support their activities, such as using sound files in music player applications both online and offline as well as in audio streaming applications. The use of sound compression techniques is very necessary considering the long duration of the sound and the large number of sound files in the storage space so that it requires a lot of storage space. Compression technique is a technique where the physical capacity of a sound file is reduced by trying not to reduce the sound quality contained in the audio file. Sound files with the MP3 extension are a sound file extension that is widely used today, which stands for MPEG Audio Layer III. To handle storage space problems and at the same time save on data packet usage when accessing and transmitting sound files, a compression method is applied. The compression method used is the Lempel Ziv Welch (LZW) algorithm. Ziv Welch's Lempel algorithm is a compression algorithm that maintains quality and is able to identify and replace repetitive patterns in data. It is hoped that this research can be used to reduce the data size of image and video files. In the initial stage, the author will analyze by looking for MP3 files which will be compressed and then converted into hexadecimal values via the HxD application. After obtaining the hexadecimal value of the Mp3 file, the compression process will be carried out and the compressed file will be obtained. The results of audio file compression with the LZW algorithm obtained a Compression Ratio value of 16%.

Keywords: Design, Applications, Compression, Files, MP3, Lempel Ziv Welch

1. PENDAHULUAN

Dalam era digital yang semakin berkembang saat ini, kebutuhan akan informasi merupakan hal yang penting bagi masyarakat umum. Pengiriman dan penyimpanan data dapat mempengaruhi pekerjaan manusia dalam melakukan pengolahan data, bertukar data serta informasi. penggunaan audio dalam berbagai konteks seperti musik, dan layanan streaming semakin banyak. Namun, ukuran *file* audio yang besar dapat menjadi kendala dalam hal pengiriman dan penyimpanan data. Maka dari itu diperlukan suatu teknik yang dapat mengurangi ukuran *file* audio dari yang berukuran besar menjadi ukuran yang lebih kecil tanpa mengurangi kualitas dari audio tersebut. Salah satu bentuk data yang sering digunakan adalah *file* audio dalam format Mp3.

Untuk mengatasi masalah tersebut, kompresi atau pemampatan data sangat diperlukan untuk memudahkan waktu pengiriman serta ruang penyimpanan menjadi lebih sedikit. ada berbagai

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

macam algoritma dalam melakukan kompresi pada *file* audio, penulis menggunakan algoritma *Lempel Ziv Welch (LZW)*. Algoritma ini melakukan kompresi dengan menggunakan *dictionary*, dimana fragmen-fragmen teks digantikan dengan indeks yang diperoleh dari sebuah kamus. *Lempel Ziv Welch* memiliki struktur yang sederhana dan dapat di implementasikan dengan relatif mudah dan pemahaman tentang cara kerjanya dapat diperoleh dengan cepat.

Penelitian Tetti Purnama Sari melakukan penelitian dengan judul "Penerapan Algoritma *Levenstein* pada Aplikasi Kompresi *File* Audio" dengan menggunakan Algoritma *Levenstein* dapat memperkecil ukuran *file* Mp3 dari sebelumnya sehingga pada saat di simpan akan menghematkan ruang penyimpanan dan saat di pindahkan *file* nya semakin cepat prosesnya [1].

Penelitian yang dilakukan oleh Aries Suharso, dkk pada tahun 2020 tentang Kompresi *File* Menggunakan Algoritma *Lempel Ziv Welch (LZW)* dapat di simpulkan bahwasannya algoritma *Lempel Ziv Welch* telah dapat digunakan untuk memperkecil ukuran *file* pada *file* teks, citra, dan suara yang mempunyai besar kapasitas berbeda yang disimpan pada memori sehingga dalam pengiriman *file* tidak membutuhkan waktu yang lama. Hal ini terbukti dari hasil kompresi algoritma *Lempel Ziv Welch* dapat melakukan proses kompresi untuk memperkecil ukuran *file* dengan rata-rata rasio kompresi terhadap *file* teks sebesar 51,04% dan rata- rata waktunya 2,56 detik, *file* citra sebesar 37,26% dan rata- rata waktunya 2,44 detik dan *file* suara sebesar 32,89% dan rata- rata waktunya 0,44 detik. Keseluruhan rata- rata tingkat rasio kompresi untuk semua tipe *file* yang diuji menggunakan algoritma *Lempel Ziv Welch* adalah 40,40% dan rata- rata waktu yang dibutuhkan selama 1,81 detik [2].

Penelitian yang dilakukan Riyo Oktavianty Finola pada tahun 2019, berhasil mengkompres *file* audio berekstensi Mp3 menggunakan metode *interpolative*. Selama kompresi, *file* masukan berekstensi *.MP3 dan *file* keluaran berekstensi *.ipc. Sebuah metode inovatif dalam menetapkan kode dinamis ke simbol data yang berbeda secara signifikan dari algoritma lainnya. Ini akan memampatkan ukuran *file* audio dan menghemat ruang penyimpanan dengan menggunakan algoritma *interpolative coding* untuk *file* audio [3].

Penelitian Bakara, J., menyatakan bahwa Algoritma *Lempel Ziv Welch (LZW)* melakukan kompresi dengan menggunakan *dictionary*, dimana fragmen-fragmen teks diganti dengan indeks yang diperoleh dari kamus[4].

Berdasarkan penelitian yang dilakukan oleh Imam Marzuki pada tahun 2018 terhadap aplikasi kompresi pengiriman data dengan metode *LZW (Lempel Ziv Welch)*, di simpulkan bahwa hasil percobaan yang dilakukan menunjukkan bahwa aplikasi kompresi data dapat mengkompres data dengan benar, hasil kompresi sesuai dengan algoritma *Lempel Ziv Welch*, dimana semakin banyak data yang berkarakter sama, maka semakin besar datanya, maka hasil kompresinya bisa semakin kecil, begitu pula sebaliknya, jadi hasilnya tidak ditentukan oleh jumlah *byte*, tetapi oleh jumlah pengaturan karakter yang sama didalam data[5].

Penelitian tentang kompresi *file* audio juga pernah dilakukan oleh Berry Richard Ornos Sinambela pada tahun 2020 dengan menggunakan algoritma *Star-Step-Stop code* telah berhasil mengkompresi *file* audio dari ukuran sebelum di kompresi dan hasil yang cukup efektif yang menghasilkan 30% telah dikompresi dari sebelumnya dengan rasio 1,14[6].

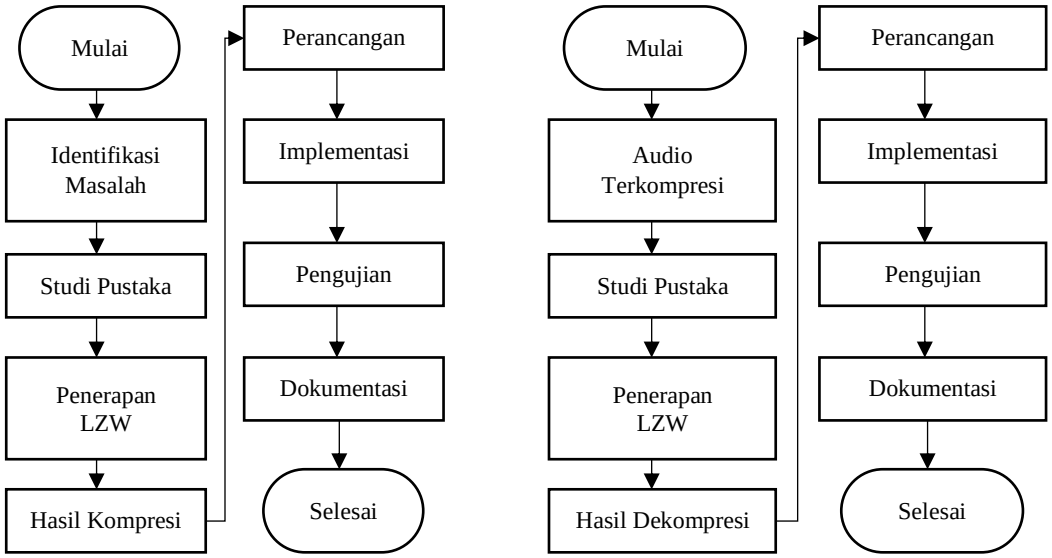
Pada pembahasan ini proses hasil sebuah *file* Mp3 yang sudah terkompres, kemudian dalam pegembalian *file* Mp3 semula akan melalui proses dekompresi. Dari beberapa penjabaran di atas, maka penulis bermaksud untuk memaparkan langkah-langkah untuk menghasilkan solusi seperti yang dijelaskan pada penelitian ini dengan judul "Perancangan Aplikasi Kompresi *File* MP3 Dengan Menggunakan Algoritma Lempel Ziv Welch (LZW)".

2. METODE PENELITIAN

Perancangan Aplikasi Kompresi File MP3 Dengan Menggunakan Algoritma *Lempel Ziv Welch* (LZW) memiliki metode penelitian sebagai berikut:

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)



Gambar 1. (a) Kerangka Metode Penelitian Kompresi, (b) Kerangka Metode Penelitian Dekompresi

2.1 Perancangan Aplikasi

Terkait tentang judul yang akan dibahas, penelitian ini dikerjakan dalam bentuk skripsi. Dalam bab ini membahas tentang perancangan, aplikasi, kompresi, *file* Mp3, serta metode atau algoritma yang digunakan dalam menulis skripsi. Perancangan adalah rancangan yang di gunakan untuk desain dalam menentukan hasil akhir dalam pembuatan suatu karya dengan jelas dan akurat. Perancangan merupakan suatu gambaran yang dirancang secara tepat atau rapi dalam membuat suatu bentuk atau sketsa untuk menghasilkan tujuan tersebut. Perancangan merupakan proses mendefinisikan apa yang harus dilakukan dengan menggunakan berbagai teknik, yang melibatkan deskripsi detail arsitektur dan komponen serta batasan yang akan ditemui sepanjang proses pengerjaannya [7].

Aplikasi berasal dari kata *application* artinya penerapan lamaran pengguna [8]. Aplikasi adalah penerapan yang melibatkan konversi data, menyimpan sesuatu data, permasalahan, dan karya menjadi suatu media yang dapat di manfaatkan secara efektif untuk mengimplementasikan atau menerapkan konsep atau permasalahan yang ada sehingga berubah menjadi suatu bentuk yang baru tanpa menghilangkan nilai-nilai dasar dari data, permasalahan, dan pekerjaan itu sendiri. Jadi aplikasi merupakan solusi yang lebih mudah dipahami serta mudah diakses oleh pengguna. Pada akhirnya, aplikasi memfasilitasi penyelesaian masalah yang lebih cepat dan tepat.

2.2 Kompresi

Kompresi adalah tindakan mengecilkan *file* yang lebih besar menjadi lebih kecil, sehingga mengurangi jumlah ruang penyimpanan yang diperlukan untuk menampungnya. kompresi adalah teknik yang bertujuan untuk meminimalkan jumlah bit yang digunakan dalam representasi digital berbagai media, termasuk audio, gambar, dan video. Pengurangan ukuran ini dicapai tanpa mengorbankan kualitas informasi yang terkandung dalam data[9]. Kompresi data sangat penting karena bisa memperkecil ukuran *bytes* data, menghemat ruang peyimpanan, mempercepat waktu pengiriman data. Kompresi data sangat banyak memiliki berbagai macam algoritma yang berbeda-beda, cocok untuk berbagai macam data, dan menghasilkan *output* yang berbeda-beda. Namun, prinsipnya tetap sama yaitu untuk mengkompresi suatu data. Teknik kompresi dapat di klasifikasikan menjadi 2 (dua) berdasakan hasil kompresinya yaitu[3]:

1. *Lossless compression*
   Lossless compression merupakan teknik kompresi yang tidak menghilangkan data sebelumnya, dimana hasil data yang di kompres dapat di kembalikan lagi ke data aslinya. Ratio kompresi dengan menggunakan metode *lossless compression* sangat rendah.

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

2. *Lossy Compression*
   Lossy compression merupakan teknik kompresi yang menghasilkan *file* data hasil kompresi yang tidak bisa dikembalikan menjadi *file* data sebelumnya. Ketika data yang dikompresi di *decode* kembali, maka beberapa bagian data yang hilang dan tidak sama jika data tersebut dikembalikan pada hasil *decoding* dan metode ini menghasilkan ratio kompresi yang lebih tinggi dari pada metode *lossless.*

Teknik yang di jadikan kriteria untuk menunjukkan kualitas atau kinerja dari suatu metode kompresi, yaitu[3]:

1. *Ratio of Compression (RC)*
   Ratioof Compression (RC) merupakan perbandingan ukuran bit data sebelum di kompresi dengan ukuran bit data yang sudah di kompresi. Maka dapat di tuliskan sebagai berikut:

$$\text{RC} = 100 - \frac{Ukuran\ Data\ Sebelum\ Dikompresi}{Ukuran\ \ Data\ Sesudah\ Dikompresi}\ x\ 100\% \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

2.3 MPEG-1 Audio Layer 3 (MP3)

Pada awalnya MPEG Audio Layer-3 banyak di pakai oleh para pengguna komputer. *File-file* MPEG Audio Layer-3 di simpan dengan ekstensi nama *file* Mp3. Kemudian MPEG Audio Layer-3 selanjutnya banyak dikenal sebagai Mp3. *File* MPEG terdiri dari bagian-bagian kecil yang disebut *frame*. Biasanya tiap *frame* dapat berdiri sendiri. Tiap *frame* memiliki *header* yang berisi informasi *frame* tersebut. *File* Mp3 adalah format audio yang populer dan sangat umum digunakan untuk menyimpan dan mengirimkan musik atau suara digital. Mp3 adalah singkatan dari MPEG Audio Layer III [1].

Audio adalah bentuk pengiriman informasi atau suara melalui gelombang bunyi yang dapat didengar oleh telinga manusia. Ini bisa berupa suara musik, ucapan, dialog, efek suara, atau bentuk lain dari komunikasi audio. Audio dapat di rekam dan di simpan dalam berbagai format, seperti *file* Mp3, WAV, FLAC, atau format lainnya. Pada umumnya, audio diproduksi menggunakan perangkat elektronik seperti mikrofon untuk merekam suara dan peralatan pemrosesan suara untuk mengedit atau memodifikasi rekaman tersebut. Audio juga dapat diputar melalui berbagai perangkat, seperti pemutar musik, komputer, telepon genggam, televisi, atau sistem suara.

File audio untuk saat ini sangat banyak diminati setiap pengguna android maupun pengguna pc atau laptop. Audio berbentuk Mp3 sangat membantu pengguna dalam melakukan kegiatan sehari-hari maupun dalam dunia pembuatan sebuah video. *File* audio memiliki ukuran berbeda-beda pada saat mengunduh *file* dari setiap situs berbeda ukurannya. Semakin banyak *file* audio yang ada dan semakin besarnya ukuran *file* audio, maka di perlukan aplikasi kompresi untuk memperkecil ukuran sebuah *file*.

2.4 *Lempel Ziv Welch* (LZW)

Lempel ziv welch (LZW) adalah algoritma kompresi *lossless* universal yang diciptakan Abraham Lempel, Jacob Ziv, dan Terry Welch. Algoritma ini melakukan kompresi dengan menggunakan *dictionary*, dimana fragmen-fragmen teks digantikan dengan indeks yang diperoleh dari sebuah kamus. Prinsip umum kerja algoritma *Lempel Ziv Welch* adalah mengecek setiap karakter yang muncul kemudian menggabungkan dengan karakter selanjutnya menjadi sebuah string jika string baru tersebut akan di indekskan maka string baru tersebut akan di indekskan kedalam *dictionary* [10].

Langkah-langkah proses dalam algoritma *(Lempel Ziv Welch) LZW* yaitu:

1. *Dictionary* (kamus) di inisialisasi dalam jumlah bit yang lebih sedikit dibandingkan string aslinya
2. P adalah karakter pertama dalam stream karakter.
3. Q adalah karakter berikutnya dalam stream karakter.
4. Lakukan pengecekan apakah (P+Q) terdapat dalam *dictionary*?
   a. Jika: ya" maka P = P+Q (gabungkan P dan Q menjadi string baru)
   b. Jika "tidak" maka:
      1. Output sebuah kode untuk menggantikan string P.
      2. Tambahkan string (P+Q) kedalam *dictionary* dan berikan nomor/kode berikutnya yang belum digunakan dalam *dictionary* untuk string tersebut.

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

3. P = Q

5. Lakukan pengecekan apakah masih ada karakter berikutnya dalam *stream* karakter?
   a. Jika "ya" maka kembali kelangkah 2.
   b. Jika "tidak" maka *output* kode yang menggantikan string P, lalu terminasi proses *(stop).*

Urutan langkah dekompresi algoritma LZW adalah sebagai berikut [9]:

1. Kamus di inisialisasikan dengan semua karakter dasar yang ada
2. CW ← input pertama
3. Lihat kamus dan simpan string dari kode tersebut (string CW sebagai output)
4. C ← karakter pertama dalam string CW
5. PW ← CW
6. CW ← indeks berikutnya dalam input
7. Jika CW terdapat dalam kamus
   Jika "tidak" :
   a. Simpan string CW ke output
   b. P ← string PW
   c. C ← string karakter pertama dari CW
   d. Tambahkan string (P+C) ke dalam kamus
   Jika "ya" :
   a. Output string S ← string PW + c ke stream karakter
   b. P ← string PW
   c. Karakter pertama dari string S
   d. Tambahkan string (P+C) ke dalam kamus
8. Apakah masih ada indeks berikutnya
   a. Jika "ya" Kembali ke langkah 5
   b. Jika "tidak" maka terminasi proses (*stop*)

3. HASIL DAN PEMBAHASAN

Analisis penelitian ini berdasarkan metodologi kinerja terhadap algoritma Lempel Ziv Welch dengan menerapkan algoritma LZW pada *audio* berformat Mp3. Selanjutnya akan dibuat perancangan perangkat lunak untuk kompresi audio. Algoritma Lempel Ziv Welch (LZW) merupakan sebuah algoritma kompresi data tanpa kehilangan. Algoritma ini digunakan untuk mengurangi ukuran data dengan cara mengidentifikasi dan menggantikan pola berulang dalam data dengan kode yang lebih pendek dari kamus yang dibangun selama proses kompresi.

Langkah awal yaitu melakukan proses kompresi dengan mengambil nilai hexadecimal melalui aplikasi HxD. Selanjutnya nilai *hexadecimal* di kompresi dengan mengikuti langkah-langkah penerapan algoritma Lempel Ziv Welch. File audio yang berekstensi Mp3 mempunyai ukuran yang cukup besar, semakin lama waktu pada file audio maka makin besar pula ruang penyimpanan yang diperlukan, dan proses pengiriman file tergolong cukup lama. Proses penerapan algoritam lempel ziv welch dilakukan pada sisi proses kompresi dan proses dekompresi.

Berikut analisis proses kompresi *fiel* Mp3 Mengunakan *Lempel Ziv Welch*

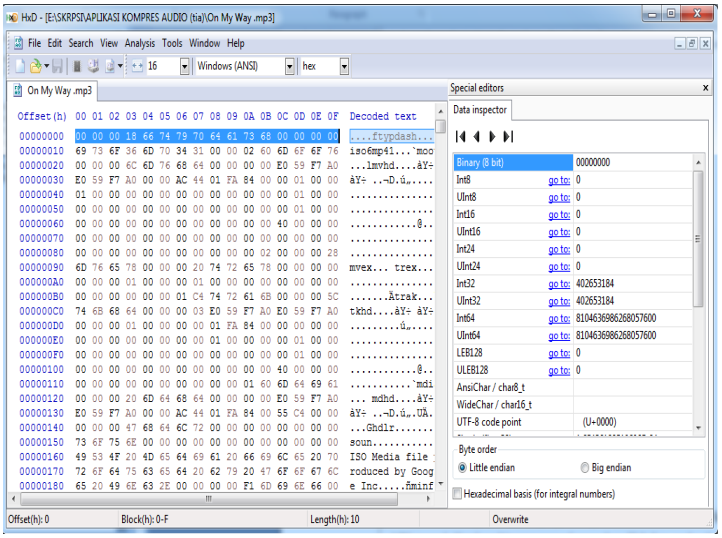
a. Menginput *File*
b.

Tabel 1. Sampel *File* yang akan dikompresi

Nama *File*	**On My Way**
Jenis item	File MP3 (mp3)
Ukuran	11,6 MB
Durasi	00:08:47

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)



Gambar 2. Niai Hexadesimal Sampel Audio

Gambar di atas yang mendapatkan nilai *Hexadecimal File* Mp3 sampel. Untuk perhitungan manual yang akan dilakukan, maka akan diambil sampel sebanyak 32 karakter nilai *Hexadecimal* pada *File* Mp3 tersebut. Metode LZW akan melakukan inisialisasi berdasarkan kode ASCII yaitu 0-255 dan indeks pada kamus dimulai dari 256 hingga seterusnya. Tahapan penerapan kompresi menggunakkan metode LZW yaitu :

1. Proses di inisialisasikan dalam jumlah bit yang lebih sedikit dibandingkan *string* aslinya.

Tabel 2. Inisialisasi

Sampel	Decimal	Sampel	Decimal
18	24	68	104
66	102	69	105
74	116	6F	111
79	121	36	54
70	112	6D	109
64	100	34	52
61	97	31	49
73	115	60	96
02	2		

2. P adalah karakter pertama dalam *stream* karakter.

Tabel 3. *Stream* karakter pertama

18	24

3. Q adalah karakter berikutnya dalam *stream* karakter.

Tabel 4. *Stream* karakter kedua

66	102

4. Apakah *string* (P + Q) terdapat dalam *dictionary* ?
   a. Jika “ya” maka P = P + Q (gabungan P dan Q menjadi *string* baru)
   b. Jika “tidak” maka:
      1) *Output* sebuah kode untuk menggantikan *string* P.
      2) Tambahkan *string* (P + Q) ke dalam *dictionary* dan berikan nomor/kode berikutnya yang belum diganti dalam *dictionary* untuk *string* tersebut.
      3) P = Q
5. Apakah masih ada karakter berikutnya dalam *stream* karakter ?
   a. Jika “ya” maka kembali ke langkah 2.
   b. Jika “tidak” maka *output* kode yang menggantikan *string* P, lalu terminasi proses (*stop)*.

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

Tabel 5. Kompresi

INPUT = P	NxCH=C	OUTPUT	ADD TO DICTIONARY	
		INDEX	INDEX	STRING
00	00	0	256	0000
00	00			
00	18	0	257	0018
18	66	24	258	1866
66	74	102	259	6674
74	79	116	260	7479
79	70	121	261	7970
70	64	112	262	7064
64	61	100	263	6461
61	73	97	264	6173
73	68	115	265	7368
68	00	2	266	6800
00	00			
00	00			
0000	00	256	267	000000
00	69	0	268	0069
69	73	105	269	6973
73	6F	115	270	736F
6F	36	111	271	6F36
36	6D	54	272	366D
6D	70	109	273	6D70
70	34	112	274	7034
34	31	52	275	3400
31	00	49	276	3100
00	00			
0000	02	0	277	000002
02	60	2	278	0260
60	6D	96	279	606D
6D	6F	109	280	6D6F
6F	6F	111	281	6F6F
6F	76	111	282	6F76
76	-	118	283	76

Sehingga hasil kompresi menggunakkan metode LZW yaitu :
0, 0, 24, 102, 116, 121, 100, 97, 115, 2, 256, 0, 105, 115, 111, 54, 109, 112, 52, 49, 0, 2, 96, 109, 111, 111, 118

Total awal sebelum kompresi = total input * bit
= 32 * 8
= 256 *bit*

Total setelah kompresi = total *input * bit dictionary*
= 27 * 8
= 216 *bit*

Hasil Rasio Kompresi :
Rasio = 100 – (216/256) x 100 %
Rasio = 100 – (0,84) x 100%
Rasio =100 - 84 %
Rasio = 16 %

Hasil kompresi yang telah dilakukan maka tahap dekompresi menggunakkan metode LZW adalah sebagi berikut :
Indeks hasil kompresi:
0, 0, 24, 102, 116, 121, 100, 97, 115, 2, 256, 0, 105, 115, 111, 54, 109, 112, 52, 49, 0, 2, 96, 109, 111, 111, 118

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

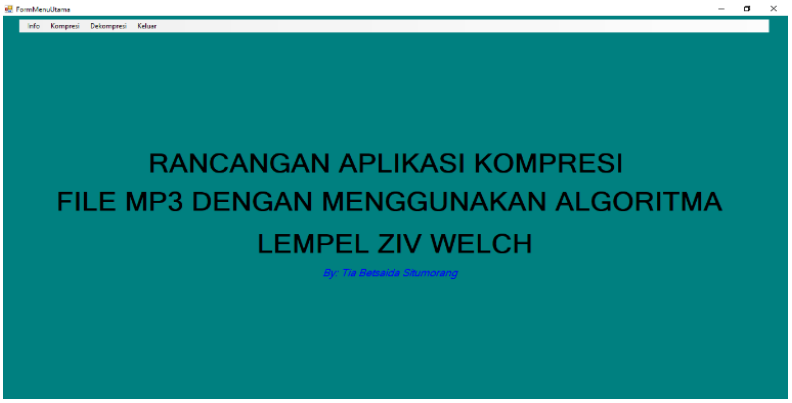
Tabel 6. Dekompresi

PW	CW		P	C	OUTPUT	ADD TO DICTIONARY	
STRING	INDEX	STRING				INDEX	STRING
NULL	0	0		0	0		
0	0	0000	0000	00	00	256	000000
00	24	18	00	18	18	257	0018
18	102	66	18	66	66	258	0018
66	116	74	66	74	74	259	1800
74	121	79	74	79	79	260	7479
79	100	64	79	64	64	261	7964
64	97	61	64	61	61	262	6461
61	115	73	61	73	73	263	6173
73	2	02	73	02	02	264	7302
02	256	0000	02	0000	0000	265	020000
0000	0	00	0000	00	00		
00	105	69	00	69	69	266	0069
69	115	73	69	73	73	267	6973
73	111	6F	73	6F	6F	268	736F
6F	54	36	6F	36	36	269	6F36
36	109	6D	36	6D	6D	270	366D
6D	112	70	6D	70	70	271	6D70
70	52	34	70	34	34	272	7034
34	49	31	34	31	31	273	3431
31	0	00	31	00	00	274	3100
00	2	02	00	02	02	275	0002
02	96	60	02	60	60	276	0260
60	109	6D	60	6D	6D	277	606D
6D	111	6F	6D	6F	6F	278	6D6F
6F	111	6F	6F	6F	6F	279	6F6F
6F	118	76	6F	76	76	280	6F76

3.1 Implementasi

Tampilan sistem merupakan tampilan akhir antarmuka dari sistem yangdirancang. Terdapat 4 tampilan halaman pada sistem ini, yaitu *Form* Menu Utama, *Form* info, *Form* Kompresi, *Form* Dekompresi.

1. *Form* Menu Utama
   Form menu utama merupakan tampilan awal saat menjalan program. Pada tampilan terdapat beberapa menu yang berfungsi untuk mengakses *form-form*pada sistem ini. Berikut tampilan *form* menu utama :

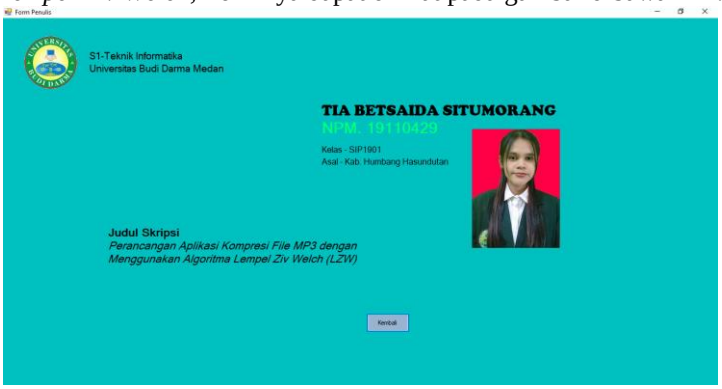


Gambar 3. *Form* Menu Utama

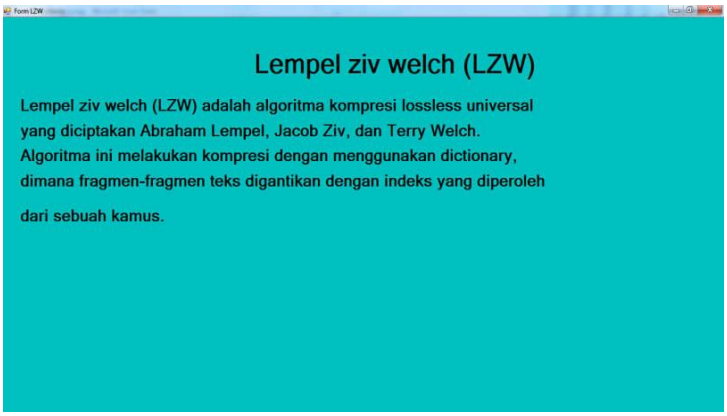
Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

2. *Form* Info
   Form info yaitu tampilan untuk menampilkan biodata penulis dan informasi tentang algoritma *Lempel Ziv Welch*, *Form*nya dapat dilihat pada gambar dibawah ini:

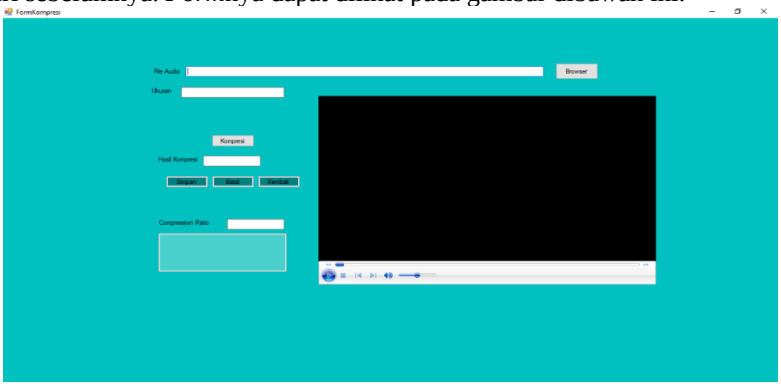


Gambar 4. *Form* Penulis



Gambar 5. *Form Lempel Ziv Welch*

3. *Form* Kompresi
   Form kompresi yaitu tampilan yang akan digunakan untuk mengkompresi *file* Audio yang akan di *input*. Setelah dilakukan proses kompresi, maka akan mendapatkan hasil yang lebih kecil dari sebelumnya. *Form*nya dapat dilihat pada gambar dibawah ini:

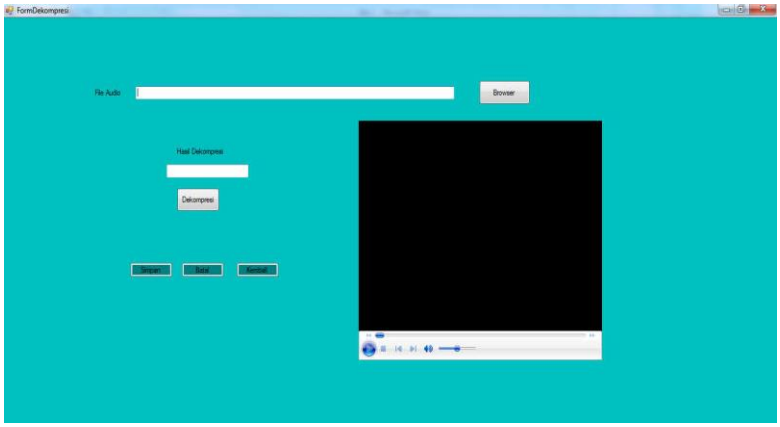


Gambar 6. *Form* Kompresi

4. *Form* Dekompresi
   Form dekompresi merupakan *form* yang akan digunakan untuk mendekompresi *file* hasil dari kompresi sebelumnya. *Form* nya dapat dilihat pada gambar dibawah ini:

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 3, Nomor 2, Maret, 2024, Page 60-70
ISSN 2828- 7921 (media online)

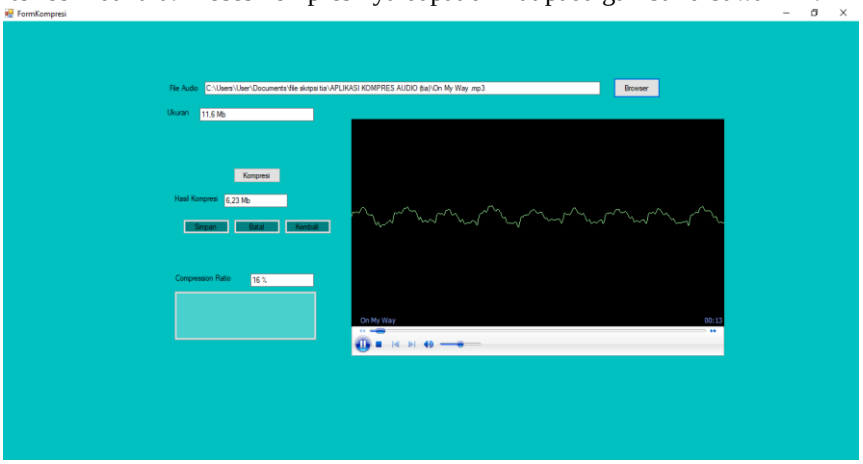


Gambar 7. *Form* Dekompresi

3.2 Hasil

1. *Form* Hasil Kompresi

Pada *Form* ini akan menampilkan proses kompresi pada *file* Audio dengan menggunakan algoritma *Lempel Ziv Welch* dimana terlebih dahulu pengguna akanmenginput *file* Audio yang akan dikompresi terlebih dahulu. Proses kompresinya dapat dilihat pada gambar dibawah ini:



Gambar 8. *Form* Hasil Kompresi menggunakan *Lempel Ziv Welch*

2. *Form* Hasil Dekompresi

Form ini berisi tentang hasil dekompresi pada *file* Audio yang sebelumnyasudah dikompresi.

Gambar 9. Form Hasil Dekompresi

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**

4. KESIMPULAN

Kesimpulan dari hasil dan pembahasan dalam penelitian ini antara lain, proses kompresi dan dekompresi dengan menggunakan algoritma Lempel Ziv Welch telah berhasil melakukan kompresi file audio berektensi .*mp3 sehingga proses kompresi dan dekompresi dapat berjalan sesuai teknik kompresi dan dekompresi. Setelah menerapkan algoritma Lempel Ziv Welch dalam mengkompresi file Mp3 dapat disimpukan bahwa kompresi file Mp3 Berhasil dijalankan dan menghasilkan nilai Compression Ratio 16%. Kompresi File audio dirancang dengan menggunakan aplikasi Microsoft Visual Studio 2010 dengan menerapakan teknik kompresi dari algoritma Lempel Ziv Welch dapat berjalan dengan baik.

REFERENCES

[1] T. P. Sari, S. D. Nasution, and R. K. Hondro, "Penerapan Algoritma Levenstein Pada Aplikasi Kompresi File Mp3," *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)*, vol. 2, no. 1, 2018.

[2] A. Suharso, J. Zaelani, and D. Juardi, "KOMPRESI FILE MENGGUNAKAN ALGORITMA LEMPEL ZIV WELCH (LZW)," *Komputasi: Jurnal Ilmiah Ilmu Komputer dan Matematika*, vol. 17, no. 2, pp. 372–380, 2020.

[3] R. O. Finola, "Penerapan Algoritma Interpolative Coding Untuk Kompresi File Audio," *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)*, vol. 3, no. 1, 2019.

[4] J. Bakara, "Implementasi Algoritma LZW dan Kuantisasi Dalam Kompresi Citra Digital," *Pelita Informatika: Informasi dan Informatika*, vol. 5, no. 3, 2020.

[5] I. Marzuki, "Aplikasi Kompresi Untuk Pengiriman Data Menggunakan Metode LZW (Lemple Ziv Welch)," *Energy-Jurnal Ilmiah Ilmu-Ilmu Teknik*, vol. 7, no. 2, pp. 38–43, 2017.

[6] B. R. O. Sinambela, M. Syahrizal, and K. Siregar, "Penerapan Algoritma Start-Step-Stop Code Pada Kompresi File Audio," *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)*, vol. 4, no. 1, 2020.

[7] N. Azis, G. Pribadi, and M. S. Nurcahya, "Analisa dan Perancangan Aplikasi Pembelajaran Bahasa Inggris Dasar Berbasis Android," *Ikraith-informatika*, vol. 4, no. 3, pp. 1–5, 2020.

[8] E. Sany, "Aplikasi eVoting Pada Pemilihan Presiden Badan Eksekutif Mahasiswa (BEM) Universitas Nurdin Hamzah," in *Seminar Nasional Informatika (SENATIKA)*, 2021, pp. 398–408.

[9] C. B. Simbolon, "Penerapan Metode Self Delimiting Codes Dalam Kompresi File," *Journal of Computing and Informatics Research*, vol. 1, no. 2, pp. 50–55, 2022.

[10] H. Chandra, "APLIKASI KOMPRESI DATA DENGAN ALGORITMA LZW DAN PENGAMANAN DATA DENGAN ALGORITMA KRIPTOGRAFI AES PADA DROPBOX." KODEUNIVERSITAS041060# UniversitasBuddhiDharma, 2018.

Submitted: **27/02/2024**; Accepted: **20/03/2024**; Published: **31/03/2024**