LAPORAN TUGAS BESAR MINIMISASI LOGIC
PEMECAHAN MASALAH DENGAN C
EL2208

Kelompok 1
Anggota :

1. **Kayyisa Zahratulfirdaus (18320011)**
2. **Putri Alfiyyahdianti (18320041)**

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2022

DAFTAR ISI

1. PENDAHULUAN

Suatu rangkaian digital merupakan representasi dari Aljabar Boolean. Aljabar Boolean dapat disusun menjadi rangkaian logika (fungsi logic) , dimana suatu rangkaian logika dapat terdiri dari satu atau lebih gerbang logika.

Suatu rangkaian logika dapat dilakukan penyederhanaan yang dinamakan dengan minimisasi logic function. Penyederhanaan rangkaian logika ini dilakukan agar diperoleh rangkaian dengan fungsi yang sama namun menggunakan gerbang yang paling sedikit. Dikarenakan rangkaian dengan gerbang yang paling sedikit akan lebih murah harga, dan dari segi tata letak komponen nya yang akan lebih sederhana.

Dalam laporan ini terdapat penjelasan mengenai pembuatan program minimisasi logic dengan menggunakan metode Quine-McCluskey dengan menggunakan bahasa C.

2. EKSPLORASI ALGORITMA MINIMISASI LOGIC FUNCTION

Berikut beberapa metode yang dapat dilakukan di minimisasi logic/ penyederhanaan rangkaian logika.

2.1 BOOLEAN ALGEBRA PROPERTIES

Boolean Algebra Properties adalah sebuah set aturan yang dapat digunakan untuk menyederhanakan suatu fungsi logic dengan tidak mengubah fungsi tersebut [4] .

Dalam Boolean Algebra terdapat *postulate*, properties dan teorema yang dapat digunakan [3]

a. Postulates (identitas)

1a: $A=1$ (if $A \neq 0$)	1b: $A=0$ (if $A \neq 1$)
2a: $0 \cdot 0=0$	2b: $0+0=0$
3a: $1 \cdot 1=1$	3b: $1+1=1$
4a: $1 \cdot 0=0$	4b: $1+0=1$
5a: $\overline{1}=0$	5b: $\overline{0}=1$

Gambar 2.1.1 Postulat Boolean Algebra

b. Properties (sifat sifat yang mirip dengan persamaan al jabar biasa)

1. Komutatif

   A.B=B.A

   A+B=B+A

2. Asosiatif

   A.(B.C)=(A.B).C

   A+(B+C) = (A+B)+C

3. Distributif

   A.(B+C)=A.B+A.C

   A+(B.C)=(A+B).(A+C)

c. Theorems (Definisi Boolean Algebra)

1a: $A \cdot 0=0$	1b: $A+0=A$
2a: $A \cdot 1=A$	2b: $A+1=1$
3a: $A \cdot A=A$	3b: $A+A=A$
4a: $A \cdot \overline{A}=0$	4b: $A+\overline{A}=1$
5a: $\overline{\overline{A}}=A$	5b: $A=\overline{\overline{A}}$
6a: $\overline{A \cdot B}=\overline{A}+\overline{B}$	6b: $\overline{A+B}=\overline{A} \cdot \overline{B}$

Gambar 2.1.2 Teorema Boolean Algebra 1

Meskipun pada gambar 2.1.2 merupakan bagian dari teorema Boolean Algebra , kita dapat menggunakan turunan (perpanjangan) dari teorema algebra untuk memudahkan penyederhanaan persamaan.

7a: $A \cdot (A+B)=A$	7b: $A+A \cdot B=A$
8a: $(A+B) \cdot (A+\overline{B})=A$	8b: $A \cdot B+A \cdot \overline{B}=A$
9a: $(A+\overline{B}) \cdot B=A \cdot B$	9b: $A \cdot \overline{B}+B=A+B$
10: $A \oplus B=\overline{A} \cdot B+A \cdot \overline{B}$	
11: $A \odot B=\overline{A} \cdot \overline{B}+A \cdot B$	
$\oplus$ = XOR, $\odot$ = XNOR	

Gambar 2.1.3. Teorema Boolean Algebra 2.

Berikut contoh penyederhanaan fungsi logic menggunakan Boolean Algebra :

F = AB(A+C)

AB(A+C) (hukum distributif)

=ABA +ABC (hukum komutatif)

=AAB+ABC (theorema 3a)

=AB+ABC (theorema 7a)

=AB

2.2 KARNAUGH MAPS (K-MAPS)

Peta Karnaugh (K-maps) adalah metode penyederhanaan rangkaian logika menggunakan representasi tabel. jumlah kolom pada tabel dapat dirumuskan 2^n , dengan nilai n merupakan jumlah variabel. K-maps dapat berbentuk Sum of Product (SOP) dan Product of Sum (POS) [6] sesuai dengan permasalahan. Dan nantinya tabel akan diisi dengan nilai 0 atau 1 kemudian diselesaikan dengan membuat grup .

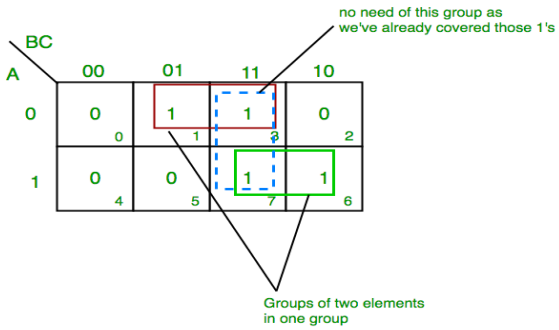
Langkah langkah dalam membuat K-Maps [6] :

1. Pilih K-map sesuai dengan jumlah variabel.
2. Identifikasi minterms atau maxterms seperti yang diberikan dalam masalah.
3. Untuk SOP, letakkan 1 di blok K-map masing-masing ke minterms (0 di tempat lain).
4. Untuk POS, letakkan 0 di blok K-map masing-masing ke maxterms(1 di tempat lain).
5. Buat kelompok persegi panjang yang berisi jumlah suku pangkat dua seperti 2,4,8 ..(kecuali 1) dan cobalah untuk mencakup elemen sebanyak mungkin dalam satu grup.
6. Dari kelompok yang dibuat pada langkah 5 , temukan hasil term dan jumlahkan untuk

Bentuk SOP :

1. K-Maps 3 variabel

   $Z = \sum \quad A, B, C(1,3,6,7)$

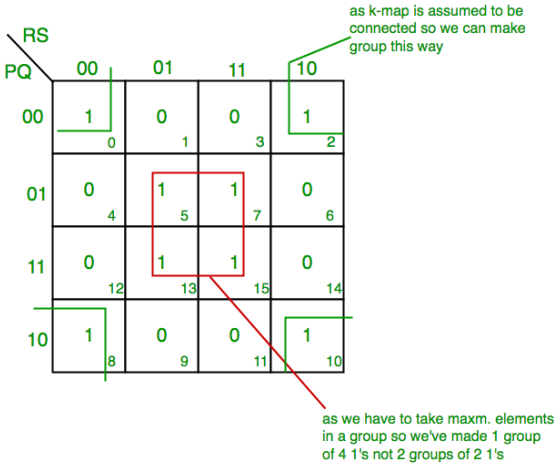


GAMBAR 2.2.1. CONTOH SOP K-MAPS 3 VARIABEL.

Pada kotak merah hasil term adalah $A'C$ dan pada kotak hijau hasil term AB . Dan penjumlahan hasil term merupakan fungsi minimisasi logic ($A'C$ +AB).

2. K-Maps untuk 4 variabel

   F(P,Q,R,S)=
   $\sum$ (0,2,5,7,8,10,13,15)



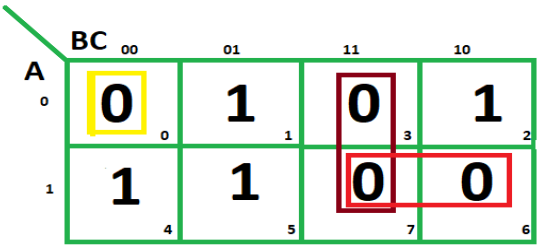
GAMBAR 2.2.2. CONTOH SOP K-MAPS 4 VARIABEL.

Pada kotak merah dihasilkan term QS, dan pada kotak hijau dihasilkan term $Q'S'$. Dan penjumlahan hasil term merupakan fungsi minimisasi logic (QS + $Q'S'$).

Bentuk POS :

1. K-Maps 3 variabel

   F(A,B,C) = π (0,3,6,7)



GAMBAR 2.2.1. CONTOH POS K-MAPS 3 VARIABEL.

Penyelesaian terdapat beberapa tahap :

1. Grup kotak merah , ditemukan hasil term A,B. Kemudian cari komplemen dari hasil term kotak merah A', B'.Dan jumlahkan $(A' + B')$
2. Grup kotak coklat , ditemukan hasil term B,C. Cari komplemen terhadap

hasil term B', C'. .Dan jumlahkan ($B' + C'$).

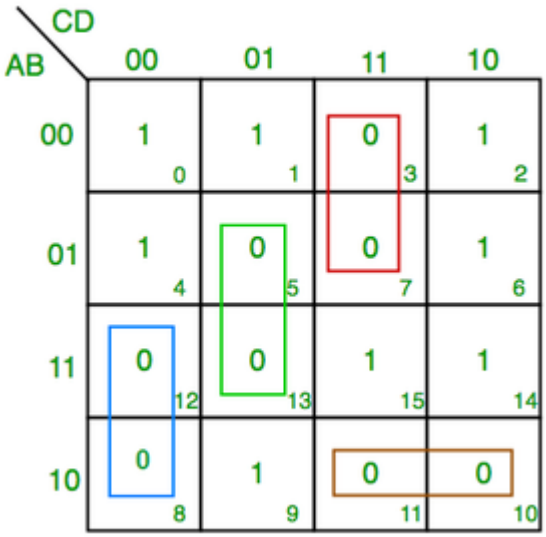
3. Grup kotak kuning , ditemukan hasil term A,B,C. Lakukan komplemen A', B', C'. Dan jumlahkan ($A' + B' + C'$).

   Hasil dari fungsi minimisasi logic

   =$(A' + B')$ ($B' + C'$) ($A' + B' + C'$).

2. K-Maps 4 Variabel

F(A,B,C,D) = π (*3,5,7,8,10,11,12,13*)



GAMBAR 2.2.1. CONTOH POS K-MAPS 4 VARIABEL.

Penyelesaian terdapat beberapa tahap:

1. Grup kotak hijau, hasil term C', D, B. Cari komplemen dan jumlahkan ($C + D' + B'$)
2. Grup kotak merah hasil term A', C, D. Cari komplemen dan jumlahkan ($A + C' + D'$)
3. Grup Kotak Biru, hasil term A, C', D'. Cari komplemen dan jumlahkan ($A' + C + D$)
4. Grup kotak coklat, hasil term A, B', C. Cari komplemen dan jumlahkan ($A' + B + C'$).

   Hasil dari fungsi minimisasi logic

   =$(C + D' + B')(A + C' + D')(A' + C + D)(A' + B + C')$.

2.3 QUINE MCCLUSKEY

Metode Quine McCluskey adalah sebuah metode yang digunakan untuk menyederhanakan fungsi Boolean yang memiliki jumlah peubah yang besar (di atas 6 buah).

Metode ini mengubah sebuah fungsi Boolean menjadi sebuah himpunan bentuk prima, dimana sebanyak mungkin peubah dieliminasi (dihilangkan secara maksimal, sehingga didapat fungsi Boolean yang paling sederhana.

Metode Quine McCluskey memiliki beberapa kelebihan jika dibandingkan dengan metode Karnaugh Maps :

1. Metode McCluskey merupakan metode yang sistematis untuk menyederhanakan fungsi Boole dengan jumlah variabel yang cukup banyak.
2. Metode Quine McCluskey lebih efektif dalam menyederhanakan suatu rangkaian digital dari *8 IC* yang semula dibutuhkan menjadi *3 IC.*

Metode Quine McCluskey juga disebut sebagai metode tabulasi. Metode ini terdiri dari dua langkah :

1. Menentukan term -term sebagai kandidat (*Prime Implicant*)
2. Memilih *prime implicant* untuk menentukan ekspresi dengan jumlah literal sedikit.

Dalam menyederhanakan persamaan dengan metode Quine McCluskey, tahapan yang dilakukan adalah :

1. Menyatakan variabel komplemen dengan 0 , dan variabel bukan komplemen dengan 1.
2. Kelompokkan suku suku berdasarkan jumlah “1”.
3. Kombinasikan suku suku tersebut dengan kelompok lain yang jumlahnya “1” nya berbeda dengan satu, sehingga diperoleh bentuk prima yang sederhana.

Setelah tahap selesai dilakukan, maka tahap selanjutnya yaitu :

1. Mencari *Prime implicant* : term yang menjadi calon term yang akan terdapat dalam fungsi sederhana.

2. Memilih *Prime Implicant* yang memiliki jumlah literal paling sedikit.

Berikut contoh penyederhanaan persamaan rangkaian logika dengan Quine McCluskey [5] :

$$Y(A,B,C,D) = \sum m\ (0,1,3,7,8,9,11,15)$$

Tahap 1 : Ubah persamaan kedalam bentuk biner dengan panjang biner sesuai variabel dan dikelompokkan berdasarkan jumlah digit "1" dari yang terkecil sampai yang terbesar.

Tabel 2.3.1. Pengubahan Ke Biner

0	0000
1	0001
3	0011
7	0111
8	1000
9	1001
11	1011
15	1111

Tabel 2.3.2. Pengelompokkan Berdasarkan Jumlah "1"

Group	Minterm	Bin Rep (ABCD)
0	M0	0000
1	M1	0001
	M8	1000
2	M3	0011
	M9	1001
3	M7	0111
	M11	1011
4	M15	1111

Tahap 2 : Lakukan pencarian minterm yang memiliki perbedaan satu di kelompok berdekatan yang diganti dengan tanda (-). Tahap kedua ini disebut dengan *prime implicant* dalam n-1. Dan dilanjutkan ke tahap 3 untuk melakukan *prime implicant* yang terdiri dari n-2 jika sudah diberi tanda "v" pada tabel .

Tabel 2.3.3.Pengecekkan Sebelum Prime Implicant 1

Group	Minterm	Bin Rep (ABCD)	
0	M0	0000	V
1	M1	0001	V
	M8	1000	V
2	M3	0011	V
	M9	1001	V
3	M7	0111	V
	M11	1011	V
4	M15	1111	V

Tabel 2.3.4. Prime Implicant 1

Group	Matched Pairs	Bin . Rep
0	M0-M1	000-
	M0-M8	-000
1	M1-M3	00-1
	M1-M9	-001
	M8-M9	100-
2	M3-M7	0-11
	M3-M11	-011
	M9-M11	10-1
3	M7-M15	-111
	M11-M15	1-11

Tahap 3 : Buat tabel baru untuk melakukan prime implicant dengan selisih yang akan ditukar 1 sehingga akan membentuk prime implicant yang lebih sederhana .

Tabel 2.3.5. Prime Implicant II

Group	Matched Pairs	Bin . Rep (ABCD)	Prime Impicant
0	M0-M1-M8-M9	-00-	$\overline{B}\overline{C}$
	M0-M8-M1-M9	-00-	
1	M1-M3-M9-M11	-0-1	$\bar{B}D$
	M1-M9-M3-M11	-0-1	
2	M3-M7-M11-M15	--11	CD
	M3-M11-M7-M15	--11	

Tahap 4 : Buat tabel baru untuk menemukan essential prime implicant sehingga lebih sederhana . Dimana *prime implicant* yang telah ditemukan pada tabel V akan dihubungkan ke minterm awal . Jika terdapat minterm yang

sesuai pada *prime implicant* yang ditemukan ditandai dengan “x” .

Tabel 2.3.6. Penyederhanaan Prime Implicant

Prime Implicant	Minterm Involved	0	1	3	7	8	9	11	15
	0,1,8,9	X*	x			X*	x		
	1,3,9,11		x	X			x	x	
	3,7,11,15			x	X*			x	X*

Tahap 5 : Apabila Setiap kolom memiliki lebih satu tanda “x” dan tidak ada baris yang mendominasi , maka pilih salah satu (utamakan pilih *prime implicant* yang lebih banyak mencakup minterm/maxterm yang belum tercakup oleh rima implicant sebelumnya). Dan pilih *prime implicant* yang memiliki 1 buah “X” pada kolom minterm .

Sehingga dapat dihasilkan persamaaan sederhananya :

$$F = \overline{BC} + \overline{CD}$$

dengan persamaan awal :

$$Y = \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}C\overline{D} + \overline{A}B\overline{C}\overline{D} + \overline{A}BCD + A\overline{B}\overline{C}\overline{D} + A\overline{B}CD + AB\overline{C}D + ABC$$

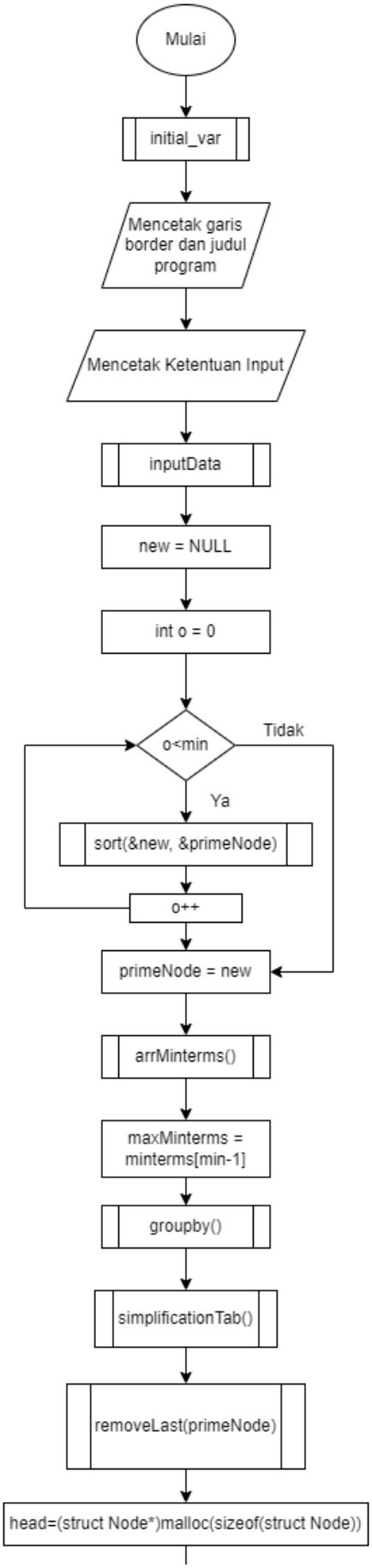
3. SIMULASI PROGRAM

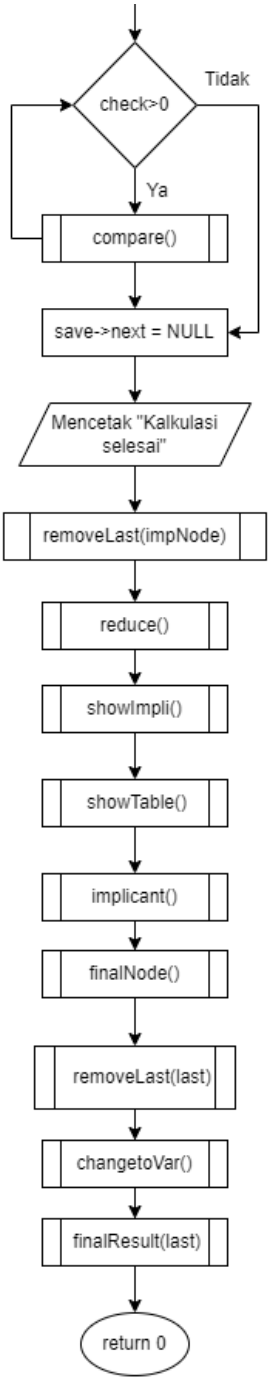
Simulasi program minimisasi logic dengan metode Quine-McCluskey dikarenakan metode ini lebih terstruktur dalam penyelesaiannya dan mudah untuk diimplementasikan ke dalam program. Selain itu, jika dibandingkan dengan metode Karnaugh-Map yang hanya bisa dilakukan maksimal untuk 4 variabel, metode ini dapat digunakan tanpa batasan variabel. Namun dalam program ini dibuat batasan hingga 26 variabel karena mengikuti jumlah abjad yang ada. Referensi dari kode yang dibuat berasal dari tautan https://github.com/bp274/Tabulation-method-Quine-McCluskey-

3.1 FLOWCHART

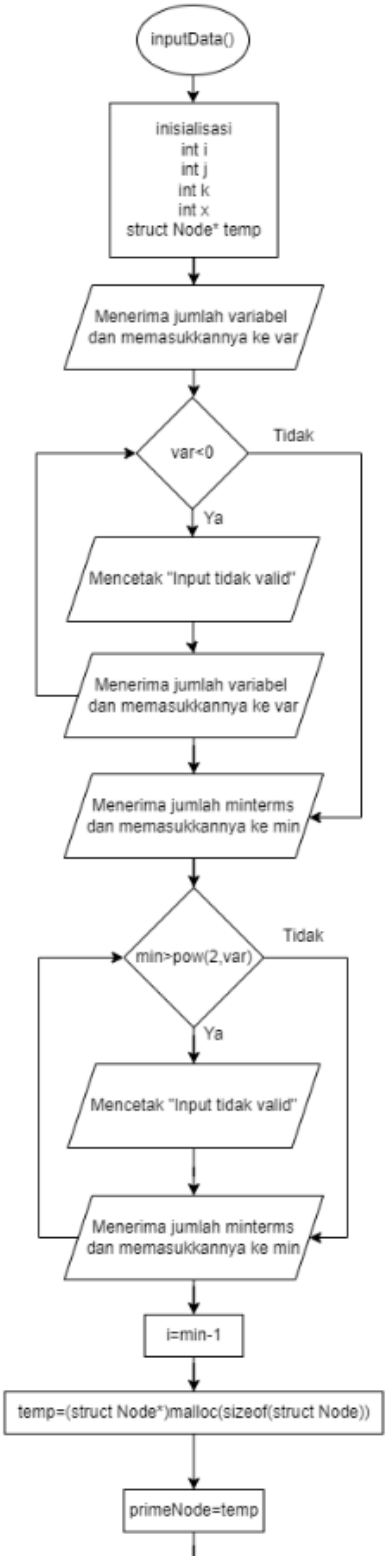
Flowchart setiap fungsi yang terdapat dalam program .

1. Fungsi main ( )

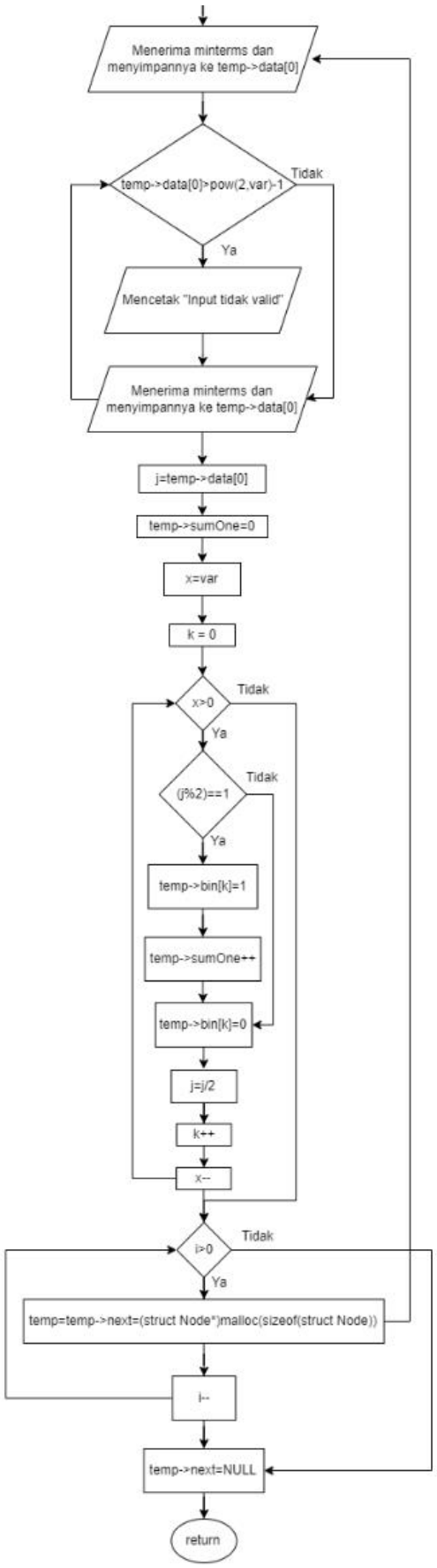


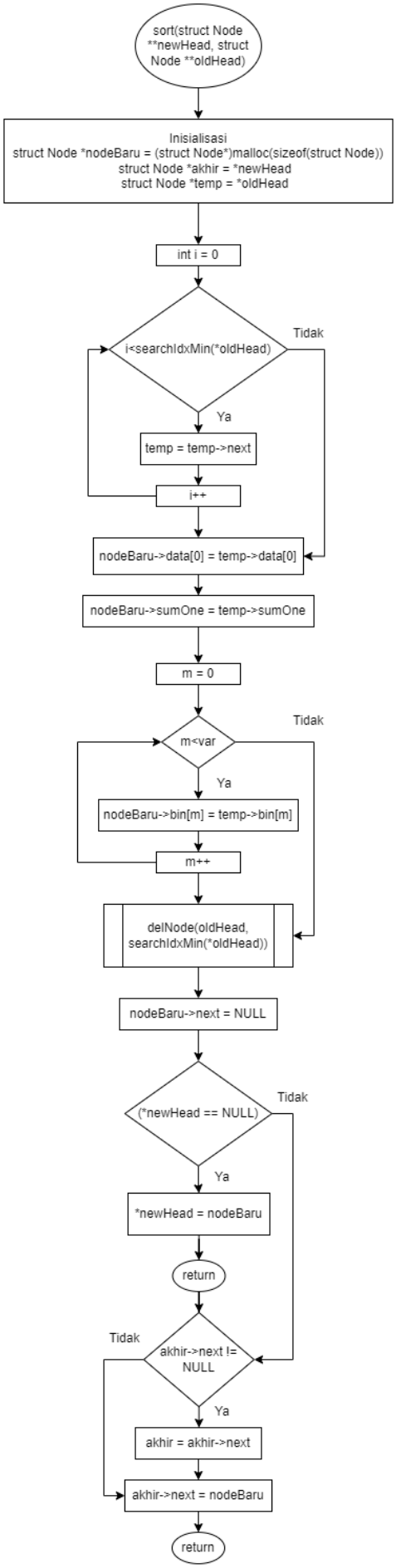


GAMBAR 3.1.1. ALUR FUNGSI MAIN( )



2. Fungsi inputData ( )



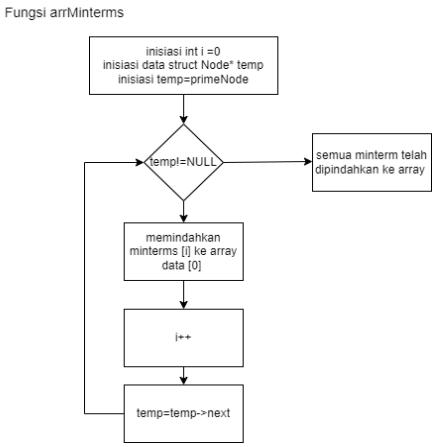


GAMBAR 3.1.2. ALUR FUNGSI INPUTDATA( )

3. Fungsi sort ( )

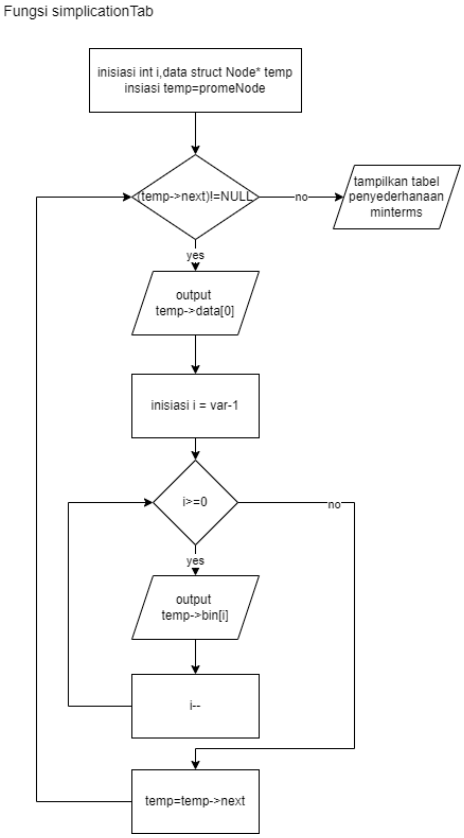
GAMBAR 3.1.3. ALUR SORT( )

4. Fungsi arrMinterms ( )



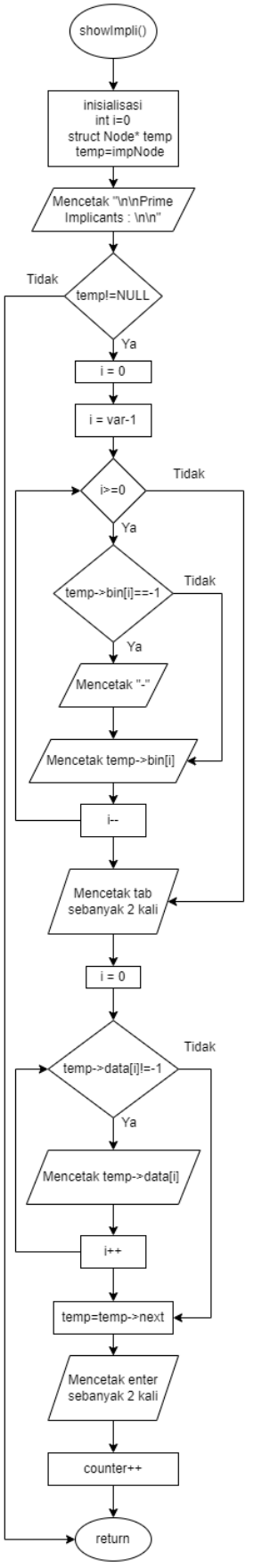
GAMBAR 3.1.4. ALUR FUNGSI ARRMINTERMS( )

5. Fungsi simplicationTab ( )



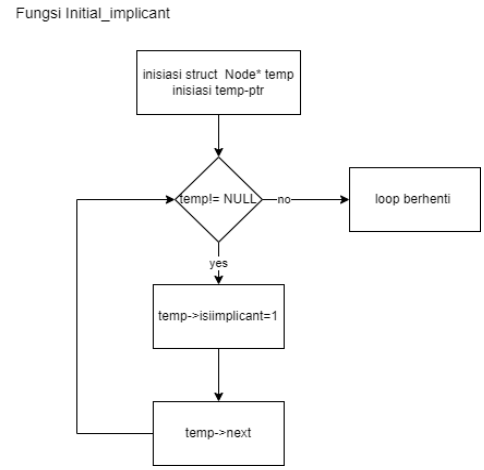
GAMBAR 3.1.5. ALUR SIMPLICATIONTAB ( )

6. Fungsi showImpli ( )



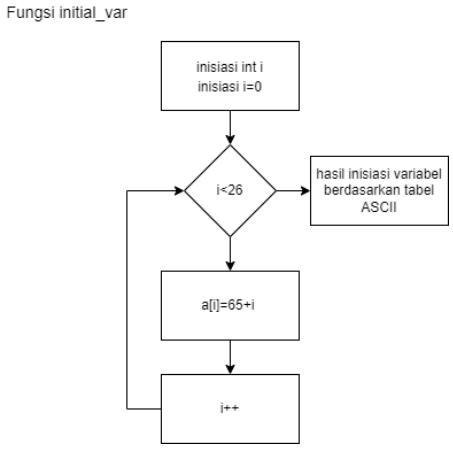
GAMBAR 3.1.6. ALUR FUNGSI SHOWIMPLI( )

7. Fungsi initial_implicants ( )



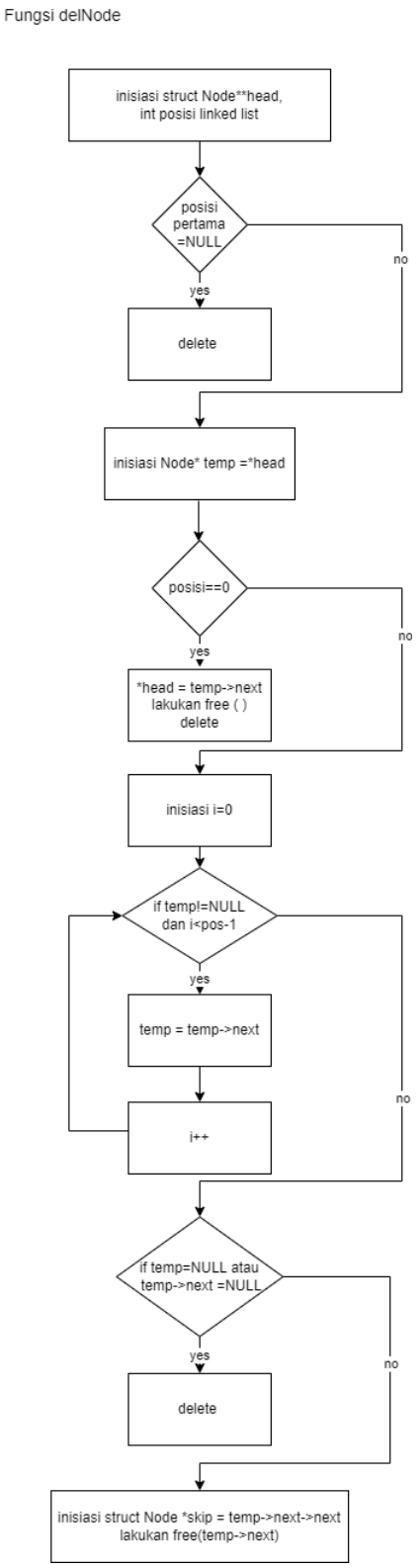
GAMBAR 3.1.7. ALUR FUNGSI INITIAL_IMPLICANTS ( )

8. Fungsi initial_var ( )



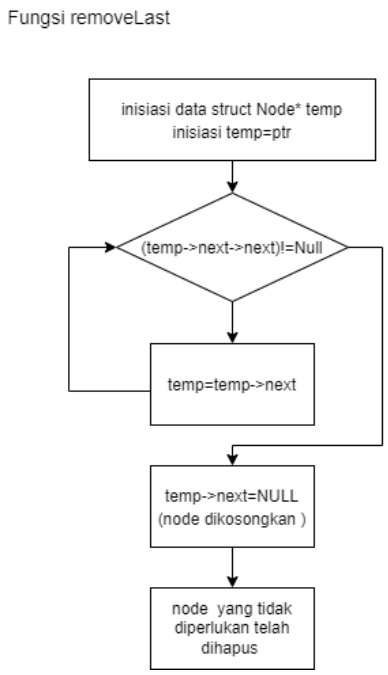
GAMBAR 3.1.8. ALUR FUNGSI INITIAL_VAR ( )

9. Fungsi delNode ( )



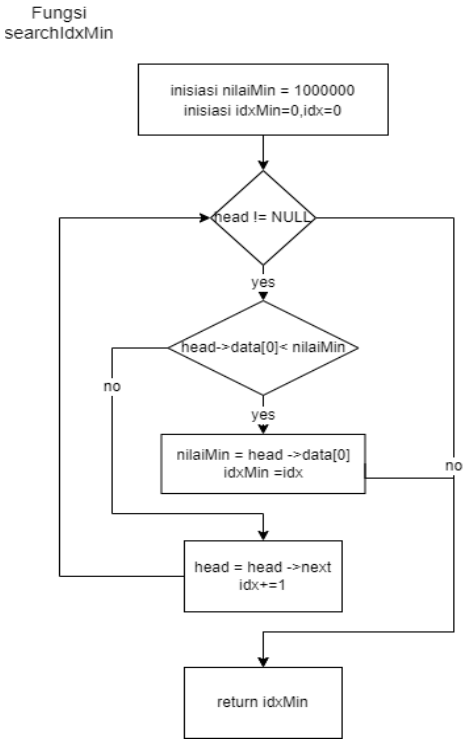
GAMBAR 3.1.9. ALUR FUNGSI DELNODE ( )

10. Fungsi removeLast ( )



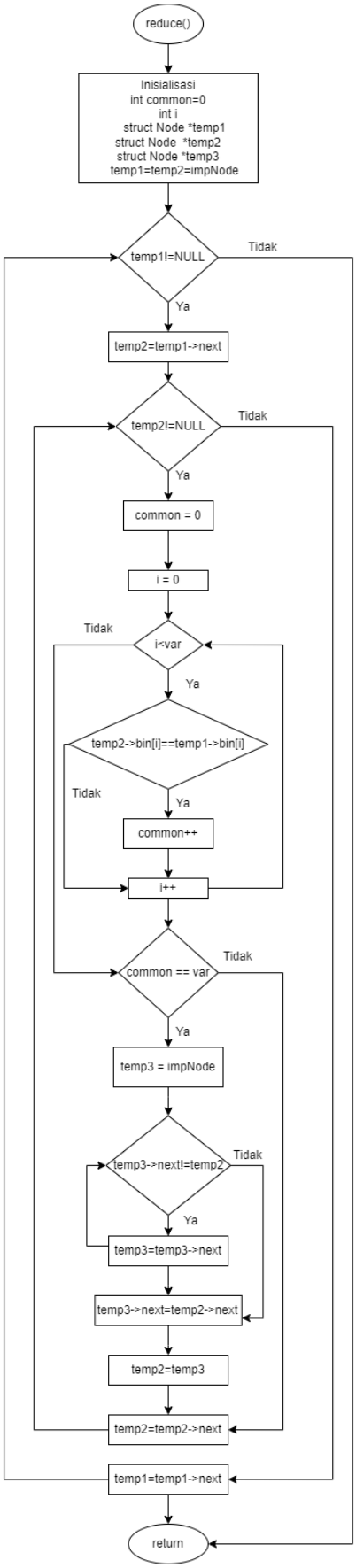
GAMBAR 3.1.10. ALUR FUNGSI REMOVELAST ( )

11. Fungsi searchIdxMin ( )



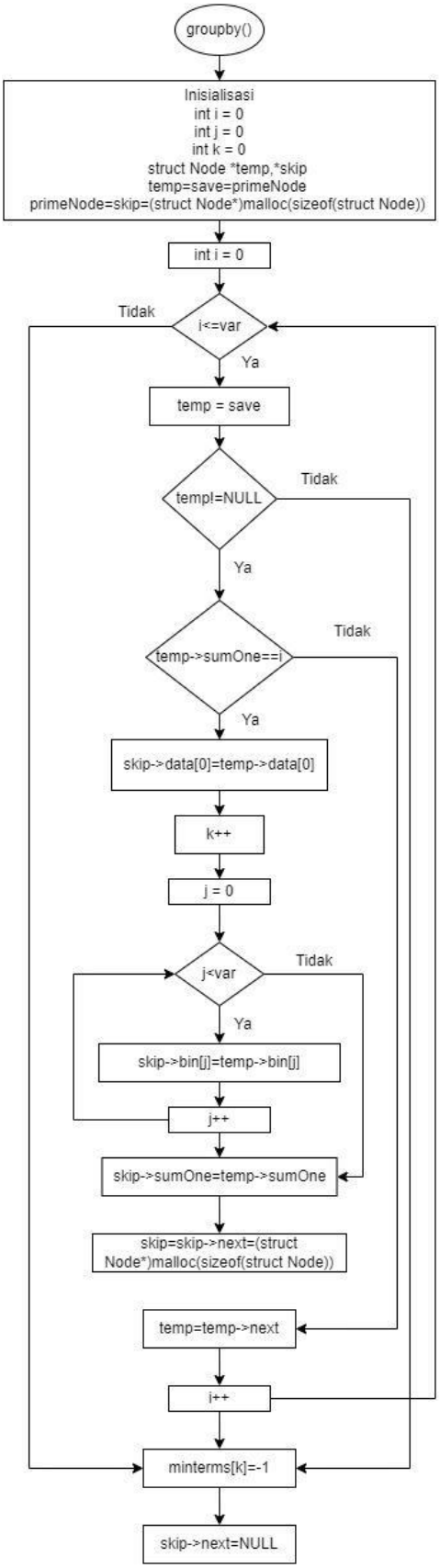
GAMBAR 3.1.11. ALUR FUNGSI SEARCHIDXMIN ( )

12. Fungsi reduce ( )



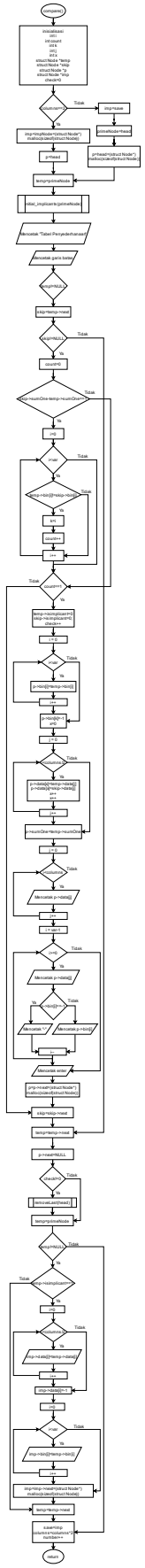
GAMBAR 3.1.4. ALUR FUNGSI REDUCES( )

13. Fungsi groupby ( )



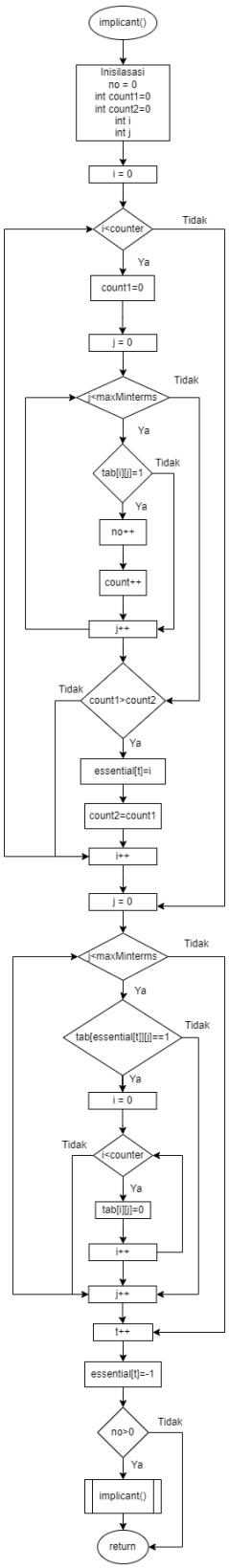
GAMBAR 3.1.13. ALUR FUNGSI GROUPBY( )

14. Fungsi compare ( )



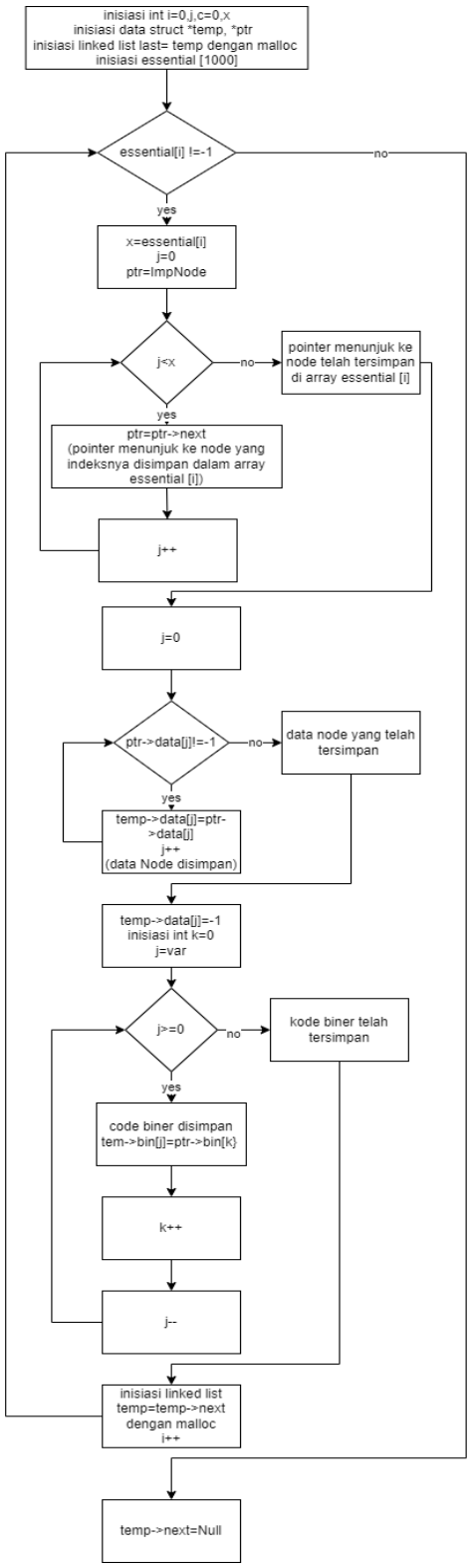
GAMBAR 3.1.14. ALUR FUNGSI COMPARE ( )

15. Fungsi implicant( )



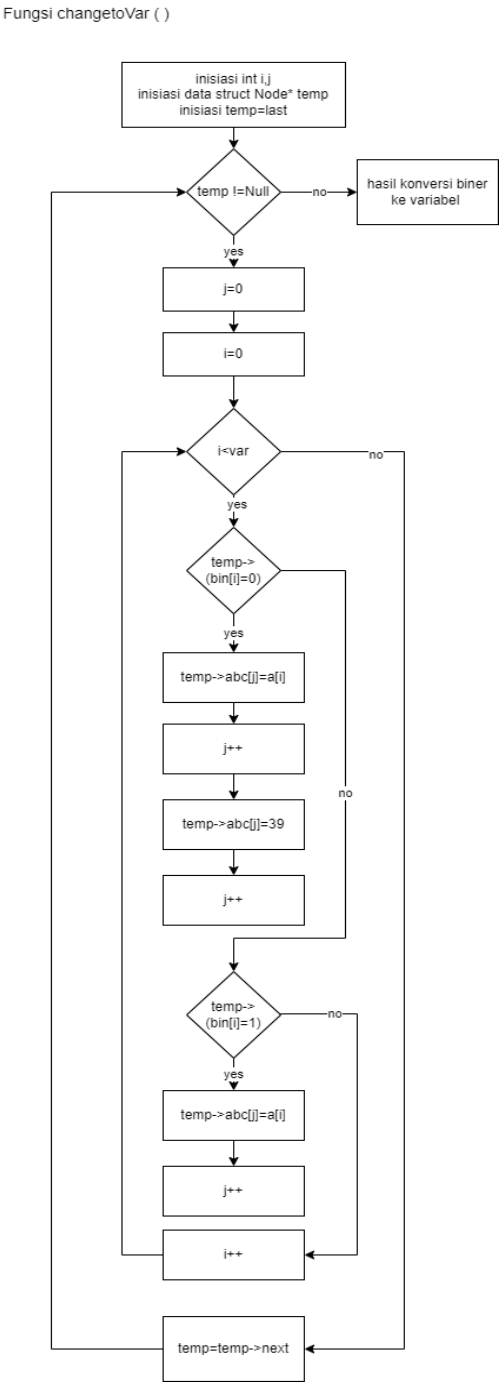
GAMBAR 3.1.15. ALUR FUNGSI IMPLICANT( )

16. Fungsi finalNode ( )



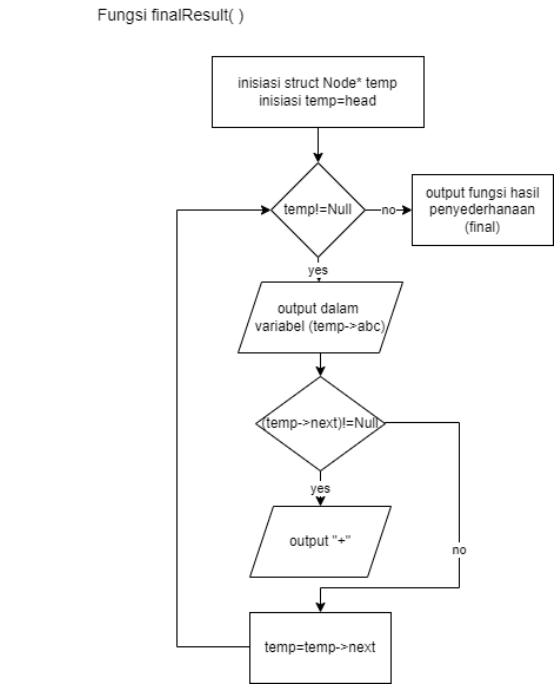
GAMBAR 3.1.16. ALUR FUNSGI FINALNODE ( )

17. Fungsi changetoVar ( )



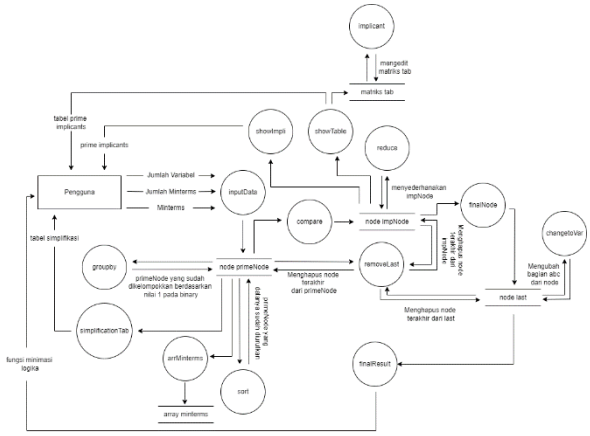
GAMBAR 3.1.17. ALUR FUNSGI CHANGETOVAR ( )

18. Fungsi finalResult ( )



GAMBAR 3.1.18. ALUR FUNGSI FINALRESULT ( )

3.2 DATA FLOW DIAGRAM



GAMBAR 3.2. DFD PROGRAM

Program dimulai dengan user menginput data kemudian akan diolah dengan fungsi fungsi yang terdapat pada Gambar 3.2 sehingga menghasilkan output tabel penyederhanaan, prime implicant, tabel prime implicant , dan hasil fungsi minimisasi logic (berupa char). Fungsi yang terdapat dalam DFD akan dijelaskan pada point 3.3 .

3.3 PROGRAM

1. Input

Program diawali dengan user menginput jumlah variabel (biner) , jumlah minterms, dan data minterms sesuai dengan inputan jumlah minterms. Dalam program ada batasan yang dilakukan yaitu :

1) Variabel input maksimal adalah 26.
2) Jumlah minterms maksimal 2^variabel
3) Elemen minterms tidak lebih dari (2^variabel)-1.

2. Proses

Jalannya proses pada program dapat dilihat pada Gambar 3.1.1. Untuk fungsi yang akan digunakan dalam program dapat dilihat sebagai berikut :

a. Fungsi main ( )

Fungsi utama program yang mencakup penerimaan input, kemudian melakukan konversi minterms dari desimal ke biner dan fungsi-fungsi lainnya untuk melakukan pemrosesan program. Fungsi yang terdapat dalam fungsi main ( ) adalah :

a. fungsi inputData( )

Menerima input berupa jumlah variabel (biner) , jumlah minterms, dan data minterms sesuai dengan inputan jumlah minterms

b. fungsi sort( )

Mengurutkan data minterms dari yang terkecil ke yang terbesar.

c. fungsi arrMinterms( )

Menyimpan minterm ke dalam sebuah array.

d. fungsi groupby( )

Mengelompokkan minterms berdasarkan jumlah angka “1” yang dimiliki.

e. fungsi simplificationTab( )

Menampilkan tabel penyederhanaan minterms.

f. fungsi showImpli()

Menampilkan prime implicants dari minterms

g. fungsi initial_implicants

Menginisialisasi setiap minterms sebagai implicants

h. fungsi initial_var()

Menginisalisasi sebuah array variabel dalam bentuk decimal berdasarkan tabel ASCII.

i. fungsi delNode( )

Menghapus node berdasarkan indeks tertentu

j. removeLast()

Menghapus node tambahan yang tidak diperlukan.

k. searchIdxMin

Mencari indeks dari nilai terkecil data pada suatu node.

l. fungsi reduce ( )

Mengurangi prime implicants yang sama

m. fungsi compare( )

Membandingkan minterms dengan minterms lain, jika ditemukan sebuah perubahan pada binary yang berdekatan akan ditandai.

n. fungsi showTable ( )

Menampilkan tabel penemuan prime implicants pada minterms.

o. fungsi implicants( )

Mencari prime implicant

p. fungsi finalNode( )

Membentuk node baru yang berisikan hasil prime implicant.

q. fungsi changetoVar( )

Mengubah minterms dari biner menjadi sebuah variabel dalam alfabet.

r. fungsi finalResult( )

Mencetak hasil minimasi logic dalam bentuk suatu persamaan fungsi

3. Output

Program akan menampilkan output berupa :

1). Tabel penyederhanaan minterms yang disusun berdasarkan pengelompokkan angka 1

2). Daftar prime implicant yang ditemukan.

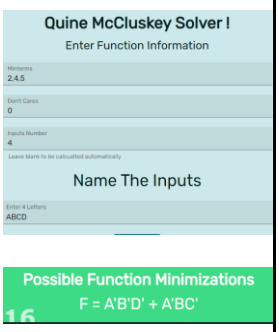
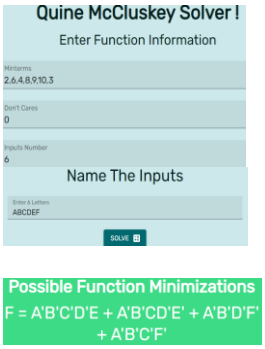
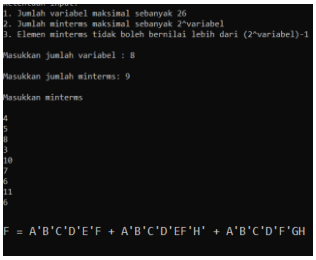
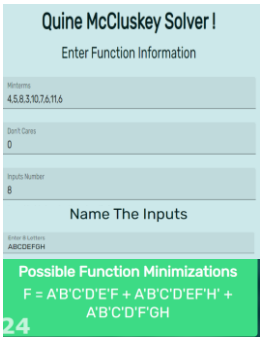
3). Tabel persebaran prime implicant berdasarkan minterms.

4). Hasil dari fungsi penyederhanaan logika Quine-McCluskey berupa huruf.

3.4 PENGUJIAN

Pengujian dilakukan untuk melihat tingkat akurasi kebenaran dari hasil penyederhanaan yang dilakukan oleh program. Pengujian dilakukan membandingkan hasil dari kode program dengan web https://geeekyboy.github.io/Quine-McCluskey-Solver/#/

Tabel 3.4. Pengujian Kode Program

Output Kode Program	Web McCluskey Solver
	
	
	

3.5 SOURCE CODE

Source code dapat diakses pada tautan https://github.com/yisyista/Tugas-Besar-PMC

4. KESIMPULAN DAN LESSON LEARNED

Dalam minimisasi logic atau penyederhanaan rangkaian logis ada 3 metode yaitu Boolean Algebra , Karnaugh Maps , dan Quine-

McCluskey. Metode Boolean Algebra dilakukan dengan menggunakan properties, postulates, dan theorem yang ada. Metode karnaugh Maps dilakukan dengan menggunakan tabel . Jumlah kolom dan bentuk tabel disesuaikan dengan jumlah variabel. Pada metode K-Maps terdapat 2 cara yaitu SOP (Sum of Products) dan POS (Product of Sum) yang digunakan disesuaikan dengan keadaan permasalahan. Dan metode Quine- McCluskey dilakukan dengan menggunakan metode tabular dan prime implicant yang cocok digunakan untuk fungsi logika yang lebih dari 6 variabel.

Dalam penerapan bahasa C , metode yang dipilih adalah Quine- McCluskey . Sehingga program mampu untuk melakukan hal berikut :

1. Program mampu menyederhanakan fungsi logika dengan jumlah peubah yang besar.
2. Program mampu untuk menyederhanakan fungsi logika dengan prinsip metode tabular.
3. Program dapat menampilkan step by step yang sesuai dengan metode quine McCluskey.

5. DAFTAR PUSTAKA

[1] Purba, Desinta, et al. "Efisiensi Komponen Rangkaian Logika dengan Menggunakan Metode Penyederhanaan Quine-McCluskey." *Citra Sains Teknologi*, vol. 1, 2021, pp. 43-49. https://publisher.yccm.or.id/index.php/cisat/article/view/30/33. Accessed 18 May 2022 .

[2] Rahardjo, A. P. (2012, July 27). Penyederhanaan Fungsi Boolean Dengan Metode Quine-McCluskey. 27-32.

[3] *Learn Boolean Algebra*. (n.d.). Boolean Algebra Solver. Retrieved May 18, 2022, from https://www.boolean-algebra.com/learn/.

[4] https://www.youtube.com/watch?v=WW-NPtIzHwk , 18-Mei-2022 , 09:00 - 14:00 WIB.

[5] https://www.youtube.com/watch?v=l1jgq0R5EwQ , 18-Mei-2022 , 09:00 - 14:00 WIB.

[6] *Introduction of K-Map (Karnaugh Map)*. (2021, November 22). GeeksforGeeks. Retrieved May 19, 2022, from https://www.geeksforgeeks.org/introduction-of-k-map-karnaugh-map/.

6. PEMBAGIAN TUGAS

Kayyisa Zaharatulfirdaus :

Kode Program

Laporan

a. Penjelasan fungsi
b. Data Flow Diagram

PPT dan Presentasi

a. Penjelasan Fungsi

Flowchart

a. inputData
b. sort
c. groupby
d. compare
e. reduce
f. showTable
g. Implicant
h. showImpli
i. Main

Putri Alfiyyahdianti :

Laporan

c. Eksplorasi metode 1, 2, dan 3
d. Data Flow Diagram
e. Penjelasan program
f. Pengujian
g. Kesimpulan dan lesson learned

PPT dan presentasi

a. Bagian eksplorasi metode 1, 2, dan 3

Flowchart

a. Initial_implicants
b. SearchIdxMin
c. delNode
d. arrMinterms

e. simplificationTab
f. removeLast
g. initial_var
h. changetoVar
i. finalResult
j. final_Node