

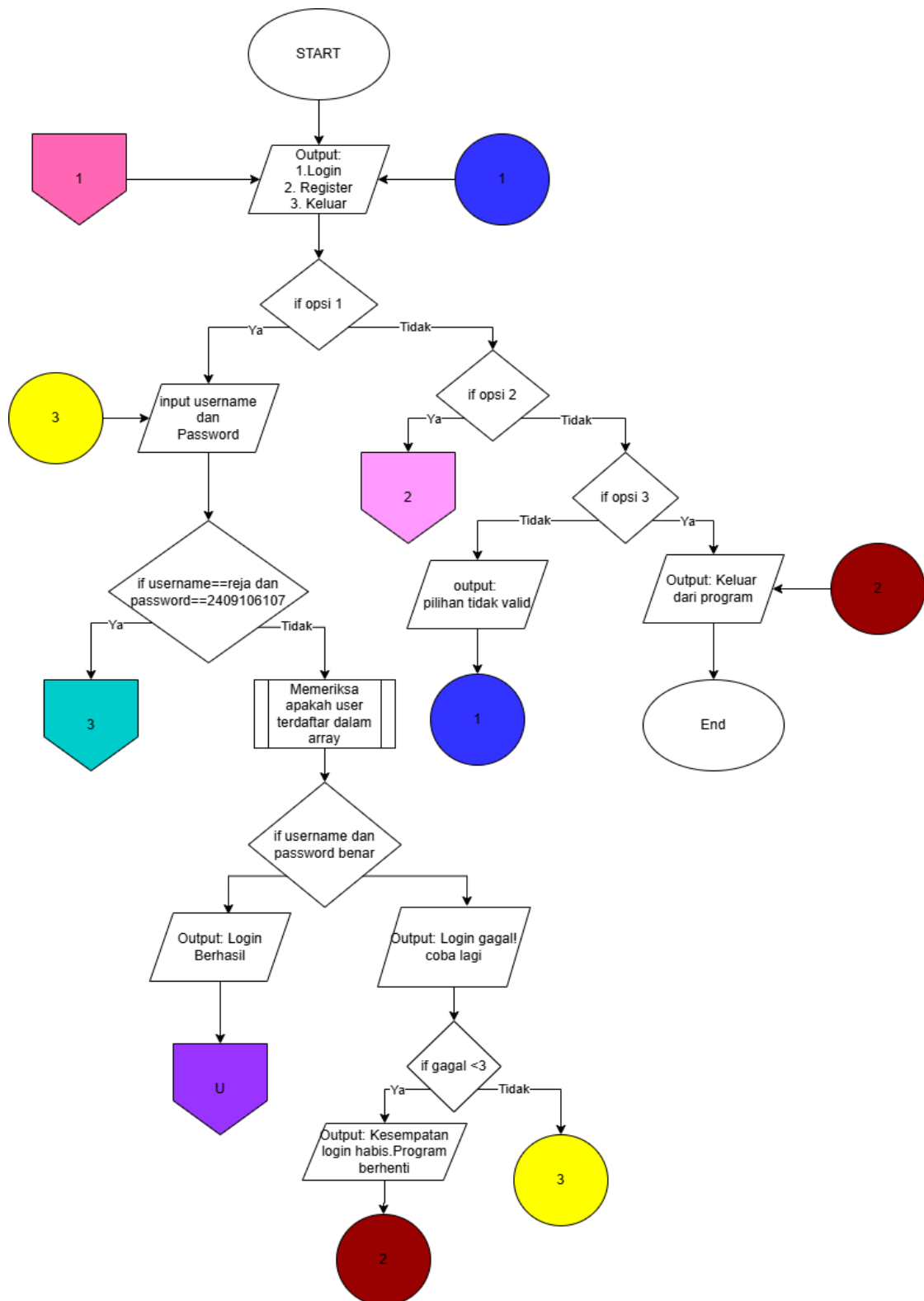
LAPORAN PRAKTIKUM
POSTTEST 4
ALGORITMA PEMROGRAMAN LANJUT

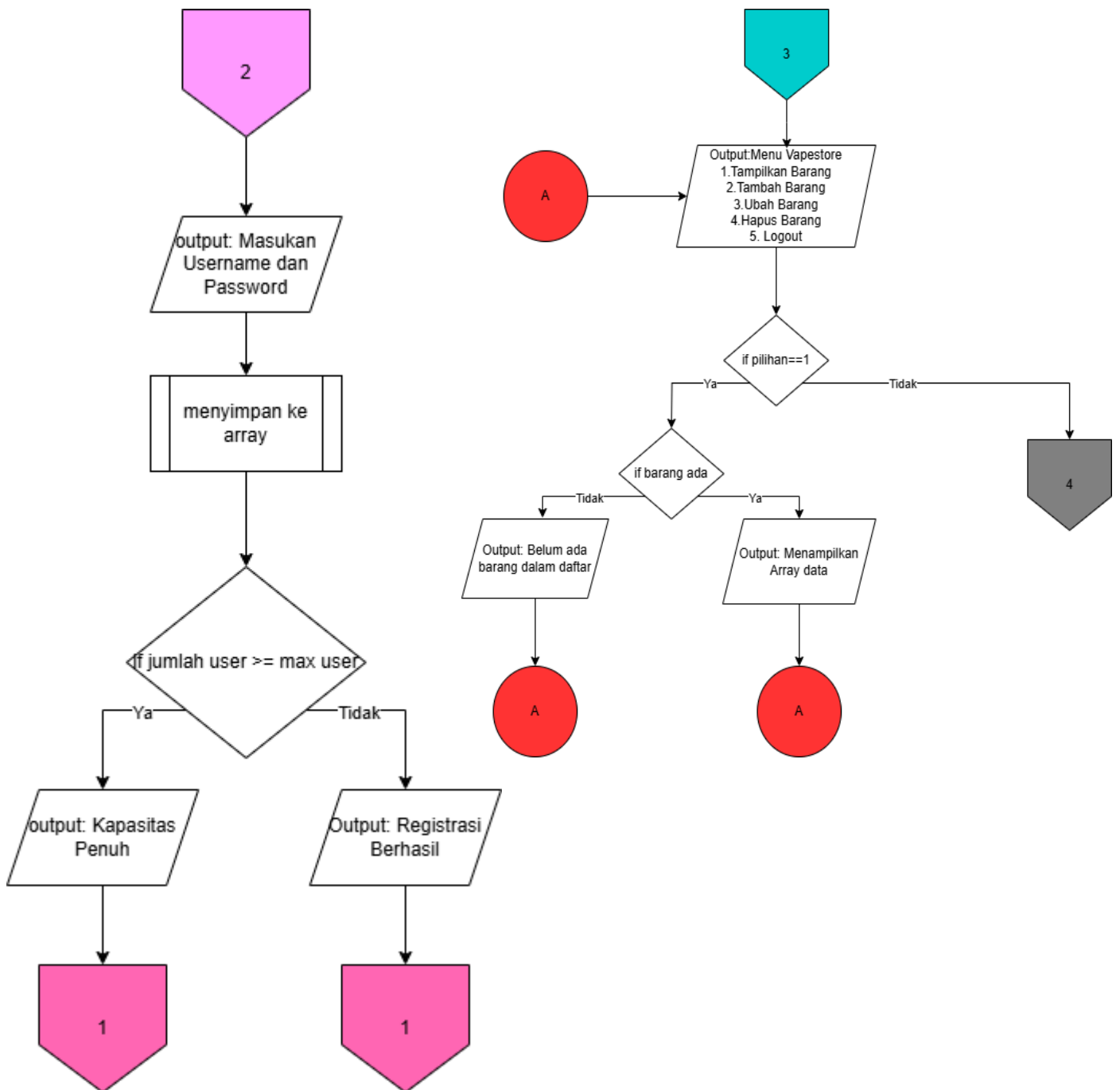


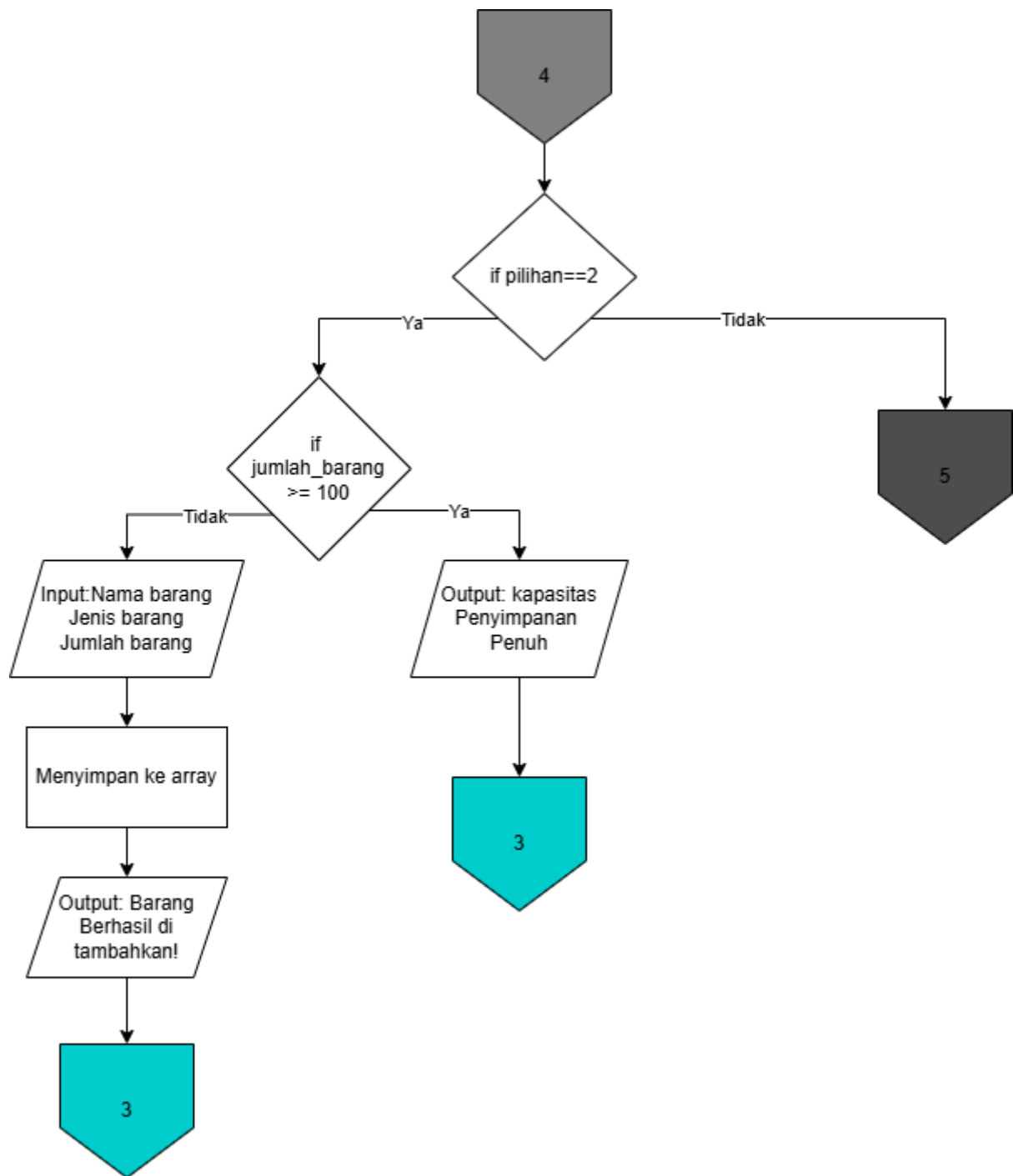
Disusun oleh:
Reza Alameka (2409106107)
Kelas (C2 '24)

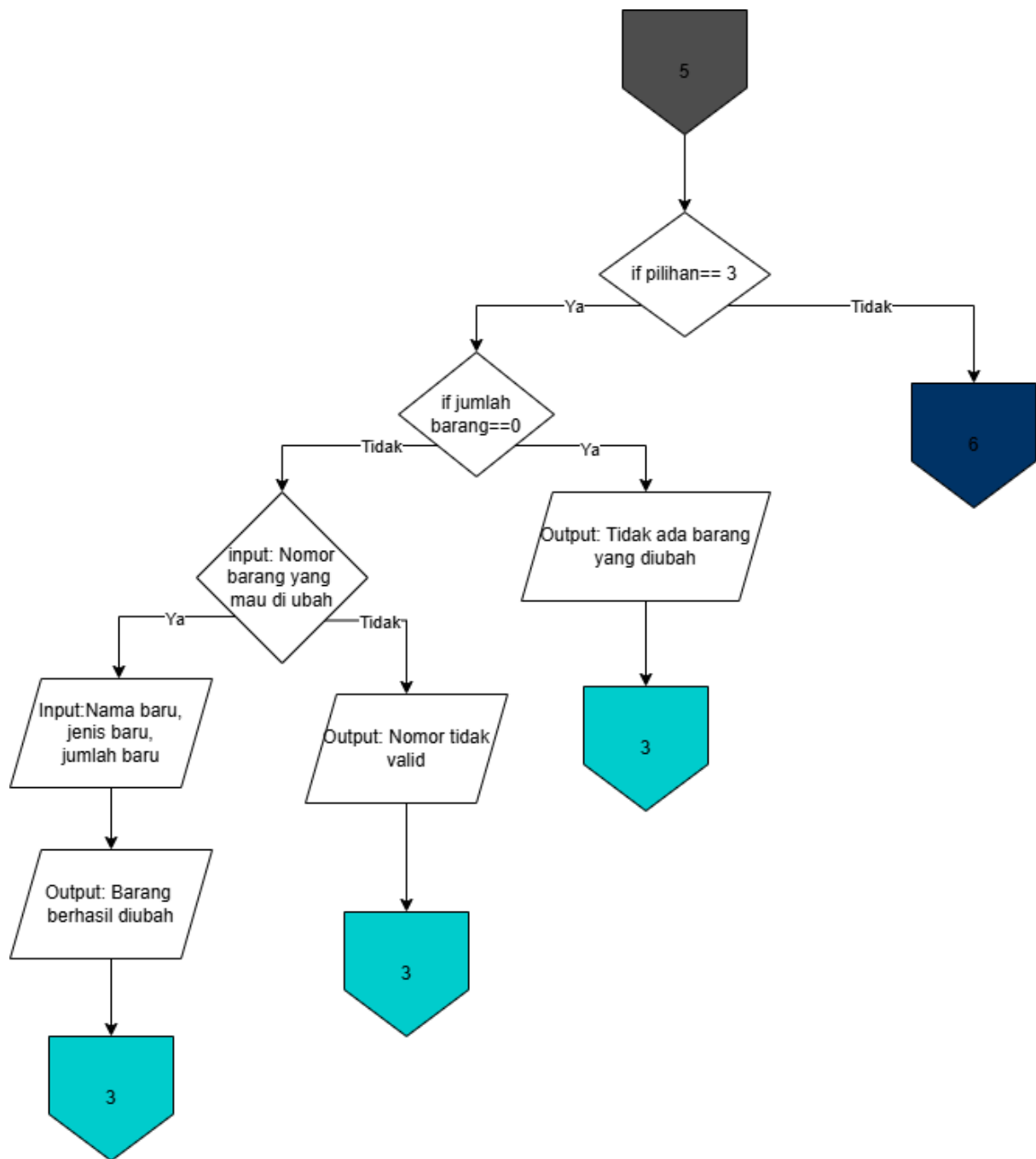
PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

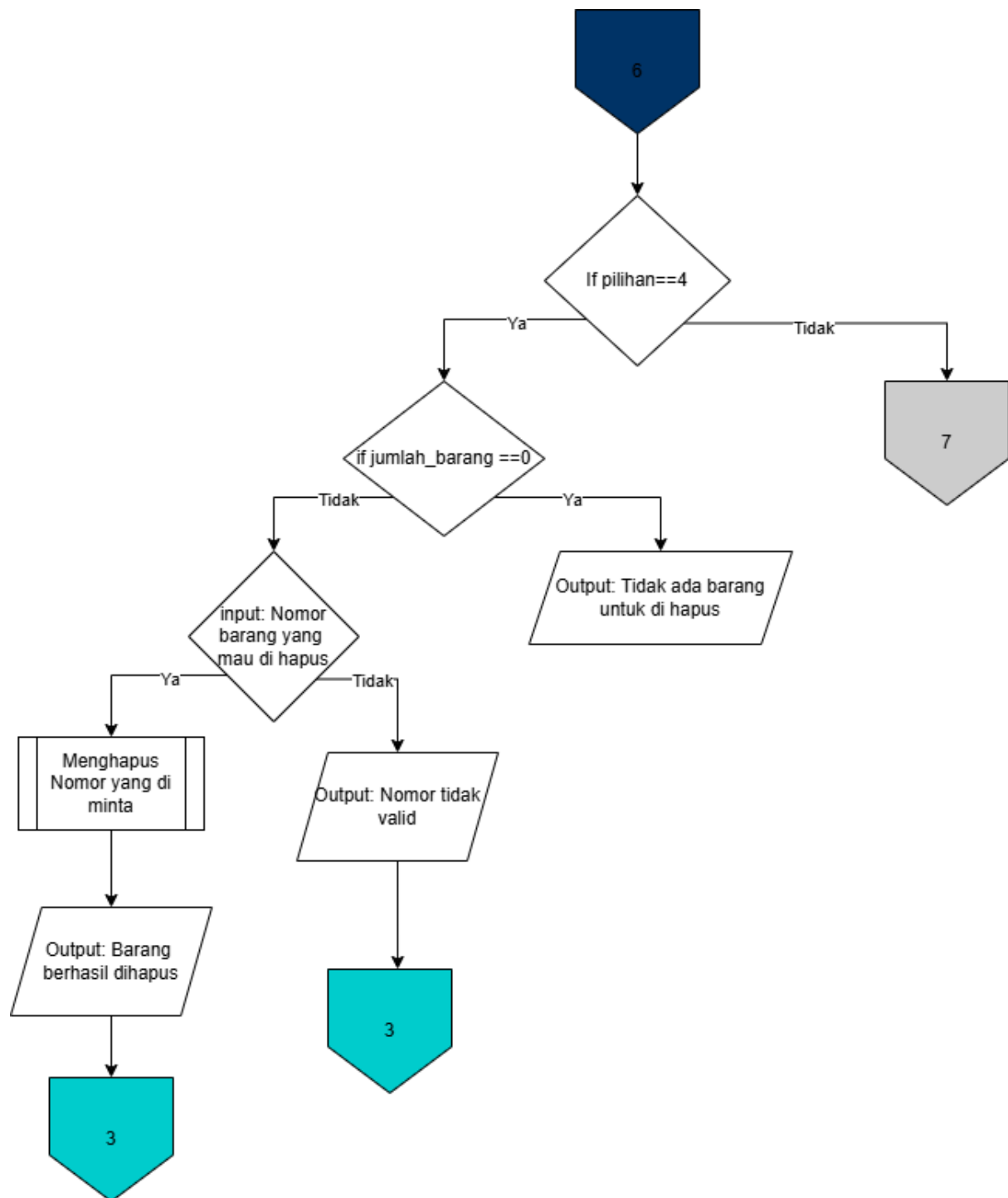
1. Flowchart

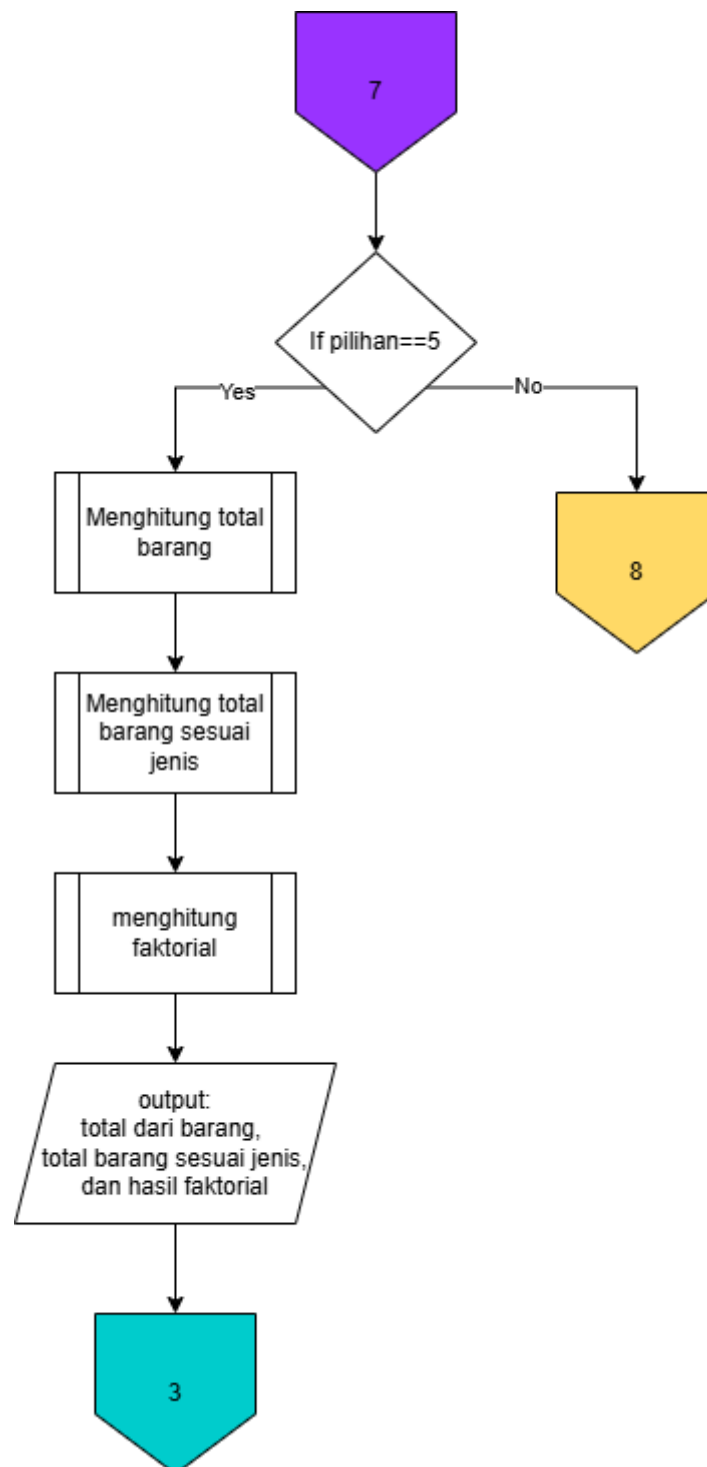


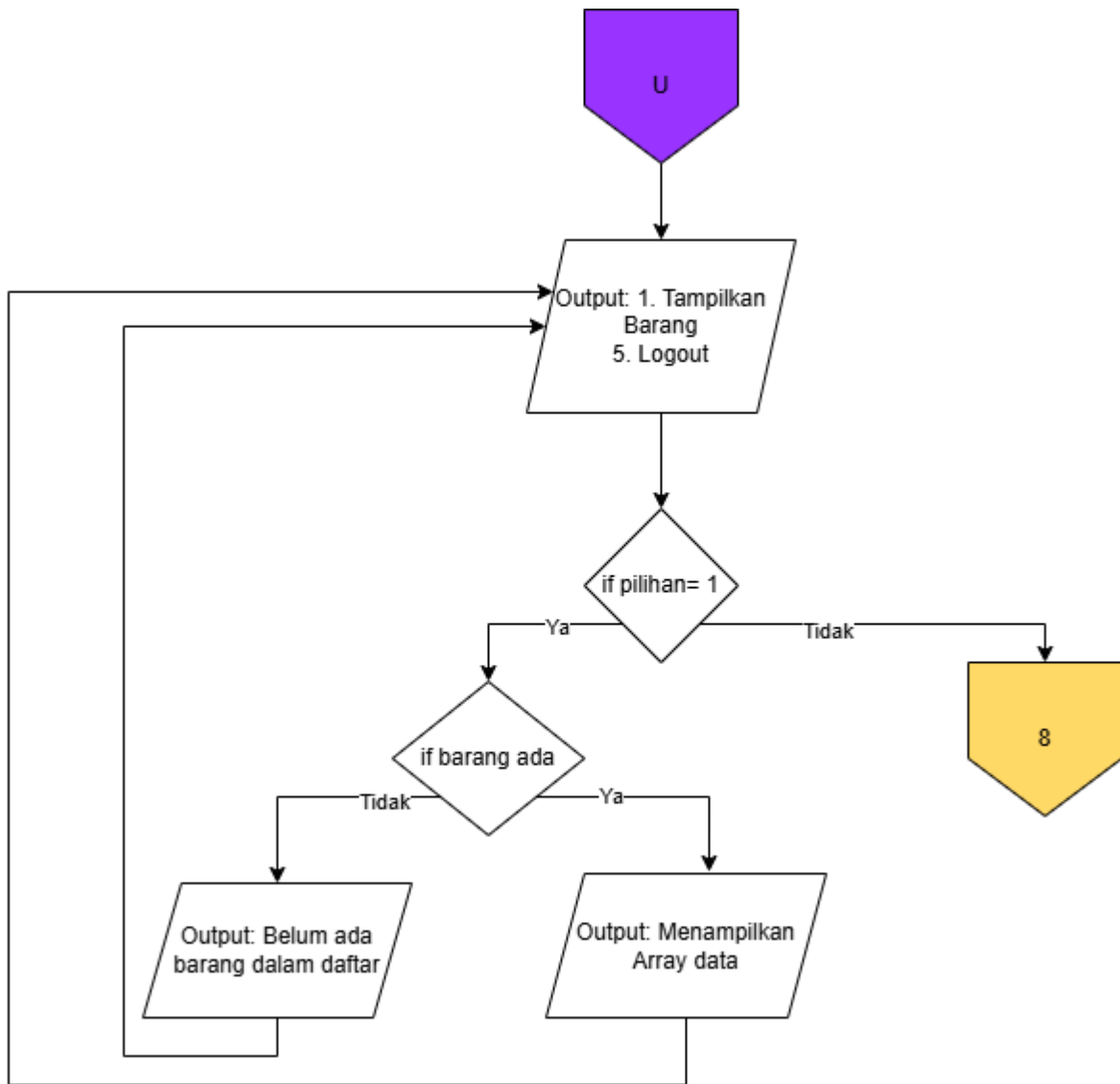


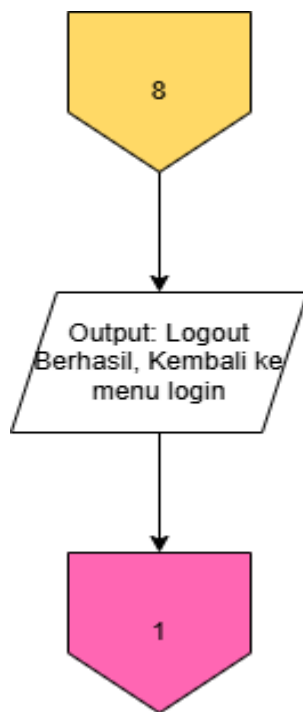












2. Analisis Program

2.2 Tujuan dan Fungsi Program

1. Login & Registrasi

A. Tujuan:

- Memastikan hanya pengguna terdaftar yang dapat mengakses sistem.
- Memberikan akses khusus kepada admin untuk mengelola data barang.

B. Fungsi:

- **Registrasi:** Pengguna dapat membuat akun baru dengan username dan password.
- **Login:** Pengguna dan admin masuk ke sistem dengan kredensial yang valid.
- **Admin Login:** Akses khusus untuk admin (username: raja, password: 2409106107).

2. Fitur Admin

A. Tujuan:

- Memudahkan admin dalam mengelola stok barang di toko vape.

B. Fungsi:

- **Menampilkan Barang:** Melihat daftar barang beserta jenis dan stok dalam format tabel.
- **Menambah Barang:** Memasukkan produk baru ke dalam sistem (nama, jenis, jumlah).
- **Mengubah Barang:** Memperbarui informasi produk (nama, jenis, atau stok).
- **Menghapus Barang:** Menghilangkan produk yang tidak lagi tersedia.
- **Info Total Barang:** Menampilkan total stok barang dan demo fungsi rekursif (faktorial).

3. Fitur User

A. Tujuan:

- Memberikan akses terbatas kepada pengguna biasa untuk melihat stok barang.

B. Fungsi:

- **Menampilkan Barang:** Melihat daftar barang yang tersedia tanpa bisa mengubah data.

2.3 Manfaat Utama

1. Mempermudah Manajemen Stok

A. Admin dapat:

- Menambah, mengedit, atau menghapus data barang secara efisien.
- Melihat ringkasan stok (total barang, kategori Device/Liquid).

B. User dapat:

- Mengecek ketersediaan produk tanpa risiko mengubah data.

2. Meningkatkan Keamanan dan Aksesibilitas

A. Sistem Login:

- Mencegah akses tidak sah.
- Admin memiliki kontrol penuh, sedangkan user hanya bisa melihat.

B. Antarmuka Berbasis Menu:

- Navigasi mudah dengan pilihan numerik (1, 2, 3, dst.).

3. Mengoptimalkan Pengelolaan Toko Vape

A. Daftar Barang Terstruktur:

- Tabel rapi dengan kolom **Nama**, **Jenis**, dan **Jumlah**.

B. Hindari Kesalahan Manual:

- Sistem menghitung total stok otomatis (termasuk per kategori).

4. Mempermudah Interaksi Pengguna

A. Menu Sederhana:

- User hanya perlu memilih angka untuk operasi tertentu.

B. Fitur Tambahan:

- Demo rekursif (faktorial) sebagai contoh penerapan konsep pemrograman.

3. Source Code

```
#include <iostream>
#include <iomanip>
#include <string>
using namespace std;

int MAX_BARANG = 100;
int MAX_USER = 100;

struct Barang {
    string nama;
    string jenis;
    int jumlah;
};

struct User {
    string username;
    string password;
};

struct VapeStore {
    Barang barang[100];
    User users[100];
    int jumlahBarang = 6;
    int jumlahUser = 0;
};

void tampilkanMenuLogin();
void prosesLogin(VapeStore &store, bool &loginBerhasil, bool &isAdmin);
void prosesRegister(VapeStore &store);
void tampilkanMenuUser();
void tampilkanMenuAdmin();
void tampilkanBarang(VapeStore &store);
void tambahBarang(VapeStore &store);
void ubahBarang(VapeStore &store);
void hapusBarang(VapeStore &store);
void tampilkanHeaderTabel();
void tampilkanBarisBarang(Barang &barang, int index);
void tampilkanFooterTabel();
int hitungTotalBarang(VapeStore &store);
int hitungTotalBarang(VapeStore &store, string &jenis);
int faktorial(int n);
void tampilkanInfoTotalBarang(VapeStore &store);

// Fungsi utama
int main() {
    VapeStore store = {
```

```

        {"Oxva xlim G0", "Device", 10}, {"Voopoo Drag X", "Device", 7},
{"TRML T99", "Device", 5},
        {"Makna V2 3mg", "Liquid", 15}, {"Bolu Lapis Talas V1 6mg",
"Liquid", 12}, {"The Orama V1 3mg", "Liquid", 20}},
    {}},
    6,
    0
};

while (true) {
    tampilkanMenuLogin();
    int menuLogin;
    cout << "Pilihan: ";
    cin >> menuLogin;

    switch (menuLogin) {
        case 1: {
            bool loginBerhasil = false;
            bool isAdmin = false;
            prosesLogin(store, loginBerhasil, isAdmin);

            if (loginBerhasil) {
                if (isAdmin) {
                    // Menu Admin
                    while (true) {
                        tampilkanMenuAdmin();
                        int pilihan;
                        cout << "Pilihan: ";
                        cin >> pilihan;

                        switch (pilihan) {
                            case 1: tampilkanBarang(store); break;
                            case 2: tambahBarang(store); break;
                            case 3: ubahBarang(store); break;
                            case 4: hapusBarang(store); break;
                            case 5: tampilkanInfoTotalBarang(store);
break;

                            case 6:
                                cout << "Logout berhasil.\n";
                                goto logout;
                            default:
                                cout << "Pilihan tidak valid!\n";
                                break;
                        }
                    }
                }
            } else {
                // Menu User Biasa
                while (true) {
                    tampilkanMenuUser();

```

```

        int pilihan;
        cout << "Pilihan: ";
        cin >> pilihan;

        switch (pilihan) {
            case 1: tampilkanBarang(store); break;
            case 2:
                cout << "Logout berhasil.\n";
                goto logout;
            default:
                cout << "Pilihan tidak valid!\n";
        }
    }
}
logout;;
}
break;
}
case 2:
    prosesRegister(store);
    break;
case 3:
    cout << "Keluar dari program.\n";
    return 0;
default:
    cout << "Pilihan tidak valid!\n";
}
}

return 0;
}

void tampilkanMenuLogin() {
    cout << "\n=== Menu Login ===\n";
    cout << "1. Login\n";
    cout << "2. Register\n";
    cout << "3. Keluar\n";
}

void prosesLogin(VapeStore &store, bool &loginBerhasil, bool &isAdmin) {
    string username, password;
    int attempts = 0;
    loginBerhasil = false;
    isAdmin = false;

    while (attempts < 3 && !loginBerhasil) {
        cout << "Username: ";
    }
}

```

```

    cin >> username;
    cout << "Password: ";
    cin >> password;

    // Cek admin
    if (username == "reja" && password == "2409106107") {
        loginBerhasil = true;
        isAdmin = true;
        cout << "Login sebagai admin berhasil!\n";
        return;
    }

    // Cek user biasa
    for (int i = 0; i < store.jumlahUser; i++) {
        if (store.users[i].username == username &&
store.users[i].password == password) {
            loginBerhasil = true;
            cout << "Login berhasil!\n";
            return;
        }
    }

    cout << "Login gagal! Coba lagi.\n";
    attempts++;
}

if (attempts == 3) {
    cout << "Kesempatan login habis. Program berhenti.\n";
    exit(0);
}
}

void prosesRegister(VapeStore &store) {
    if (store.jumlahUser >= MAX_USER) {
        cout << "Kapasitas user penuh!\n";
        return;
    }

    cout << "Username: ";
    cin >> store.users[store.jumlahUser].username;
    cout << "Password: ";
    cin >> store.users[store.jumlahUser].password;
    store.jumlahUser++;
    cout << "Registrasi berhasil!\n";
}

void tampilkanMenuUser() {
    cout << "\n=== Menu User ===\n";
}

```

```

        cout << "1. Tampilkan Barang\n";
        cout << "2. Logout\n";
    }

    void tampilkanMenuAdmin() {
        cout << "\n=== Menu Admin ===\n";
        cout << "1. Tampilkan Barang\n";
        cout << "2. Tambah Barang\n";
        cout << "3. Ubah Barang\n";
        cout << "4. Hapus Barang\n";
        cout << "5. Info Total Barang\n";
        cout << "6. Logout\n";
    }

    void tampilkanBarang(VapeStore &store) {
        if (store.jumlahBarang == 0) {
            cout << "Belum ada barang dalam daftar.\n";
            return;
        }

        tampilkanHeaderTabel();
        for (int i = 0; i < store.jumlahBarang; i++) {
            tampilkanBarisBarang(store.barang[i], i + 1);
        }
        tampilkanFooterTabel();
    }

    void tambahBarang(VapeStore &store) {
        if (store.jumlahBarang >= MAX_BARANG) {
            cout << "Kapasitas penyimpanan penuh!\n";
            return;
        }

        cout << "Nama barang: ";
        cin.ignore();
        getline(cin, store.barang[store.jumlahBarang].nama);
        cout << "Jenis (Device/Liquid): ";
        getline(cin, store.barang[store.jumlahBarang].jenis);
        cout << "Jumlah: ";
        cin >> store.barang[store.jumlahBarang].jumlah;
        store.jumlahBarang++;
        cout << "Barang berhasil ditambahkan!\n";
    }

    void ubahBarang(VapeStore &store) {
        if (store.jumlahBarang == 0) {
            cout << "Tidak ada barang untuk diubah.\n";
            return;
        }
    }

```



```

    }

    tampilkanBarang(store);
    int index;
    cout << "Pilih nomor barang yang ingin diubah: ";
    cin >> index;

    if (index < 1 || index > store.jumlahBarang) {
        cout << "Nomor tidak valid!\n";
        return;
    }

    cin.ignore();
    cout << "Nama baru: ";
    getline(cin, store.barang[index - 1].nama);
    cout << "Jenis baru (Device/Liquid): ";
    getline(cin, store.barang[index - 1].jenis);
    cout << "Jumlah baru: ";
    cin >> store.barang[index - 1].jumlah;
    cout << "Barang berhasil diubah!\n";
}

void hapusBarang(VapeStore &store) {
    if (store.jumlahBarang == 0) {
        cout << "Tidak ada barang untuk dihapus.\n";
        return;
    }

    tampilkanBarang(store);
    int index;
    cout << "Pilih nomor barang yang akan dihapus: ";
    cin >> index;

    if (index < 1 || index > store.jumlahBarang) {
        cout << "Nomor tidak valid!\n";
        return;
    }

    for (int i = index - 1; i < store.jumlahBarang - 1; i++) {
        store.barang[i] = store.barang[i + 1];
    }
    store.jumlahBarang--;
    cout << "Barang berhasil dihapus!\n";
}

void tampilkanHeaderTabel() {
    cout << "+-----+-----+-----+-----+\n";
    cout << "| No   | Nama Barang           | Jenis       | Jumlah | \n";

```

```

        cout << "+-----+-----+-----+-----+\n";
    }

    void tampilkanBarisBarang(Barang &barang, int index) {
        cout << "| " << setw(3) << index << " | " << setw(20) << barang.nama <<
        " | "
            << setw(10) << barang.jenis << " | "
            << setw(6) << barang.jumlah << " |\n";
    }

    void tampilkanFooterTabel() {
        cout << "+-----+-----+-----+-----+\n";
    }

    int hitungTotalBarang(VapeStore &store) {
        int total = 0;
        for (int i = 0; i < store.jumlahBarang; i++) {
            total += store.barang[i].jumlah;
        }
        return total;
    }

    int hitungTotalBarang(VapeStore &store, string &jenis) {
        int total = 0;
        for (int i = 0; i < store.jumlahBarang; i++) {
            if (store.barang[i].jenis == jenis) {
                total += store.barang[i].jumlah;
            }
        }
        return total;
    }

    int faktorial(int n) {
        if (n <= 1) return 1;
        return n * faktorial(n - 1);
    }

    void tampilkanInfoTotalBarang(VapeStore &store) {
        cout << "\n=== Informasi Total Barang ===\n";
        cout << "Total semua barang: " << hitungTotalBarang(store) << endl;

        string jenis1 = "Device";
        string jenis2 = "Liquid";
        cout << "Total Device: " << hitungTotalBarang(store, jenis1) << endl;
        cout << "Total Liquid: " << hitungTotalBarang(store, jenis2) << endl;
    }

```

```
cout << "\nDemo Fungsi Rekursif (Faktorial):\n";  
for (int i = 1; i <= 5; i++) {  
    cout << "Faktorial " << i << ": " << faktorial(i) << endl;  
}  
}
```

4. Uji Coba dan Hasil Output

```
Menu Login:  
1. Login  
2. Register  
3. Keluar  
Pilihan: 2  
Masukkan Username: Alameka  
Masukkan Password: 1  
Registrasi berhasil!
```

Gambar 4.1 Output Register

```
Menu Vape Store:  
1. Tampilkan Barang  
5. Logout  
Pilihan: 5  
Logout berhasil. Kembali ke menu login.
```

Gambar 4.2 Menu User

```

Menu Login:
1. Login
2. Register
3. Keluar
Pilihan: 1
Masukkan Username: reja
Masukkan Password: 2409106107

Menu Vape Store:
1. Tampilkan Barang
2. Tambah Barang
3. Ubah Barang
4. Hapus Barang
5. Logout
Pilihan: 

```

Gambar 4.3 Login dan Menu Admin

Pilihan: 1

No	Nama Barang	Jenis	Jumlah
1	Oxva xlim GO	Device	10
2	Voopoo Drag X	Device	7
3	TRML T99	Device	5
4	Makna V2 3mg	Liquid	15
5	Bolu Lapis Talas V1 6mg	Liquid	12
6	The Orama V1 3mg	Liquid	20

Gambar 4.4 Output Pilihan 1 Admin

```

Pilihan: 2
Masukkan nama barang: Makna V3 9 mg
Masukkan jenis (Device/Liquid): Liquid
Masukkan jumlah barang: 99
Barang berhasil ditambahkan!

```

Gambar 4.4 Output Pilihan 2

```

Pilihan: 3
Pilih nomor barang yang ingin diubah: 1
Masukkan nama baru: Oxva Sq Pro
Masukkan jenis baru (Device/Liquid): Device
Masukkan jumlah baru: 59
Barang berhasil diubah!

```

Gambar 4.5 Output Pilihan 3

```
Pilihan: 4
Masukkan nomor barang yang akan dihapus: 5
Barang berhasil dihapus!
```

Gambar 4.6 Output Pilihan 4

```
Pilihan: 6
Logout berhasil.
```

Gambar 4.7 Ouput Pilihan

```
Pilihan: 1
+-----+-----+-----+-----+
| No | Nama Barang | Jenis | Jumlah |
+-----+-----+-----+-----+
| 1 | Oxva xlim GO | Device | 10 |
| 2 | Vopoo Drag X | Device | 7 |
| 3 | TRML T99 | Device | 5 |
| 4 | Makna V2 3mg | Liquid | 15 |
| 5 | Bolu Lapis Talas V1 6mg | Liquid | 12 |
| 6 | The Orama V1 3mg | Liquid | 20 |
+-----+-----+-----+-----+
```

Gambar 4.8 Output Pilihan 1

```
Pilihan: 5

=== Informasi Total Barang ===
Total semua barang: 69
Total Device: 22
Total Liquid: 47

Demo Fungsi Rekursif (Faktorial):
Faktorial 1: 1
Faktorial 2: 2
Faktorial 3: 6
Faktorial 4: 24
Faktorial 5: 120
```

Gambar 4.9 Output Pilihan 5

5. Langkah-Langkah Git pada VSCode

```
MINGW64:/d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL
$ git config --global user.email "rezalameka19@gmail.com"

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL
$ git init
Initialized empty Git repository in D:/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL/.git/

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (master)
$ git add .

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (master)
$ git branch -M main

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (main)
$ git remote add origin https://github.com/rezalameka/Praktikum-APL.git

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (main)
$ git commit -m "rejaaja"
[main (root-commit) 3c0dc41] rejaaja
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Posttest/Posttest_1/2409106107-RezaAlameka-Pt-1.cpp

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 322 bytes | 322.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/rezalameka/Praktikum-APL.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

Acer@LAPTOP-HD3ENN1L MINGW64 /d/KULIAH/SEM 2/PRAKTIKUM APL/Praktikum-APL (main)
$ |
```

1. Inisialisasi Git (git init): Membuat repository Git lokal.
2. Konfigurasi Git (git config): Menetapkan email pengguna.
3. Menambahkan file (git add .): Menambahkan semua file ke staging area.
4. Membuat branch utama (git branch -M main): Mengubah nama branch ke main.
5. Menambahkan remote repository (git remote add origin <URL>): Menghubungkan repository lokal ke GitHub.
6. Commit perubahan (git commit -m "rejaaja"): Menyimpan perubahan ke repository lokal.
7. Push ke GitHub (git push -u origin main): Mengunggah perubahan ke repository di GitHub