**MINIMISASI LOGIC**

**LAPORAN TUGAS BESAR**

Sebagai salah satu bagian dari Tugas Besar mata kuliah Pemecahan Masalah dengan C (EL2008) Kelas 1 pada Semester IV Tahun Akademik 2021/2022

oleh

Adro Anra Purnama 13220005

Surya Dharma 13220027

Fariz Iftikhar Falakh 13220029

Senggani Fatah Sedayu 13220035

A picture containing text

Description automatically generated

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**BANDUNG**

**2022**

**Daftar Isi**

[Laporan Inti 1](#_Toc105415845)

[A. Studi Pustaka 1](#_Toc105415846)

[B. Spesifikasi Program 2](#_Toc105415847)

[C. Flowchart dan Data Flow Diagram 3](#_Toc105415848)

[*1)* *Flowchart* 3](#_Toc105415849)

[*2)* *Data Flow Diagram* 14](#_Toc105415850)

[Kesimpulan dan Lesson Learned 15](#_Toc105415851)

[Pembagian Tugas dalam Kelompok 16](#_Toc105415852)

[Link *Repository* 16](#_Toc105415853)

[Referensi 17](#_Toc105415854)

# Laporan Inti

## Studi Pustaka

Minimisasi logika adalah suatu proses untuk mencari persamaan yang lebih sederhana dari suatu rangkaian logika dengan syarat tertentu. Tujuan dari minimisasi logika adalah mengurangi banyaknya gerbang logika atau sirkuit yang digunakan supaya proses lebih sederhana dan penyelesaian persamaan memakan lebih sedikit waktu.

Metode minimisasi logika yang digunakan oleh kelompok ini adalah metode tabular atau metode Quine-McCluskey. Metode ini dipilih karena pembuatan kodenya lebih mudah dibandingkan dengan metode Karnaugh Map yang lebih bergantung pada visual. Metode tabular juga lebih mudah digunakan dalam menyelesaikan masalah dengan variabel yang banyak dibandingkan dengan Karnaugh Map. Dalam minimisasi logika, metode tabular menggunakan sistem eliminasi di mana pengaplikasian algoritma dapat menggunakan rekursi tanpa mengubah fungsi yang telah ada. Dibandingkan dengan Karnaugh Map yang perlu mengubah letak urutan dari bentuk tabel tergantung dengan banyak variabel dan menjadi lebih kompleks setelah ada 4 variabel, metode tabular lebih mudah digunakan.

Langkah dalam menggunakan metode tabular adalah pertama susun minterm yang diberikan dari yang terkecil hingga yang terbesar dan buat grup berdasarkan jumlah bilangan satu yang ada dalam representasi binernya. Sehiangga, akan ada maksimal grup jika ada variabel Boolean dalam fungsi Boolean atau bit dalam ekuivalen biner dari minterm

Kedua, bandingkan minterm yang ada pada suatu grup dengan minterm lain yang ada pada grup selanjutnya. Jika perbedaan antara minterm satu dengan yang lainnya hanya satu bit, maka gabungkan dua minterm tersebut menjadi sebuah term baru. Tempatkan simbol atau di bit yang berbeda dan biarkan bit lainnya. Selanjutnya ulangi langkah kedua dengan term yang baru terbuat hingga kita mendapatkan semua *prime implicants*.

Ketiga, buat *prime implicant* table. Tabel tersebut terdiri dari kumpulan baris dan kolom. *Prime implicant*s disusun sebagai baris dari tabel dan minterm disusun sebagai kolom dari tabel. Tempatkan '1' di sel yang bisa direpresentasikan oleh *Prime implicant*s serta minterm pada sel tersebut.

Keempat, cari *prime implicant* penting dengan mengamati tiap kolom. Apabila minterm hanya tercakup oleh satu *prime implicant*, maka *prime implicant* tersebut penting. *Prime implicant* tersebut akan menjadi bagian dari penyederhanaan fungsi boolean.

Terakhir, Kurangi tabel *prime implicant* dengan menghilangkan baris dari tiap *prime implicant* penting dan kolom yang bersesuaian dengan minterm yang telah tercakup didalam *prime implicant* penting tersebut. Ulangi langkah keempat untuk mengurangi tabel *prime implicant*. Proses dihentikan ketika semua minterm dari fungsi boolean yang diberikan telah tercakup.

## Spesifikasi Program

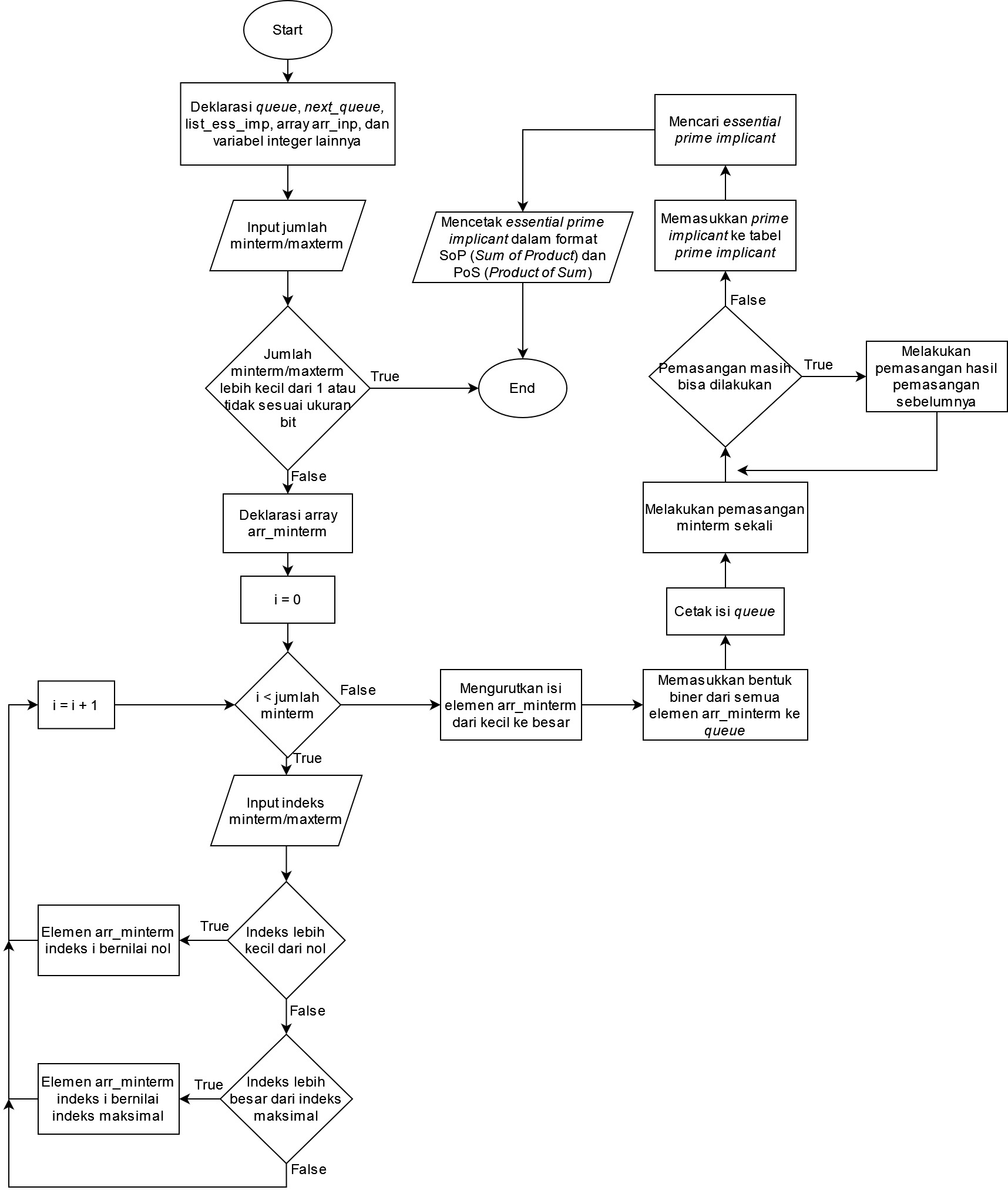
Spesifikasi program minimisasi adalah:

1. Program menerima input jumlah minterm dan indeks minterm (terdapat tambahan input ukuran bit untuk quinecluskey\_bitsSizeInput.c)
2. Program menampilkan hasil minimisasi berdasakran minterm yang diberikan
3. Program bekerja untuk ukuran 1 – 8 bit (1 – 26 bit untuk quinecluskey\_bitsSizeInput.c)
4. Ukuran bit hanya bisa diganti dengan mengganti definisi bitsSize pada *header* kode (hanya untuk quinecluskey.c)
5. Program tidak menerima minterm *don’t care* dan menganggap tidak ada minterm *don’t care*

## Flowchart dan Data Flow Diagram

### *Flowchart*

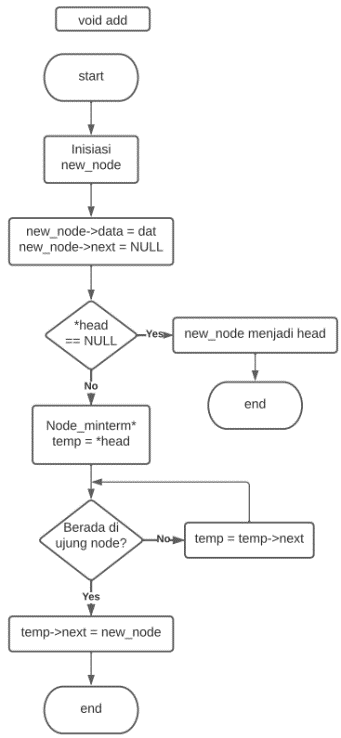
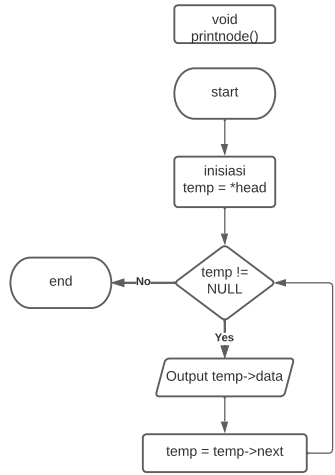
* 1. *Flowchart* algoritma



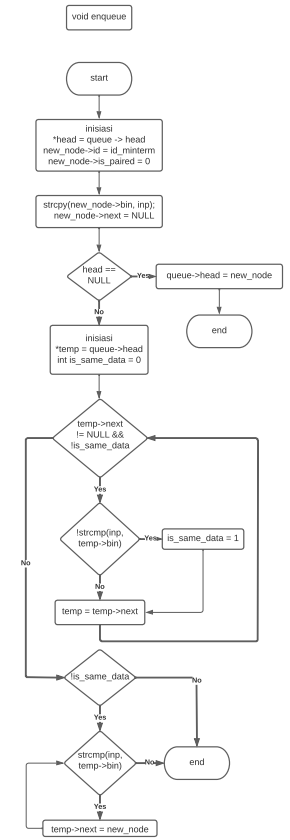
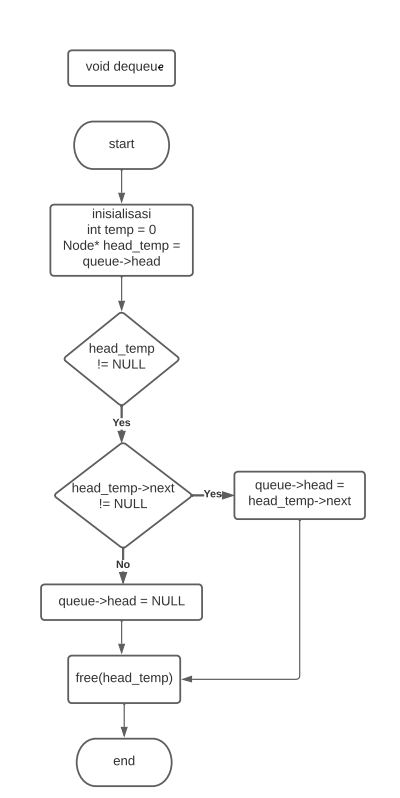
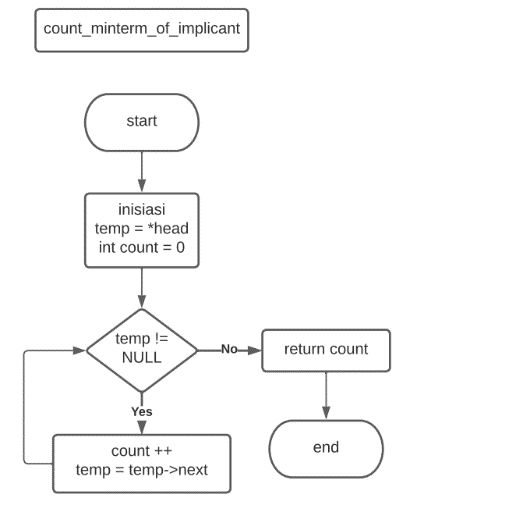
Gambar 1. *Flowchart* program secara keseluruhan

* 1. *Flowchart* fungsi dan prosedur yang digunakan

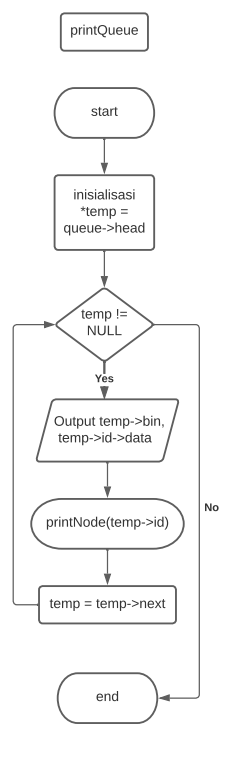
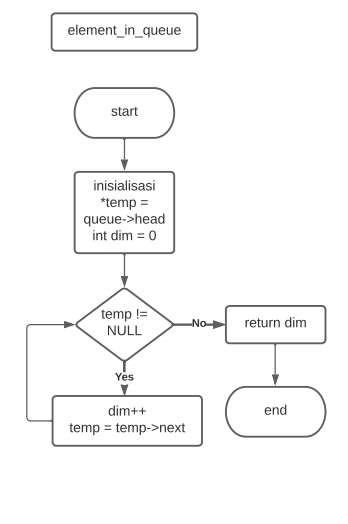
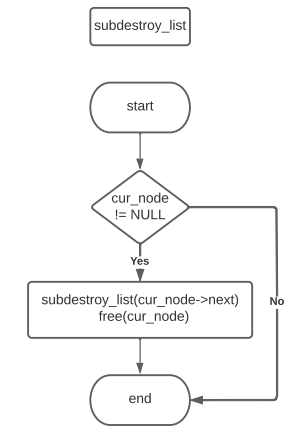
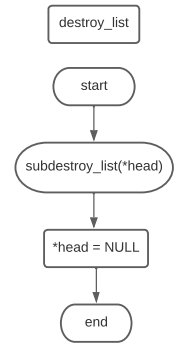
Gambar 2. *Flowchart* prosedur add dan prosedur printNode

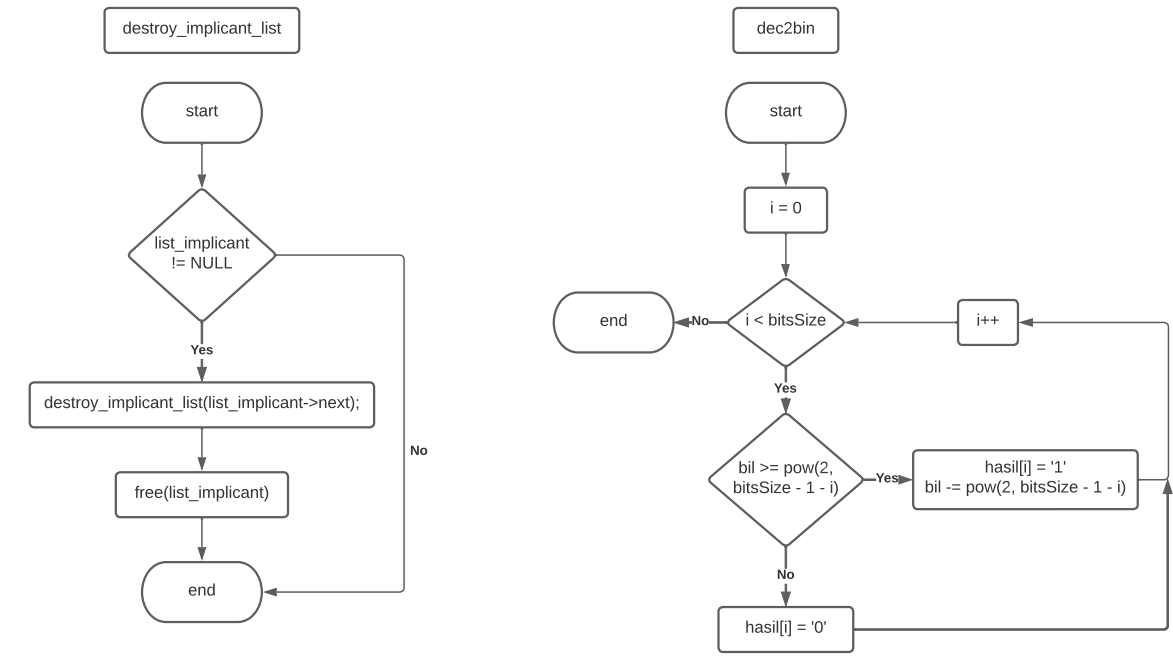


Gambar 3. *Flowchart* fungsi count\_minterm\_of\_implicant, prosedur enqueue, dan prosedur dequeue

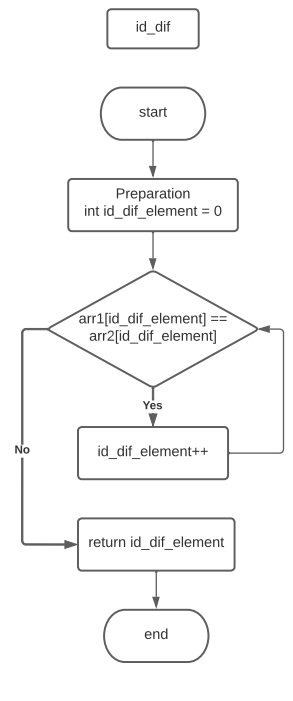
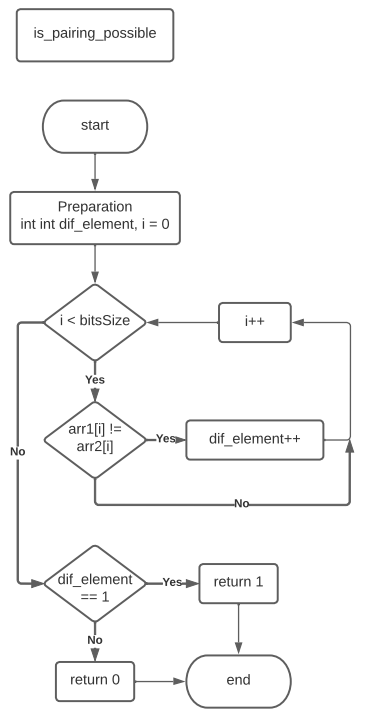


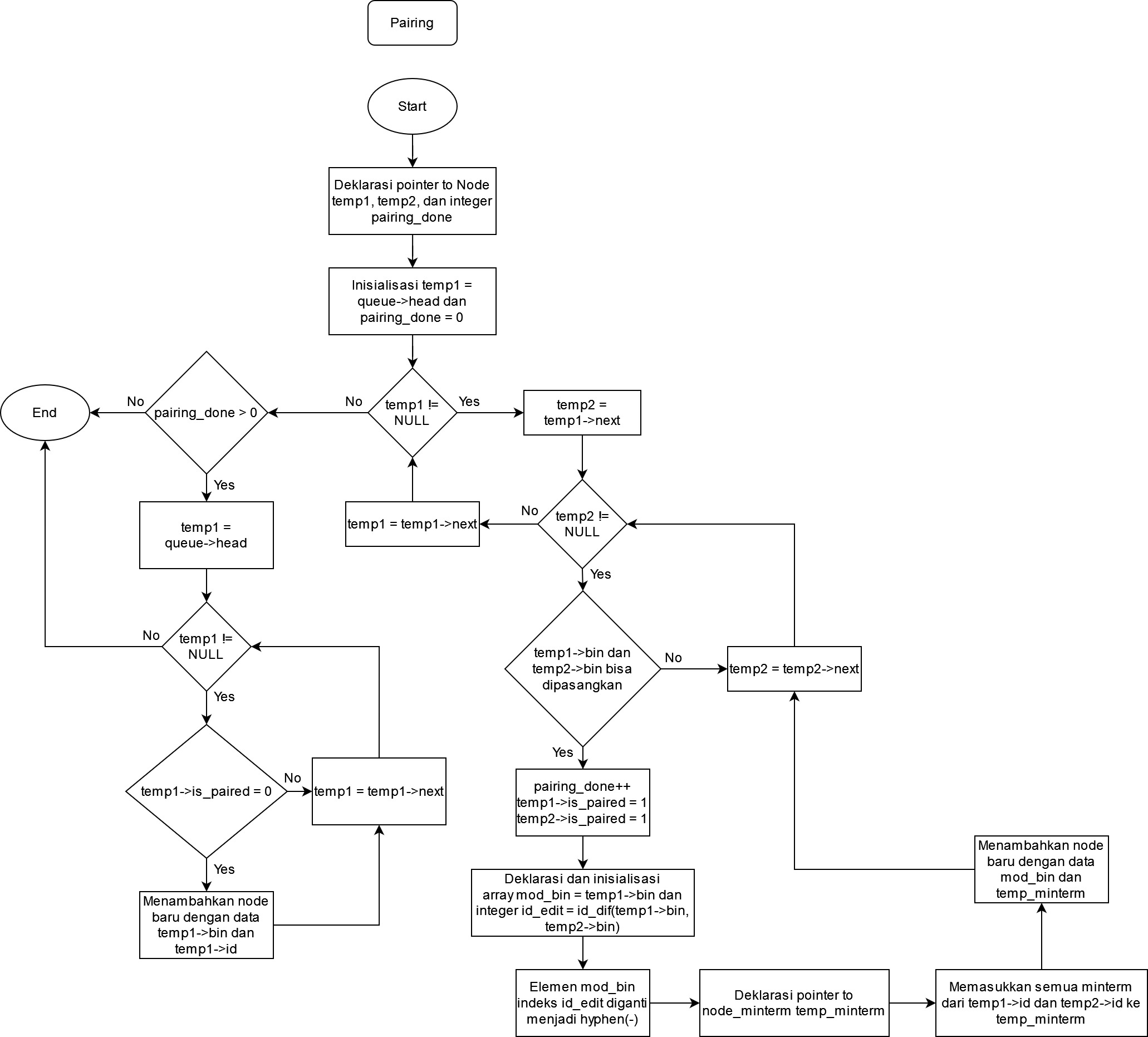
Gambar 4. *Flowchart* prosedur printQueue, fungsi element\_in\_queue, prosedur subdestroy\_list, dan prosedur destroy\_list

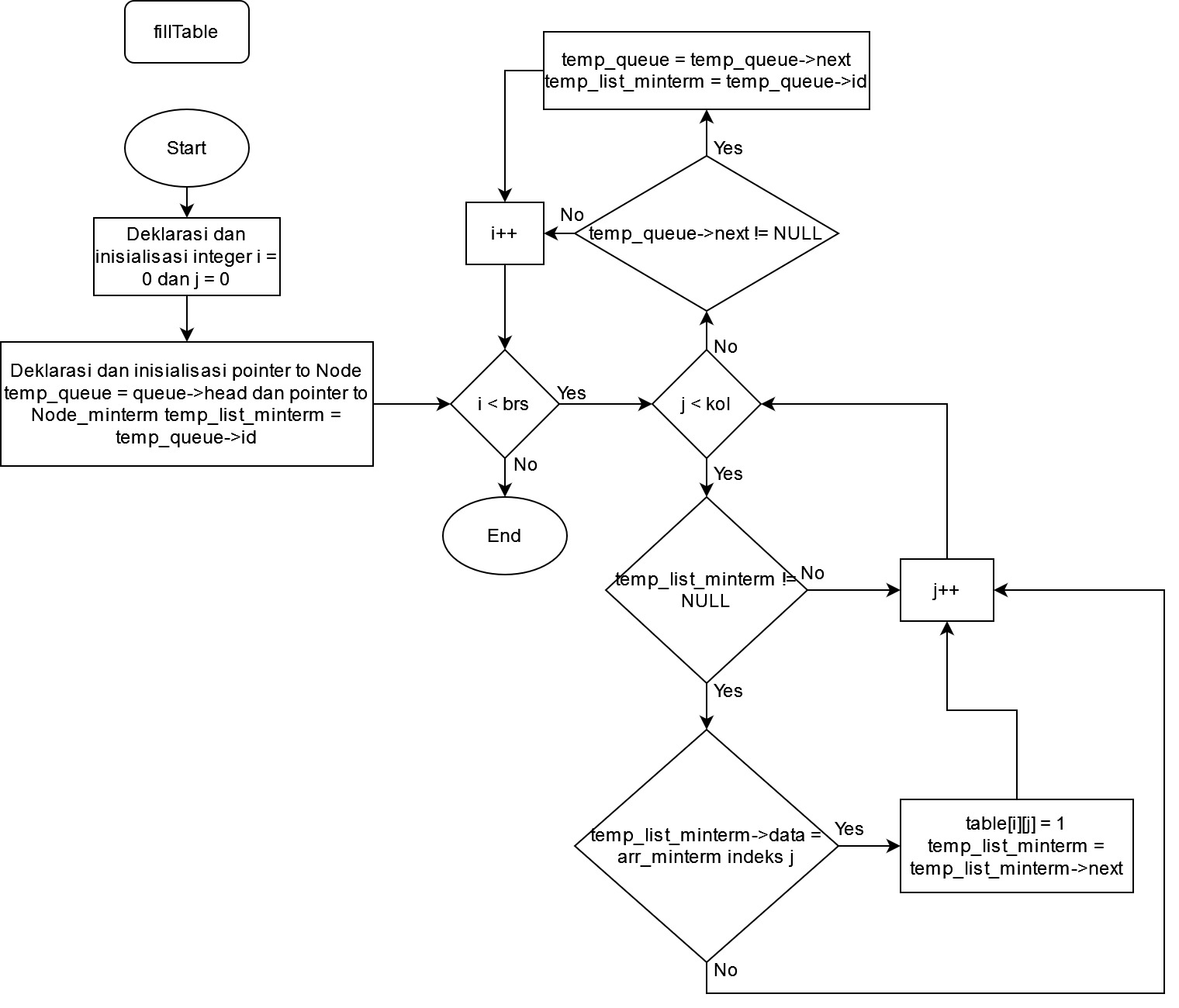


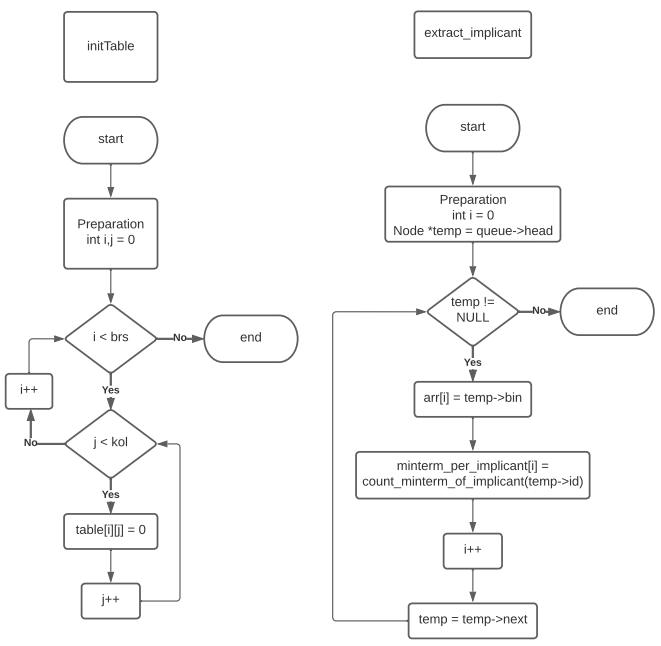
  
Gambar 5. *Flowchart* prosedur destroy\_implicant\_list dan prosedur dec2bin

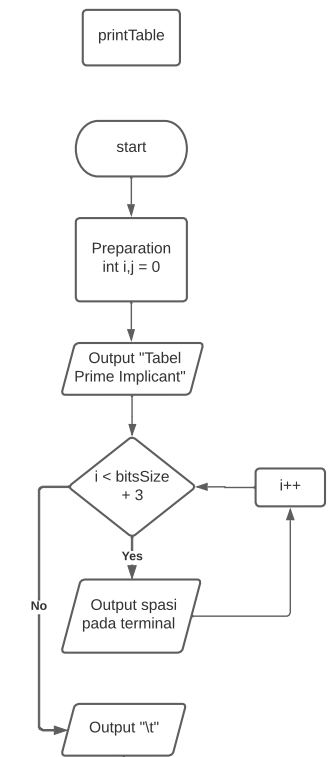
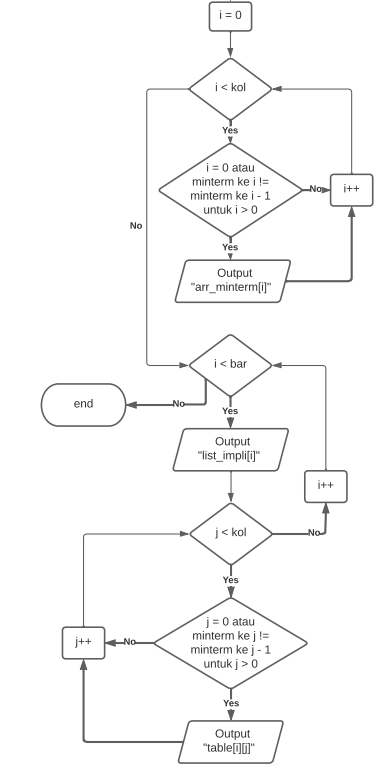
Gambar 6. *Flowchart* fungsi id\_dif dan fungsi is\_pairing\_possible



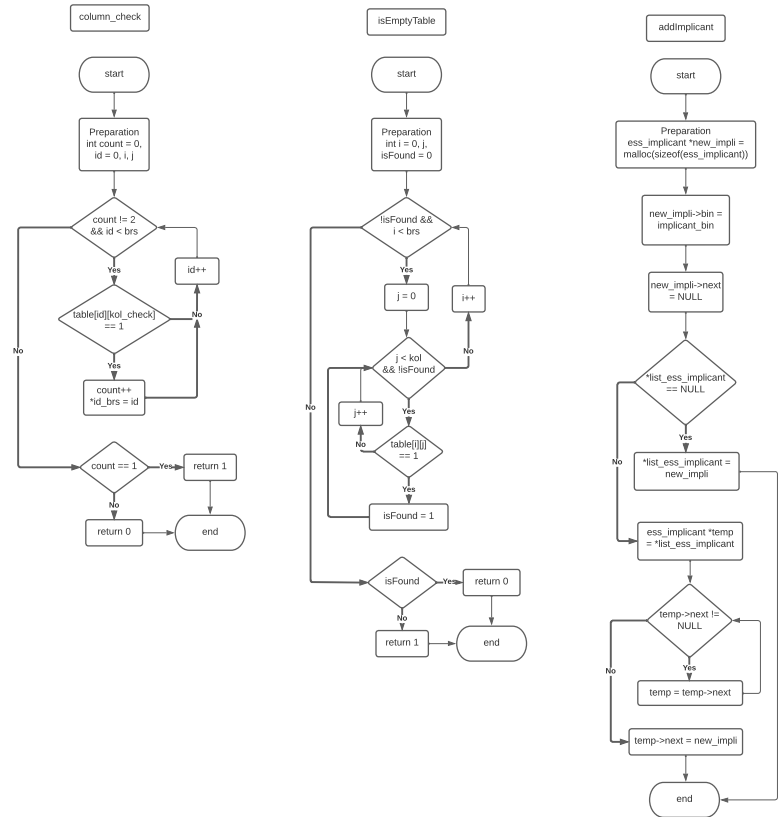
  
Gambar 7. *Flowchart* prosedur pairing

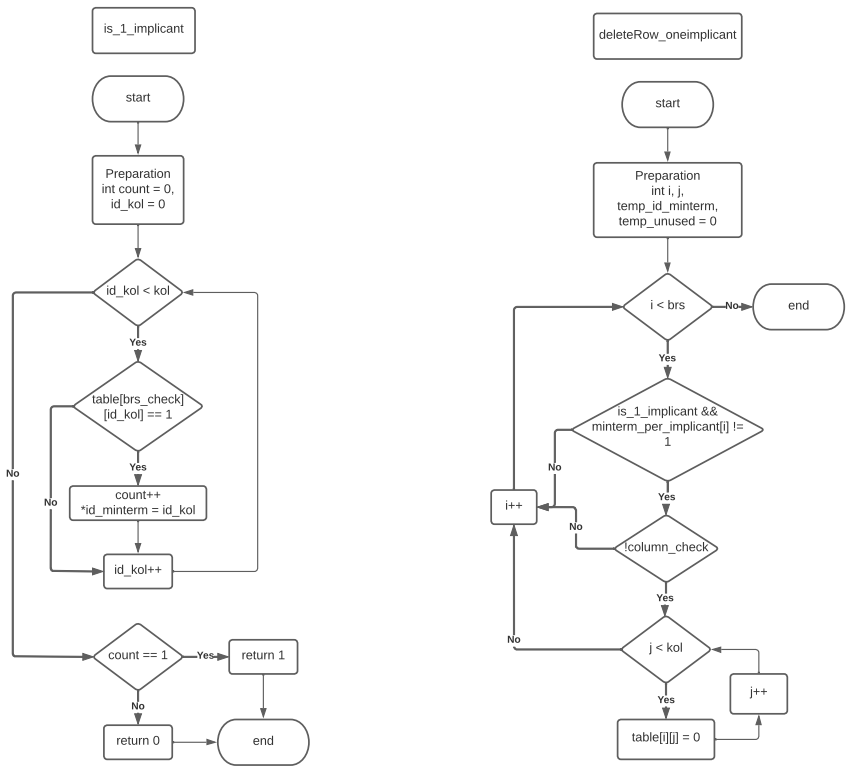
  
Gambar 8. *Flowchart* prosedur fillTable

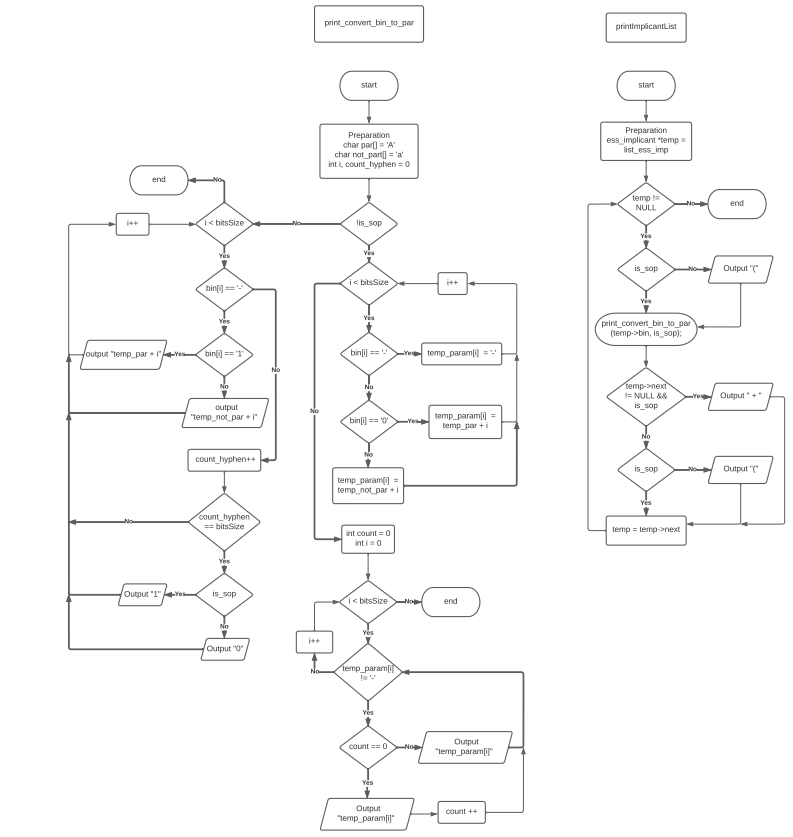
  
Gambar 9. *Flowchart* prosedur initTable dan prosedur extract\_implicant



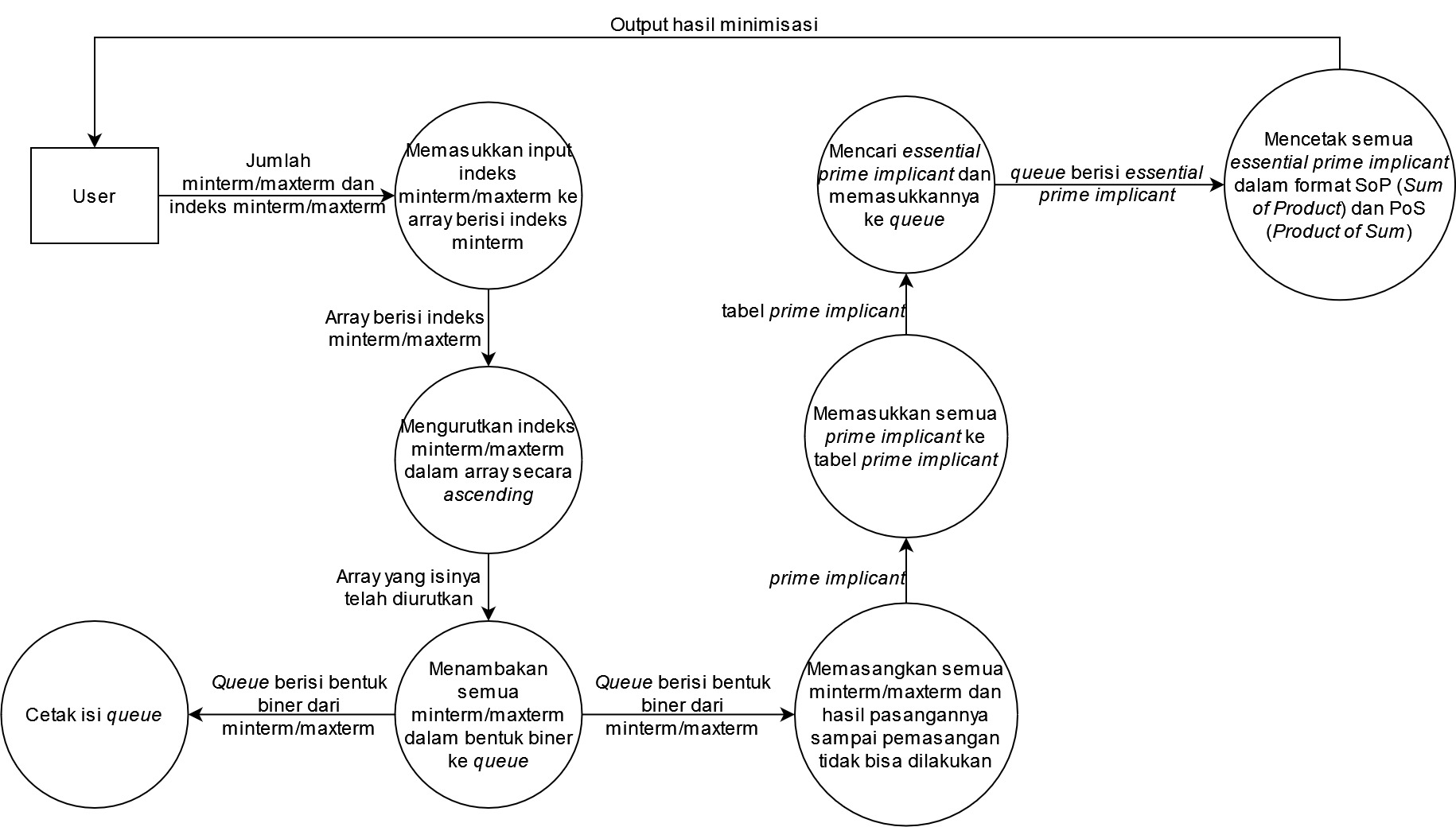
Gambar 10. *Flowchart* prosedur printTable

  
Gambar 11. *Flowchart* fungsi column\_check, fungsi isEmptyTable, dan prosedur addImplicant

  
Gambar 12. *Flowchart* fungsi is\_1\_implicant dan prosedur deleteRow\_oneimplicant

  
Gambar 13. *Flowchart* prosedur print\_convert\_bin\_to\_par dan prosedur printImplicantList

### *Data Flow Diagram*

  
Gambar 14. *Data Flow Diagram* program minimisasi

# Kesimpulan dan Lesson Learned

* Minimisasi merupakan sebuah proses penyederhanaan sebuah fungsi boolean yang sangat bermanfaat, karena dapat mengurangi *cost* sebuah fungsi boolean
* Quine-McCluskey merupakan salah satu metode minimisasi fungsi boolean yang relatif mudah untuk diimplementasikan dalam suatu program karena metode ini menggunakan sistem eliminasi yang berulang sehingga dapat diselesaikan secara rekursif. Selain itu metode ini juga lebih efektif untuk menyederhanakan suatu fungsi boolean yang memiliki banyak variabel
* Deklarasi array sebagai isi suatu *structure* dengan panjang yang bergantung dengan nilai variabel di luar *struct* tidak bisa dilakukan, kecuali dengan *header define*

# Pembagian Tugas dalam Kelompok

|  |  |
| --- | --- |
| Nama Anggota Kelompok | Kontribusi |
| Adro Anra Purnama | Membuat flowchart fungsi dan prosedur |
| Surya Dharma | Membuat *source code* program, *data flow diagram*, dan flowchart algoritma |
| Fariz Iftikhar Falakh | Melengkapi laporan, membuat file presentasi |
| Senggani Fatah Sedayu | Melengkapi laporan |

# Link *Repository*

<https://github.com/DentAlpha/Minimisasi-Quine-McCluskey>

# Referensi

1. Brown S. D. dan Vranesic Z. G, *Fundamentals of digital logic with VHDL design*, 2000.