Penerapan Kode Huffman Dalam Kompresi Data Digital

Bintang Hijriawan Jachja - 13521003[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13521003@std.stei.itb.ac.id

***Abstract*—Pertumbuhan data di era digital sangat pesat, sehingga perlu dilakukan kompresi terhadap data-data tersebut agar proses transfer dan penyimpanan data lebih efisien. Pada makalah ini akan dibahas mengenai salah satu metode kompresi data yaitu metode kompresi Huffman**

***Keywords*—Huffman, kompresi, data, pohon.**

I. PENDAHULUAN

Pertumbuhan data di era digital ini sangat pesat. Penggunaan *gadget*, komputer, dan internet semakin tidak bisa dihindari. Data yang dihasilkan dari penggunaan komputer dan *gadget* juga semakin besar. Berdasarkan data yang diambil dari IDC, Pada akhir tahun 2022, terdapat 97×10^{21} *byte* data di dunia, dan diproyeksi mencapat 175×10^{21} *byte* data pada tahun 2025. Data yang sangat besar ini tentunya perlu dikompresi agar pengolahan data bisa semakin cepat.

Kompresi adalah metode yang bertujuan untuk mengurangi ukuran data, baik berupa video, gambar, suara, ataupun teks. Dalam dunia digital, kompresi berarti mengurangi jumlah bit yang dibutuhkan untuk menyimpan data. Secara umum kompresi data digital dibagi menjadi dua yaitu *lossless compression* dan *lossy compression*. *Lossless compression* adalah metode kompresi yang menjaga keutuhan data sama seperti sebelum dikompres, tanpa ada ada data yang dibuang. Sebaliknya, *lossy compression* adalah kompresi data yang mengakibatkan adanya data yang hilang setelah dikompresi. Metode yang termasuk dalam *lossless compression* diantaranya adalah DCT (Discreet Cosine Transform), Vector Quantisation, Entropy coding. Sementara yang termasuk dalam *lossy compression* adalah RLE (Run Length Encoding), string-table compression, LZW (Lempel Ziff Welch).

Kompresi Huffman adalah metode kompresi yang termasuk dalam *lossless compression*. Ini berarti tidak ada data yang hilang selama proses kompresi. Algoritma huffman dilakukan berdasarkan frekuensi kemunculan dari satuan data. Misal dari satuan data adalah karakter pada data text, atau pixel pada data gambar. Dibandingkan dengan menggunakan jumlah bits yang konstan untuk menyimpan data, Algoritma Huffman menggunakan jumlah bit yang lebih sedikit untuk data yang sering muncul, dansebaliknya menggunnakan jumlah bit yang lebih banyak untuk data yang jarang muncul. Hal ini mengakibatkan pengurangan dari jumlah bit rata-rata untuk merepresentasikan data.

II. TEORI DASAR

2.1 Data Digital

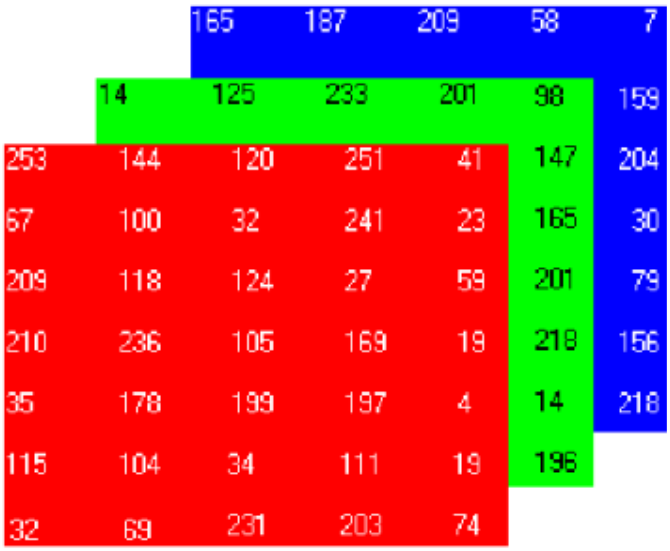
Data digital adalah representasi elektrik dari sebuah informasi dalam format yang bisa dimengerti oleh komputer. Semua data, baik itu berupa video, gambar, suara, ataupun tulisan, disimpan dalam bentuk biner.

Pada data berupa teks, masing-masing karakter akan dikodekan menjadi bilangan biner. Pengkodean yang umum digunakan pada komputer adalah pengkodean UTF-8. Setiap karakter komputer diwakili oleh string 8 bit.

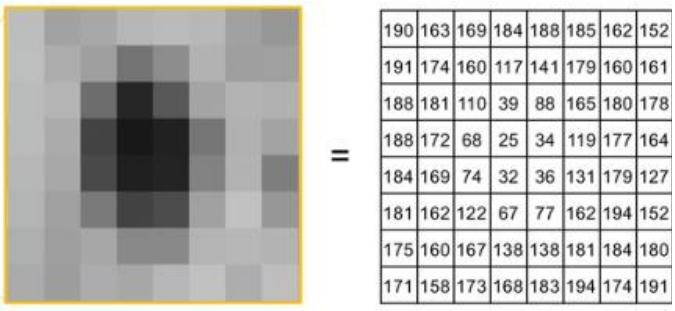
Decimal	Hexadecimal	Binary	Octal	Char
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F

Gambar 2.1 Potongan Tabel Pengkodean UTF-8

Data berupa gambar disimpan dalam bentuk matriks dimana setiap elemen matriks mewakili pixel gambar dan nilai elemen matriks mewakili kecerahan pixel yang diwakili. Pada gambar RGB (red-green-blue), setiap pixel memiliki 3 nilai, yang biasanya ada di rentang 0 sampai 255, yang merepresentasikan kecerahan warna merah, hijau dan biru pixel tersebut. Setiap nilainya dapat diwakili oleh 8 bit atau 1 byte string, sehingga setiap pixel berukuran 3 byte.



Pada gambar grayscale, setiap pixel hanya memiliki 1 nilai kecerahan sehingga 1 pixel berukuran 1 byte.



Data video merupakan kumpulan data gambar, sehingga penyimpanannya sama. Tanpa kompresi, jika ada data sebuah video beresolusi 1280 × 720 pixel 60 fps berdurasi 5 menit, data tersebut akan berukuran 49,8 GB. Ukuran ini sangat besar dan akan memakan banyak tempat, oleh karena itu diperlukan adanya kompresi data.

2.2 Kompresi Data

Terdapat dua tipe utama dalam kompresi data di berbagai jenis data, yaitu lossy data compression yang biasa digunakan untuk gambar dan lossless data compression yang biasa digunakan untuk teks dan file binary.

1. Lossy Data Compression

Lossy Data Compression adalah metode kompresi dimana terdapat pembuangan data pada file yang telah dikompresi, sehingga tidak sama persis dengan file awalnya, tetapi tetap berfungsi sama. Data yang dikompresi tidak akan pernah bisa kembali persis seperti semula dimana data belum dikompresi. Bila data didekompresi maka hasilnya tidak akan sama persis dengan data awal. Akibatnya, lossy data compression tidak cocok untuk melakukan kompresi terhadap data yang bersifat kritikal, semisal data teks. Jenis kompresi ini banyak digunakan pada Digitally Sampled Analog Data (DSAD). DSAD terdiri dari file audio, video, dan gambar.

Contoh dari pembuangan data, pada file suara misalnya, data yang menyimpan suara dengan frekuensi sangat rendah atau sangat tinggi yang tidak dapat didengar oleh manusia, dapat dibuang. Contoh sehari-hari dari lossy compression misalnya daam streaming media dari internet. Contoh dari format lossy compression adalah JPEG, MPEG, MP3.

2. Lossless Data Compression

Sebaliknya, Lossless data compression adalah metode kompresi dimana tidak ada data yang dibuang dari file asli, sehingga ketika data didekompresi, akan menghasilkan data yang sama persis dengan data awal, sehingga kompresi dapat dilakukan berulang-ulang tanpa ada data yang hilang. Lossless compression digunakkan jika penting untuk data hasil kompresi dapat kembali ke data awal tanpa perubahan. Contoh lossless compression yang sering kita pakai adalah format ZIP.

2.3 Algoritma Huffman

Algoritma huffman adalah algoritma enkoding yang sering digunakan di komputer sains dan teori informasi. Algoritma huffman didasarkan pada frekuensi kemunculan data. Prinsipnya adalah dengan menggunakan jumlah bit yang sedikit untuk menyimpan data yang sering muncul. Panjang rata-rata dari kode huffan bergantung pada statistik frekuensi kemunculan data. Algoritma Huffman dipakai secara luas dimana-mana. Banyak format file yang populer yang menggunakan algoritma Huffman, diantaranya GZIP, PKZIP, BZIP2, dan pada gambar JPEG dan PNG. Algoritma Huffman sangat efektif dalam melakukan kompresi terhadap data yang banyak memiliki pengulangan, dimana data tersebut banyak dijumpai, semisal teks dalam bahasa inggris, dan gambar. Algoritma Huffman banyak menjadi back-end dari metode kompresi lainnya. Misalnya Brotli Compression, salah satu algoritma kompresi yang dikeluarkan oleh google, menggunakan algoritma Huffman. Huffman banyak dipakai karena soal masalah paten, algoritma Huffman dapat bebas dipakai. Algoritma Huffman menggunakan sebuah pohon biner dalam proses kompresi yang dinamakan pohon Huffman, yaitu pohon yang menyimpan enkoding dari item-item data dalam bits. Pohon huffman ini dibuat pada saat data dikompresi dan digunakan untuk mendekompresi data. Dalam algoritma huffman pertama-tama hitung frekuensi kemunculan dari setiap data. Lalu berdasarkan data itu kita buat pohon Huffman. Algoritma untuk membentuk pohon huffman adalah sebagai berikut :

1. Pilih dua data dengan frekuensi kemunculan paling sedikit. Kemudian dua data tersebut dijadikan sebagai anak dari simpul baru, yaitu gabungan dari kedua data dengan frekuensi kemunculannya merupakan akumulasi dari frekuensi kemunculan kedua data yang kita pilih.
2. Pilih dua data berikutnya, termasuk simpul yang baru kita buat, yang memiliki peluang terkecil.
3. Selanjutnya kita ulangi langkah 1 hingga seluruh data sudah terpilih.

Dari pohon Huffman kita mendapatkan kode huffman untuk simbol kita. Caranya adalah dengan memberi nomor pada sisi pohon dengan angka 0 pada sisi kiri dan 1 pada sisi kanan. Untuk mendapatkan kode untuk simbol kita, kita perlu menelusuri jalan dari bagian paling atas pohon kita hingga ke daun dimana simbol kita berada. Nomor yang kita lalui sepanjang jalan adalah kode Hufman untuk simbol kita.

Representasi kode huffman bisa berbeda-beda, tergantung pembuatan pohon huffmannya. Hal ini disebabkan jika ada lebih dari 2 data dengan frekuensi kemunculan sama pada langkah 1 atau 2, dapat dipilih 2 data manapun sehingga urutan pohon akan berbeda. Namun walaupun representasi kode berbeda, rasio kompresi akan tetap sama.

Pada kompresi gambar, akan dilakukan kompresi huffman pada setiap pixelnya. Setiap pixel mengandung 3 nilai yang masing-masing nilai direpresentasikan dengan angka 8 bit. Nilai-nilai ini yang akan menjadi data yang kemudian dikompres menjadi ukuran yang lebih kecil.

Algoritma Huffman cukup mudah untuk implementasikan pada bahasa pemrograman, disini penulis menggunakan bahasa pemrograman Python dengan bantuan modul heapq.

```
class node:
    def __init__(self, freq, symbol, left=None, right=None):

        self.freq = freq

        self.symbol = symbol

        self.left = left

        self.right = right

        self.huff = ''
```

Disini penulis membuat class node sebagai node pada Huffman Tree dengan atribut frekuensi, simbol, anak kiri, anak kanan, dan atribut huff yang bernilai 0 jika node merupakan anak kiri, dan 1 jika node merupakan anak kanan dari parent node

```
chars = []
freq = {}
nodes = []

word = input()

for character in word:
    if character not in chars:
        chars.append(character)
    if character in freq:
        freq[character] += 1
    else:
        freq[character] = 1

for x in range(len(chars)):
    heapq.heappush(nodes, node(freq[chars[x]], chars[x]))
```

.

Disini penulis membuat *list* chars yang berisi huruf-huruf berbeda pada kata, *dictionary* freq yang berisi frekuensi dari huruf-huruf yang ada di char dan nodes yang berisi node setiap karakter yang ada di chars. *Loop* pertama berfungsi untuk memasukan huruf-huruf yang ada di 'word' kedalam list chars dan menghitung frekuensinya, kemudian *loop* kedua berfungsi membuat node untuk setiap karakter yang ada di chars.

```
while len(nodes) > 1:
    left = heapq.heappop(nodes)
    right = heapq.heappop(nodes)

    left.huff = 0
    right.huff = 1

    newNode = node(left.freq+right.freq, left.symbol+right.symbol, left, right)

    heapq.heappush(nodes, newNode)
```

Lalu program akan menjalankan loop yang akan terus berjalan hingga hanya ada 1 node dalam lost nodes, yaitu node parent yang berisi semua karakter. Loop ini akan mengambil 2 node dengan frekuensi paling kecil, mengisi atribut huff dengan 0 dan 1, dan membuan node baru sebagai parent dari node lama dengan frekuensi akumulatif dari node kiri dan kanan, serta simbol gabungan dari anak kiri dan anak kanan.

```
MATEMATIKA DISKRIT
I -> 00
D -> 10110
R -> 1010
M -> 011
E -> 0100
  -> 0101
T -> 111
A -> 110
S -> 10111
K -> 100
```

Hasil algoritma Huffman pada kalimat "MATEMATIKA DISKRIT" adalah seperti diatas.

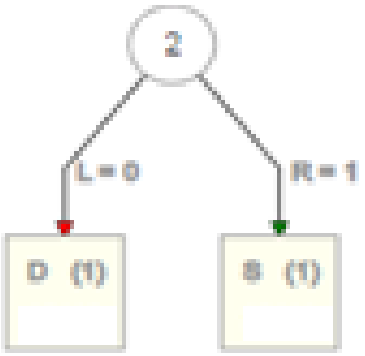
III. ANALISIS

3.1 Kompresi Teks

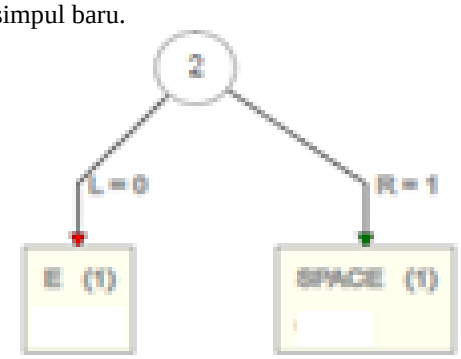
Misalkan kita memiliki sebuah kalimat "MATEMATIKA DISKRIT". Langkah pertama membuat pohon Huffman adalah dengan membuat tabel frekuensi. Tabel frekuensi kalimat diatas seperti berikut:

M	2
A	3
T	3
E	1
I	3
K	2
SPASI	1
D	1
S	1
R	1

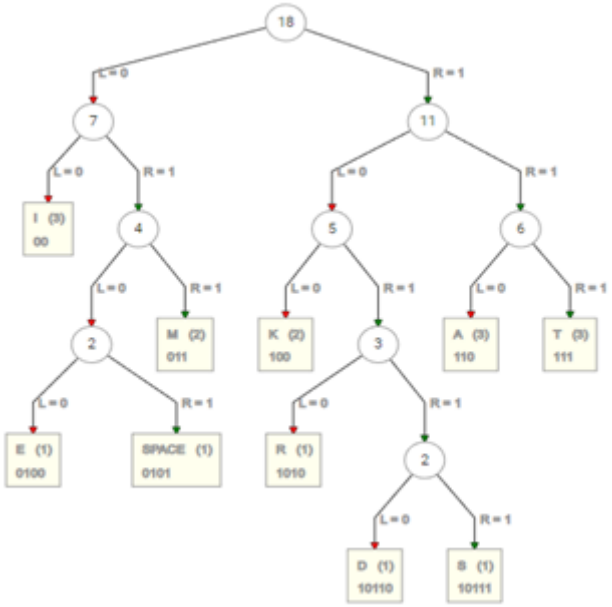
Langkah kedua, pilih 2 data dengan frekuensi terkecil, yang menjadi anak dari simpul baru, yaitu gabungan dari 2 data tersebut.



Selanjutnya, pilih 2 data berikutnya dengan frekuensi terkecil, termasuk simpul baru.



Ulangi langkah 1 hingga semua simpul terpakai. Maka akan terbentuk pohon Huffman seperti berikut:



Tanpa kompresi, kata “MATEMATIKA DISKRIT” akan dikodekan menjadi :
01001101 01000001 01010100 01000101 01001101 01000001 01010100 01001001 01001011 01000001 00100000 01000100 01001001 01010011 01001011 01010010 01001001 01010100

Data tersebut akan berukuran 144 bit, namun setelah kompresi, Data tersebut menjadi :
011 110 111 0100 011 110 111 00 100 110 0101 10110 00 10111 100 1010 00 111

Data seletah kompresi akan berukuran 58 bit, dengan rasio kompresi 40.28%.

3.2 Kompresi Gambar

Misalnya kita punya matrix yang merepresentasikan gambar grayscale berukuran 3 × 3 seperti berikut:

100	150	100
50	200	150
100	150	100

Data diatas akan diubah menjadi vektor satu baris menjadi sebagai berikut :

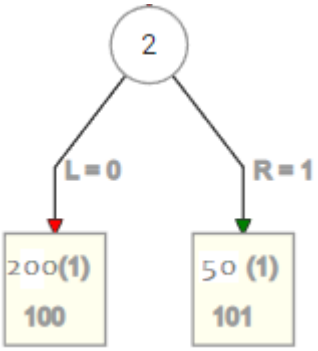
[100 150 100 50 200 150 100 150 100]

Setelah itu dibentuk pohon Huffman dari data-data diatas dengan langkah-langkah sebagai berikut :

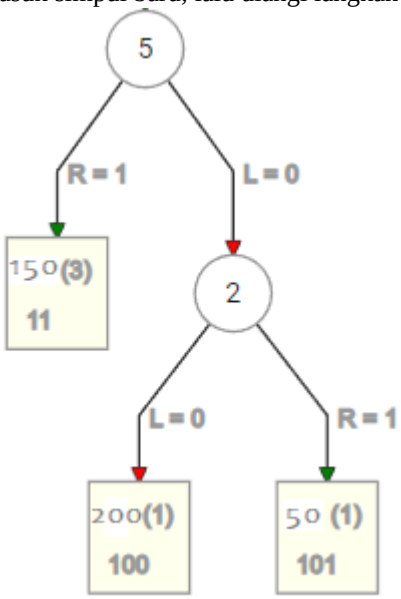
1. Buat tabel frekuensi dari data data diatas

Data	Frekuensi
100	4
150	3
200	1
50	1

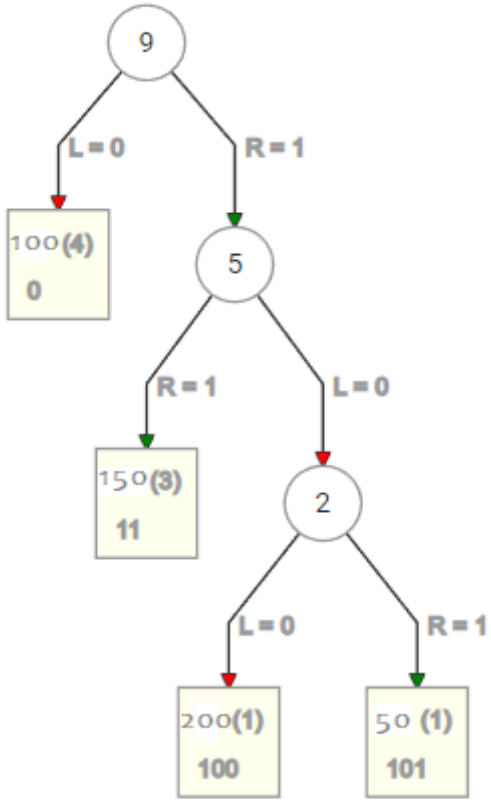
2. Pilih 2 data dengan frekuensi terkecil, yang menjadi anak dari simpul baru, yaitu gabungan dari 2 data tersebut.



3. Pilih 2 data berikutnya dengan frekuensi terkecil, termasuk simpul baru, lalu ulangi langkah 2



Maka akan terbentuk pohon huffman seperti gambar berikut:



Setelah itu, dibuat matriks baru dengan elemen yang merepresentasikan nilai elemen matriks asli.

0	11	0
101	100	11
0	11	0

Tanpa kompresi, data gambar akan berukuran 72 bit, sedangkan setelah kompresi, data berukuran 16 bit dengan rasio kompresi 22.22%.

IV. KESIMPULAN

Algoritma Huffman menggunakan aplikasi pohon prefix biner dalam menentukan kode Huffman. Algoritma Huffman banyak digunakan dalam kompresi data, karena tingkat kompresi data yang tinggi, bergantung pada data yang dikompresi. Algoritma Huffman juga tidak memiliki paten khusus sehingga dapat digunakan secara luas. Terdapat juga algoritma kompresi terkini, semisal Brotli Compression yang menggunakan algoritma Huffman sebagai salah satu algoritma kompresinya. Algoritma Huffman juga dapat digunakan sebagai kompresi gambar digital, dan digunakan sebagai salah satu algoritma kompresi pada format JPEG.

V. UCAPAN TERIMA KASIH

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmatNya sehingga makalah ini dapat diselesaikan dengan baik dan tepat waktu dalam rangka memenuhi tugas mata kuliah IF2120 – Matematika Diskret. Penulis mengucapkan terima kasih kepada Ibu Dra. Harlili S., M.Sc. sebagai dosen pembimbing kelas mata kuliah Matematika Diskret dan para pihak yang telah menghasilkan karya-karya yang berguna untuk menyelesaikan makalah ini. Penulis berharap agar makalah ini sekiranya dapat berguna bagi pembaca dan masyarakat.

REFERENCES

[1] Munir, Rinaldi, (2022), Pohon (Bag. 2), https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2021-2022/Pohon-2021-Bag2.pdf diakses pada 10 Desember 2022

[2] GeeksForGeeks, (2022), Huffman Coding | Greedy Algo-3, https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/ diakses pada 12 Desember 2022

[3] Sather, Hayden, (2022), Applications of Huffman Coding, https://experiencestack.co/applications-of-huffman-coding-73c661f9ef03 diakses pada 12 Desember 2022

[4] Prodanoff, Jordan T. (2022), How Much Data Is Created Every Day in 2022, https://experiencestack.co/applications-of-huffman-coding-73c661f9ef03 diakses pada 12 Desember 2022

[5] Marr, Bernard, (2022), How Muc Data Is There In The World, https://bernardmarr.com/how-much-data-is-there-in-the-world/ diakses pada 12 Desember 2022

[6] Reinsel, David. Gantz, John. Rydning, John. (2018), The Digitization Of The World From Edge to Core, https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf diakses pada 12 Desember 2022

[7] EGNYTE, (2022), What Is Digital Data?, https://www.egnyte.com/guides/governance/digital-data diakses pada 12 Desember 2022

[8] Gianats, (2017), ASCII Table, https://favpng.com/png_view/binary-numbers-extended-ascii-character-encoding-ebcdic-png/TwHZRErS diakses pada 12 Desember 2022

[9] Courtney, Jane, (2001), Three-dimensional RGB Matrix https://www.researchgate.net/figure/A-three-dimensional-RGB-matrix-Each-layer-of-the-matrix-is-a-two-dimensional-matrix_fig6_267210444 diakses pada 12 Desember 2022

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Desember 2022

Bintang Hijriawan Jachja - 13521003