

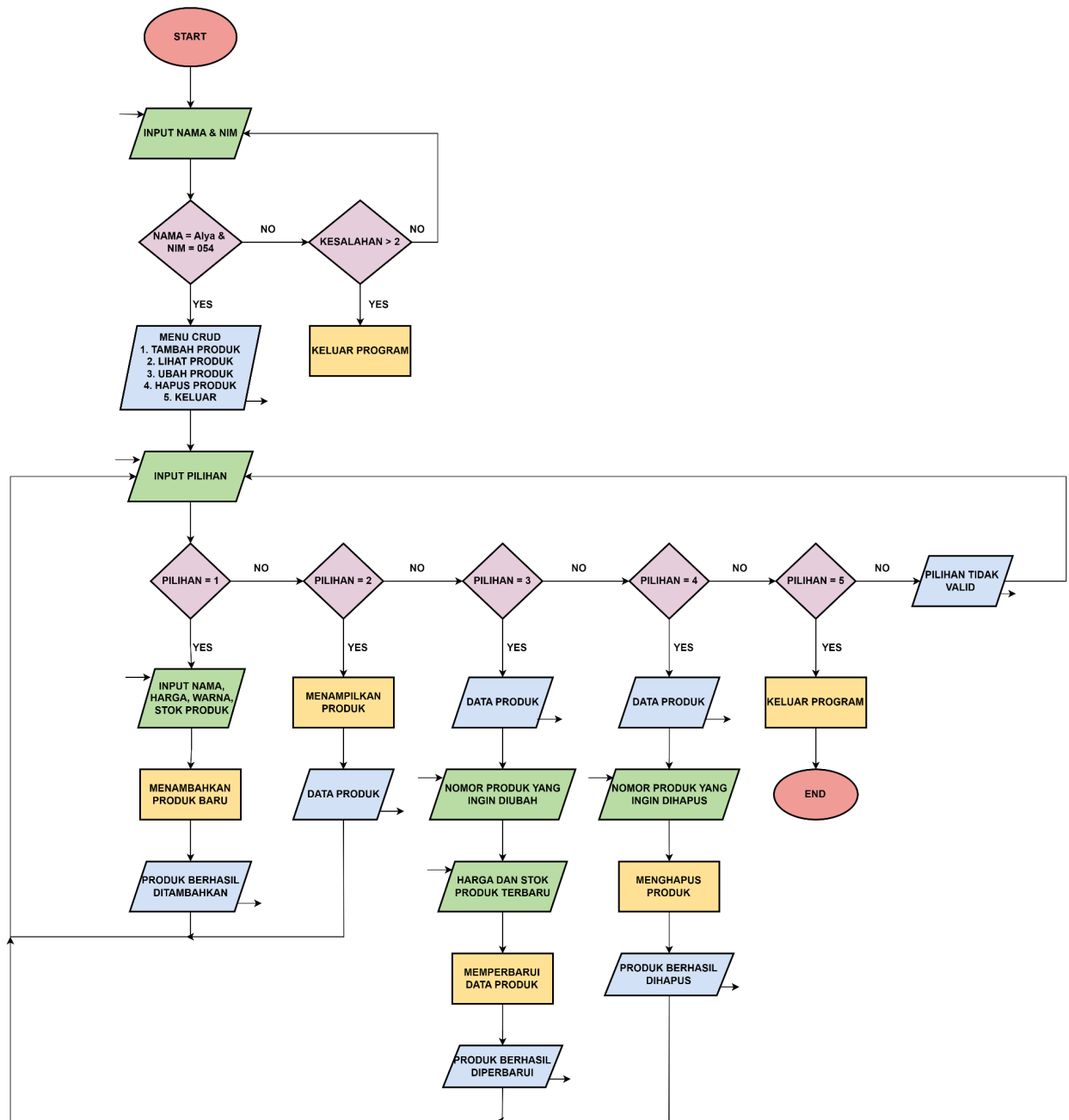
**LAPORAN PRAKTIKUM**  
**POSTTEST 3**  
**ALGORITMA PEMROGRAMAN LANJUT**



**Disusun oleh:**  
**Alya Mayasha (2409106054)**  
**Kelas (B1'24)**

**PROGRAM STUDI INFORMATIKA**  
**UNIVERSITAS MULAWARMAN**  
**SAMARINDA**  
**2025**

# 1. Flowchart



Gambar 1.1 Flowchart Program Manajemen Tas Rajut

## 2. Analisis Program

Program ini adalah sebuah program manajemen produk tas rajut berbasis C++. Program memungkinkan pengguna untuk mengelola data produk dengan berbagai fitur, seperti *Create*, pengguna dapat menambahkan produk baru ke dalam daftar dengan memasukkan nama, harga, warna, dan stok produk. Kemudian *Read*, pengguna dapat melihat daftar produk dalam bentuk tabel dengan informasi lengkap. Selanjutnya *Update*, pengguna dapat mengedit harga dan stok produk. Dan *Delete*, pengguna dapat menghapus produk dari daftar berdasarkan nomor yang dipilih. Pada program ini pengguna diberikan kesempatan tiga kali untuk login dengan memasukkan nama (Alya) dan nim (054). Jika berhasil pengguna dapat mengakses menu CRUD yang tersedia. Program akan terus berjalan hingga pengguna memilih opsi keluar. Jika pengguna gagal login sebanyak tiga kali, maka program akan otomatis berhenti.

Dalam program ini digunakan struct untuk merepresentasikan data produk. Struct adalah tipe data bentukan yang memungkinkan pengelompokan beberapa data berbeda dalam satu unit. Struct yang digunakan di program ini meliputi *DetailProduk*, yang menyimpan informasi detail seperti warna dan stok produk, serta *Produk*, yang menyimpan informasi utama seperti nama dan harga produk, sekaligus mencakup struct *DetailProduk* di dalamnya sebagai nested struct. Penggunaan nested struct ini membantu menyusun data dengan lebih terstruktur dan memudahkan pengelolaan informasi produk secara hierarkis.

### 3. Source Code

#### A. Struct dan Nested Struct

Struct *DetailProduk* digunakan untuk menyimpan detail produk, seperti warna dan jumlah stok. Sedangkan Struct *Produk* merepresentasikan data utama produk, seperti nama dan harga. Struct ini juga memiliki anggota berupa *DetailProduk* sebagai nested struct untuk menyimpan informasi tambahan. Dalam penggunaan nested struct, data diakses melalui notasi titik (dot notation). Misalnya, `produk[0].detail.warna` mengakses warna produk pertama, dan `produk[0].detail.stok` mengakses stok produk pertama. Hal ini membuat data terorganisir secara hierarkis.

#### Source Code:

```
struct DetailProduk {
    string warna;
    int stok;
};

struct Produk {
    string nama;
    int harga;
    DetailProduk detail;
};

int main() {
    const int max_produk = 100;
    Produk produk[max_produk];
    int jumlahproduk = 5;

    produk[0] = {"Backpack Rajut", 300000, {"Pink", 20}};
    produk[1] = {"Tote Bag Rajut", 125000, {"Baby Blue", 45}};
    produk[2] = {"Sling Bag Rajut", 95000, {"Hitam", 30}};
    produk[3] = {"Hand Bag Rajut", 80000, {"Cream", 15}};
    produk[4] = {"Pouch Rajut", 65000, {"Maroon", 50}};

    cout << "Nama Produk: " << produk[0].nama << endl;
    cout << "Harga: " << produk[0].harga << endl;
    cout << "Warna: " << produk[0].detail.warna << endl;
    cout << "Stok: " << produk[0].detail.stok << endl;

    return 0;
}
```

## B. Fitur Tambah Produk

Fitur ini memungkinkan pengguna untuk menambahkan produk baru ke dalam daftar jika kapasitas belum penuh.

### Source Code:

```
if (jumlahproduk < max_produk) {
    cin.ignore();
    cout << "Masukkan nama produk: ";
    getline(cin, produk[jumlahproduk].nama);
    cout << "Masukkan harga produk: ";
    cin >> produk[jumlahproduk].harga;
    cin.ignore();
    cout << "Masukkan warna produk: ";
    getline(cin, produk[jumlahproduk].detail.warna);
    cout << "Masukkan stok produk: ";
    cin >> produk[jumlahproduk].detail.stok;
    jumlahproduk++;
    cout << "Produk berhasil ditambahkan!\n";
} else {
    cout << "Data penuh! Tidak bisa menambah produk
lagi.\n";
}
```

## C. Fitur Tampilkan Produk

Menampilkan daftar produk dalam bentuk tabel yang rapi.

### Source Code:

```
if (jumlahproduk == 0) {
    cout << "Belum ada produk yang tersedia.\n";
} else {
    cout << "\n+---+-----+-----+-----+-----+-----+\n";
    cout << "| No | Nama Produk          | Harga      | Stok | Warna          |\n";
    cout << "+---+-----+-----+-----+-----+-----+\n";
    for (int i = 0; i < jumlahproduk; i++) {
        cout << "| " << setw(2) << i + 1
            << " | " << left << setw(18) << produk[i].nama
            << " | " << right << setw(10) << produk[i].harga
            << " | " << setw(4) << produk[i].detail.stok
            << " | " << setw(10) << left << produk[i].detail.warna << " |\n";
    }
    cout << "+---+-----+-----+-----+-----+-----+\n";
}
```

## D. Fitur Ubah Produk

Fitur ini memungkinkan pengguna untuk mengedit harga dan stok produk berdasarkan nomor urutnya.

### Source Code:

```
if (jumlahproduk == 0) {
    cout << "Tidak ada produk untuk diubah.\n";
} else {
    int index;
    cout << "Masukkan nomor produk yang ingin diubah: ";
    cin >> index;
    if (index > 0 && index <= jumlahproduk) {
        index--;
        cout << "Masukkan harga baru: ";
        cin >> produk[index].harga;
        cout << "Masukkan stok baru: ";
        cin >> produk[index].detail.stok;
        cout << "Produk berhasil diperbarui!\n";
    } else {
        cout << "Nomor produk tidak valid!\n";
    }
}
```

## E. Fitur Hapus Produk

Fitur ini digunakan untuk menghapus produk dari array berdasarkan nomor produk yang dipilih.

### Source Code:

```
if (jumlahproduk == 0) {
    cout << "Tidak ada produk yang bisa dihapus.\n";
} else {
    int index;
    cout << "Masukkan nomor produk yang ingin dihapus: ";
    cin >> index;
    if (index > 0 && index <= jumlahproduk) {
        index--;
        for (int i = index; i < jumlahproduk - 1; i++) {
            produk[i] = produk[i + 1];
        }
        jumlahproduk--;
        cout << "Produk berhasil dihapus!\n";
    } else {
        cout << "Nomor produk tidak valid!\n";
    }
}
```

#### 4. Hasil Output

```
Masukkan Nama: Alya
Masukkan NIM: 041
Login gagal! Sisa percobaan: 2
Masukkan Nama: Milly
Masukkan NIM: 054
Login gagal! Sisa percobaan: 1
Masukkan Nama: Ssulki
Masukkan NIM: 013
Login gagal! Sisa percobaan: 0
Mohon maaf percobaan login gagal! Program berhenti.
PS C:\Users\alyam\coding>
```

Gambar 4.1 Output Ketika Pengguna Gagal Login

```
=== Manajemen Produk Tas Rajut ===
1. Tambah Produk
2. Tampilkan Produk
3. Ubah Produk
4. Hapus Produk
5. Keluar
Pilihan: 2
```

No	Nama Produk	Harga	Stok	Warna
1	Backpack Rajut	300000	20	Pink
2	Tote Bag Rajut	125000	45	Baby Blue
3	Sling Bag Rajut	95000	30	Hitam
4	Hand Bag Rajut	80000	15	Cream
5	Pouch Rajut	65000	50	Maroon

Gambar 4.2 Output Ketika Pengguna Memilih Pilihan 2

```

=== Manajemen Produk Tas Rajut ===
1. Tambah Produk
2. Tampilkan Produk
3. Ubah Produk
4. Hapus Produk
5. Keluar
Pilihan: 1
Masukkan nama produk: Goodie Bag Rajut
Masukkan harga produk: 175000
Masukkan warna produk: Army
Masukkan stok produk: 7
Produk berhasil ditambahkan!

```

No	Nama Produk	Harga	Stok	Warna
1	Backpack Rajut	300000	20	Pink
2	Tote Bag Rajut	125000	45	Baby Blue
3	Sling Bag Rajut	95000	30	Hitam
4	Hand Bag Rajut	80000	15	Cream
5	Pouch Rajut	65000	50	Maroon
6	Goodie Bag Rajut	175000	7	Army

Gambar 4.3 Output Ketika Pengguna Memilih Pilihan 1

```

=== Manajemen Produk Tas Rajut ===
1. Tambah Produk
2. Tampilkan Produk
3. Ubah Produk
4. Hapus Produk
5. Keluar
Pilihan: 3

+---+---+---+---+---+---+
| No | Nama Produk | Harga | Stok | Warna |
+---+---+---+---+---+---+
| 1 | Backpack Rajut | 300000 | 20 | Pink |
| 2 | Tote Bag Rajut | 125000 | 45 | Baby Blue |
| 3 | Sling Bag Rajut | 95000 | 30 | Hitam |
| 4 | Hand Bag Rajut | 80000 | 15 | Cream |
| 5 | Pouch Rajut | 65000 | 50 | Maroon |
| 6 | Goodie Bag Rajut | 175000 | 7 | Army |
+---+---+---+---+---+---+

Masukkan nomor produk yang ingin diubah: 5
Masukkan harga baru: 75000
Masukkan stok baru: 30
Produk berhasil diperbarui!

```

Gambar 4.4 Output Ketika Pengguna Memilih Pilihan 3



```

=== Manajemen Produk Tas Rajut ===
1. Tambah Produk
2. Tampilkan Produk
3. Ubah Produk
4. Hapus Produk
5. Keluar
Pilihan: 4

+---+-----+-----+-----+-----+
| No | Nama Produk      | Harga   | Stok | Warna   |
+---+-----+-----+-----+-----+
| 1  | Backpack Rajut   | 300000  | 20   | Pink    |
| 2  | Tote Bag Rajut   | 125000  | 45   | Baby Blue |
| 3  | Sling Bag Rajut  | 95000   | 30   | Hitam   |
| 4  | Hand Bag Rajut   | 80000   | 15   | Cream   |
| 5  | Pouch Rajut      | 75000   | 30   | Maroon  |
| 6  | Goodie Bag Rajut | 175000  | 7    | Army    |
+---+-----+-----+-----+-----+

Masukkan nomor produk yang ingin dihapus: 3
Produk berhasil dihapus!

```

*Gambar 4.5 Output Ketika Pengguna Memilih Pilihan 4*

```

=== Manajemen Produk Tas Rajut ===
1. Tambah Produk
2. Tampilkan Produk
3. Ubah Produk
4. Hapus Produk
5. Keluar
Pilihan: 5
Terima kasih telah menggunakan program ini. Have a great day >_<!
PS C:\Users\alyam\coding>

```

*Gambar 4.6 Output Ketika Pengguna Memilih Pilihan 5*

## 5. Langkah-Langkah Git

### 5.1 Git Add

Git Add berfungsi untuk menambahkan file tertentu ke staging area agar siap untuk dikommit.

```
HP-GK@DESKTOP-N8AD402 MINGW64 ~/desktop/praktikum-apl/post-test/post-test-3 (main)
$ git add .
```

*Gambar 5.1 Git Add*

### 5.2 Git Commit

Git Commit berfungsi untuk menyimpan perubahan yang ada di staging area ke dalam repository disertai dengan pesan deskriptif.

```
HP-GK@DESKTOP-N8AD402 MINGW64 ~/desktop/praktikum-apl/post-test/post-test-3 (main)
$ git commit -m "file cpp dan exe"
[main 44e261d] file cpp dan exe
2 files changed, 187 insertions(+)
create mode 100644 post-test/post-test-3/2409106054-AlyaMayasha-PT-3.cpp
create mode 100644 post-test/post-test-3/2409106054-AlyaMayasha-PT-3.exe
```

*Gambar 5.2 Git Commit*

### 5.3 Git Push Origin Main

Git Push Origin Main berfungsi untuk mengirim (mengunggah) perubahan dari branch main di repository lokal ke repository remote yang telah dikonfigurasi.

```
HP-GK@DESKTOP-N8AD402 MINGW64 ~/desktop/praktikum-apl/post-test/post-test-3 (main)
$ git push origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 679.47 KiB | 1.47 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/AlyaMayasha/praktikum-apl.git
   fe94ce5..44e261d  main -> main
```

*Gambar 5.3 Git Push Origin Main*