

**LAPORAN PRAKTIKUM**  
**POSTTEST 6**  
**ALGORITMA PEMROGRAMAN LANJUT**



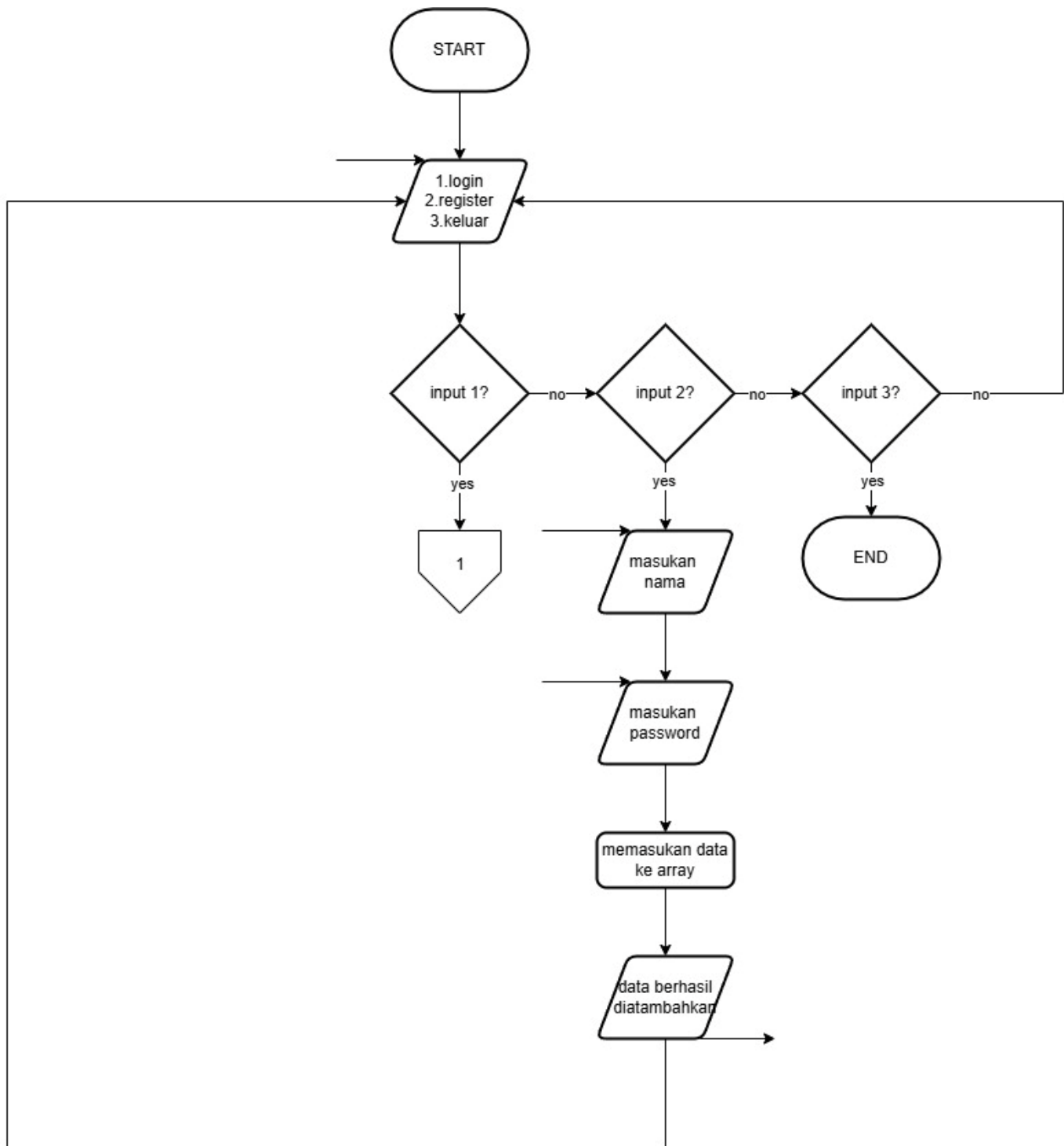
**Disusun oleh:**

**Nama (2409106056)**

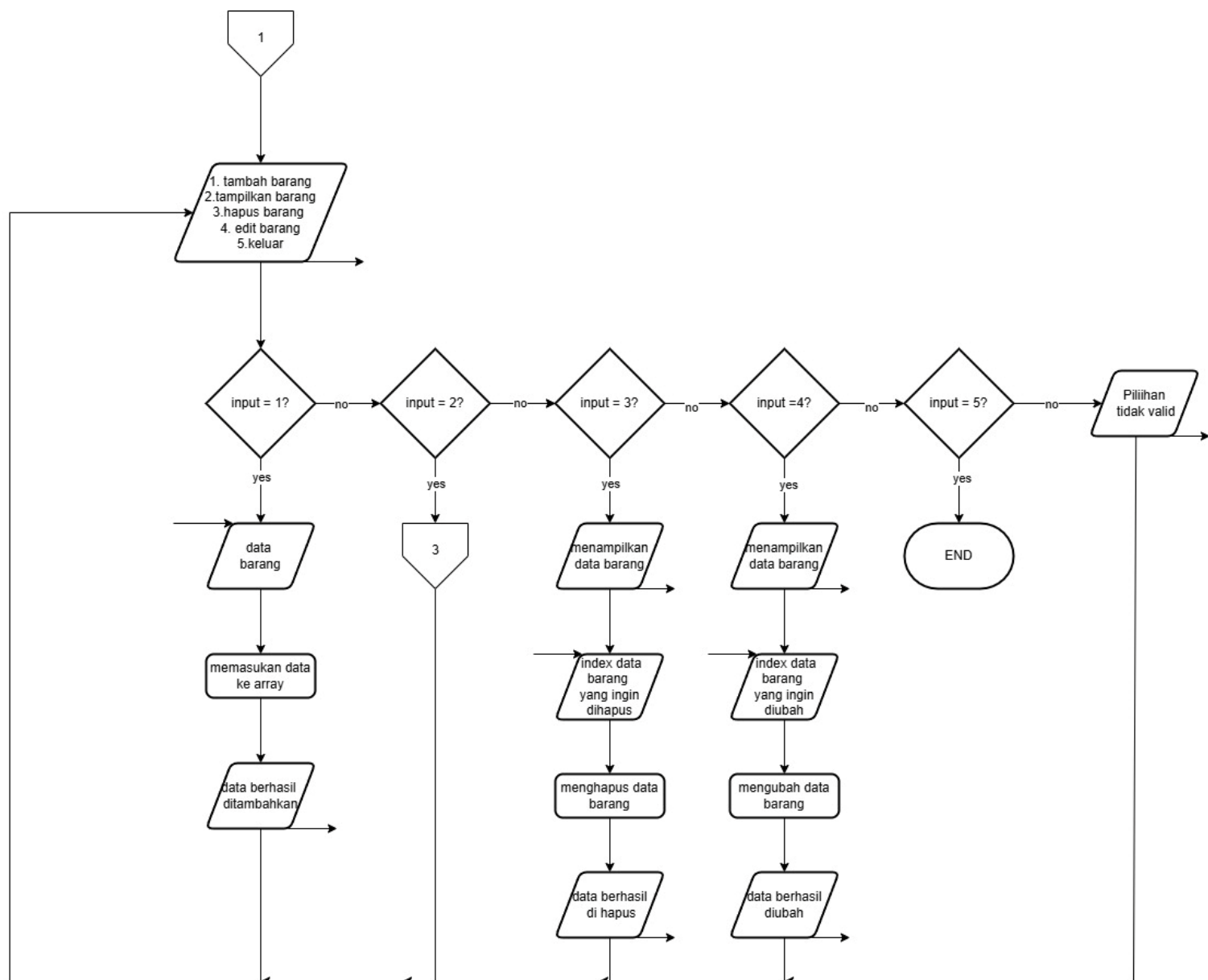
**Kelas (B1 '24)**

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

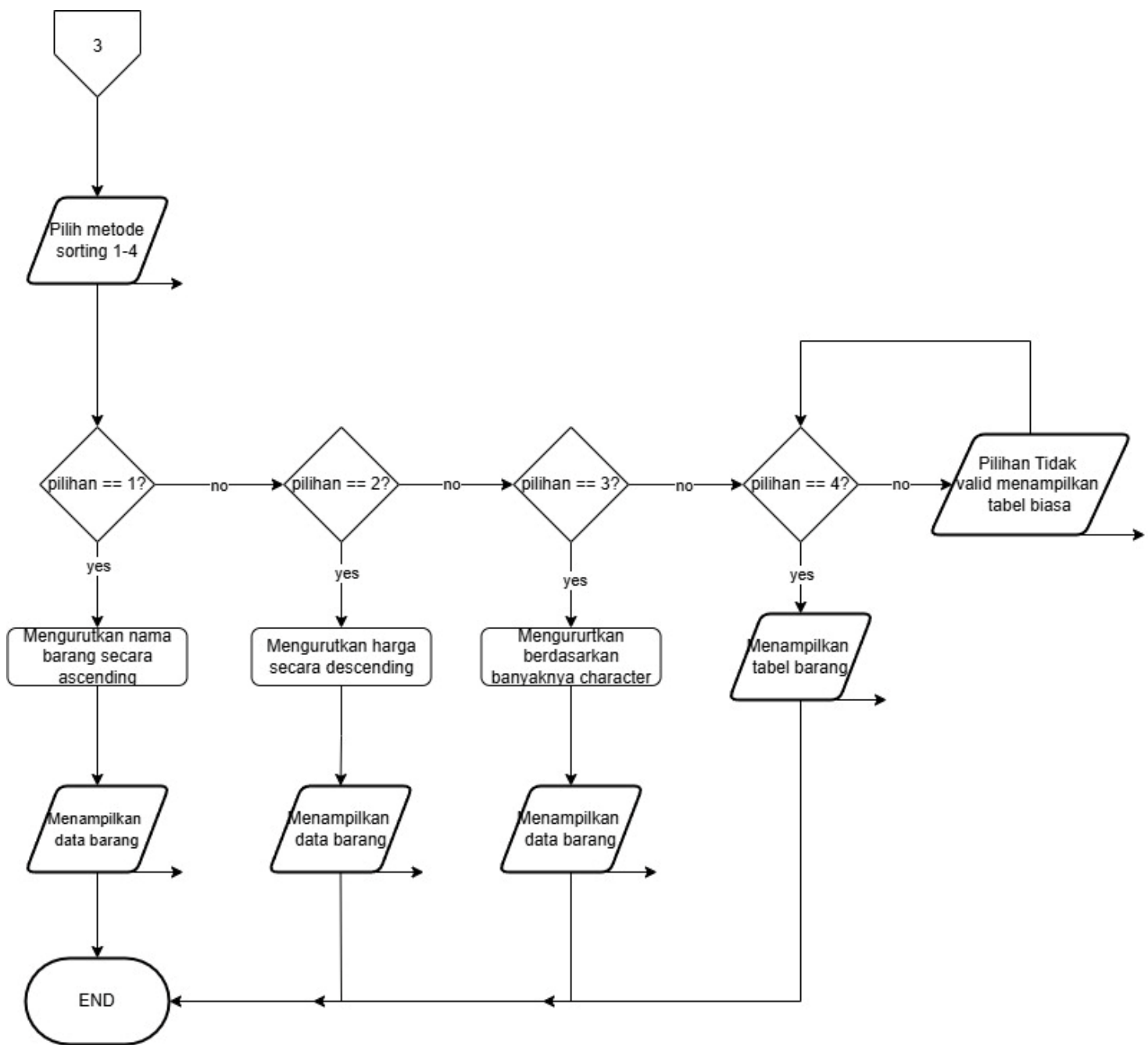
## 1. Flowchart



Gambar 1.1 Menu login dan register



Gambar 1.2 CRUD



Gambar 1.3 Sorting

## 2. Analisis Program

Program ini adalah sistem manajemen barang aksesoris komputer berbasis dengan fitur login dan register

### 1. Autentikasi Pengguna

- Pengguna harus login terlebih dahulu dengan username dan password.
- Jika salah memasukkan password lebih dari 3 kali, login akan gagal.
- Terdapat fitur pendaftaran pengguna baru (register).

### 2. Manajemen Barang (CRUD - Create, Read, Update, Delete)

- **Tambah Barang:** Pengguna bisa menambahkan barang dengan kode, nama, dan harga.
- **Tampilkan Barang:** Menampilkan daftar barang yang telah tersimpan.
- **Hapus Barang:** Menghapus barang dari daftar berdasarkan nomor urutnya.
- **Edit Barang:** Mengubah nama dan harga barang berdasarkan nomor urutnya.

### 3. Struktur Data

- Menggunakan struct untuk menyimpan data barang dan pengguna.
- Data barang disimpan dalam array dataBarang[MAX\_DATA].
- Data pengguna disimpan dalam array users[MAX\_USER].

### 4. Looping Menu

- Program berjalan dalam loop hingga pengguna memilih opsi keluar.
- Setelah login berhasil, pengguna dapat mengakses menu utama untuk mengelola barang.

## 3. Source code

### A. Deklarasi dan Inisialisasi

```

#include <iostream>
#include <iomanip>
#include <string>
#include <cstdlib>

using namespace std;

const int MAX_DATA = 100;
const int MAX_USER = 15;

struct InfoBarang {
    string nama;
    int harga;
};

struct Barang {
    string kode;
    InfoBarang info;
};

struct loginData {
    string nama;
    string password;
};

struct User {
    loginData login;
};

Barang dataBarang[MAX_DATA];
User users[MAX_USER];
int jumlahData = 0;
int jumlahUser = 1;

```

## B. Fungsi Validasi dan Input

```
// Fungsi rekursif validasi angka
bool isNumber(string input, int index = 0) {
    if (index >= input.length()) return true;
    if (!isdigit(input[index])) return false;
    return isNumber(input, index + 1);
}
```

## C. Fungsi Sorting

```
// Fungsi Sorting Nama Barang Ascending (A-Z)
void sortAscendHuruf(Barang* arr, int count) {
    for (int i = 0; i < count - 1; i++) {
        for (int j = i + 1; j < count; j++) {
            if (arr[i].info.nama > arr[j].info.nama) {
                swap(arr[i], arr[j]);
            }
        }
    }
}

// Fungsi Sorting Harga Barang Descending (besar ke kecil)
void sortDescendHarga(Barang* arr, int count) {
    for (int i = 0; i < count - 1; i++) {
        for (int j = i + 1; j < count; j++) {
            if (arr[i].info.harga < arr[j].info.harga) { // descending
                swap(arr[i], arr[j]);
            }
        }
    }
}
```

```
// Fungsi Sorting Panjang Nama Barang Descending (panjang ke pendek)
void sortDescendPanjangNama(Barang* arr, int count) {
    for (int i = 0; i < count - 1; i++) {
        for (int j = i + 1; j < count; j++) {
            if (arr[i].info.nama.length() < arr[j].info.nama.length()) { //
descending
                swap(arr[i], arr[j]);
            }
        }
    }
}
```

#### D. Fungsi untuk Menampilkan Barang

```
void cetakBarang(const Barang& barang) {
    cout << barang.kode << " - " << barang.info.nama << " - Rp" <<
barang.info.harga << endl;
}

void cetakBarang(const Barang* arr, int n) {
    cout << "\n-----\n";
    cout << "| No | Kode | Nama Barang | Harga |";
    cout << "\n-----\n";
    for (int i = 0; i < n; i++) {
        cout << "| " << setw(2) << i + 1 << " | " << setw(8) << arr[i].kode << " |
"
        << setw(17) << arr[i].info.nama << " | " << setw(8) <<
arr[i].info.harga << " |\n";
    }
    cout << "-----\n";
}
```



```

// Fungsi Tampilkan barang
void tampilkanBarang(Barang* arr, int count) {
    system("cls");
    if (count == 0) {
        cout << "Data masih kosong!\n";
        return;
    }

    int pilihanSort;

    cout << "|=====|\n";
    cout << "|          Pilih metode sorting:          |\n";
    cout << "|=====|\n";
    cout << "|1. Huruf (Nama Barang) Ascending          |\n";
    cout << "|2. Harga Descending                       |\n";
    cout << "|3. Panjang Nama Descending                |\n";
    cout << "|4. Tanpa Sorting (Tampilan Asli)          |\n";
    cout << "|=====|\n";
    cout << "Pilihan: ";
    cin >> pilihanSort;

    switch (pilihanSort) {
        case 1:
            sortAscendHuruf(arr, count); // pakai nama baru
            break;
        case 2:
            sortDescendHarga(arr, count);
            break;
        case 3:
            sortDescendPanjangNama(arr, count);
            break;
        case 4:
            // Tidak sorting
            break;
        default:

```

```

        cout << "Pilihan tidak valid. Menampilkan tanpa sorting.\n";
        break;
    }

    // Setelah sorting (atau tidak sorting), cetak tabel
    cetakBarang(arr, count);
}

```

## E. Fungsi untuk Mengelola Data Barang

```

// Fungsi Tambah barang
void tambahBarang(Barang* arr, int& count) {
    system("cls");
    if (count >= MAX_DATA) {
        cout << "Data penuh!\n";
        return;
    }
    Barang b;
    string hargaStr;
    bool valid = false;

    cout << "Masukkan Kode Barang: ";
    cin.ignore();
    getline(cin, b.kode);
    cout << "Masukkan Nama Barang: ";
    getline(cin, b.info.nama);

    do {
        cout << "Masukkan Harga Barang: ";
        getline(cin, hargaStr);
        valid = isNumber(hargaStr);
        if (!valid) cout << "Harga harus berupa angka!\n";
    } while (!valid);
}

```

```

        b.info.harga = stoi(hargaStr);
        arr[count++] = b;
        cout << "Data berhasil ditambahkan!\n";
    }

// Fungsi Tampilkan barang
void tampilkanBarang(const Barang* arr, int count) {
    system("cls");
    if (count == 0) {
        cout << "Data masih kosong!\n";
    } else {
        cetakBarang(arr, count);
    }
}

// Fungsi Hapus barang
void hapusBarang(Barang* arr, int& count) {
    system("cls");
    if (count == 0) {
        cout << "Data masih kosong!\n";
        return;
    }
    cetakBarang(arr, count);
    cout << "Masukkan nomor barang yang akan dihapus: ";
    int index;
    cin >> index;

    if (index > 0 && index <= count) {
        for (int i = index - 1; i < count - 1; i++) {
            arr[i] = arr[i + 1];
        }
        count--;
        cout << "Data berhasil dihapus!\n";
    } else {

```

```

        cout << "Nomor tidak valid!\n";
    }
}

// Fungsi Edit barang
void editBarang(Barang* arr, int count) {
    system("cls");
    if (count == 0) {
        cout << "Data masih kosong!\n";
        return;
    }
    cetakBarang(arr, count);
    cout << "Masukkan nomor barang yang akan diedit: ";
    int index;
    cin >> index;
    cin.ignore();

    if (index >= 1 && index <= count) {
        cout << "Masukkan Nama Baru: ";
        getline(cin, arr[index - 1].info.nama);
        cout << "Masukkan Harga Baru: ";
        cin >> arr[index - 1].info.harga;
        cout << "Data berhasil diubah!\n";
    } else {
        cout << "Nomor tidak valid!\n";
    }
}

```

## F. Fungsi Login dan Registrasi

```

// Fungsi Login
int login(User* users, int jumlahUser) {
    string nama, password;
    int attempts = 0;

```

```

while (attempts < 3) {
    system("cls");
    cout << "\n===== LOGIN =====\n";
    cout << "Nama: ";
    cin.ignore();
    getline(cin, nama);
    cout << "Password: ";
    getline(cin, password);

    for (int i = 0; i < jumlahUser; i++) {
        if (nama == users[i].login.nama && password ==
users[i].login.password) {
            return i;
        }
    }

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

return -1;
}

// Fungsi Registrasi
void registrasi(User* users, int& jumlahUser) {
    string nama, password;
    system("cls");
    cout << "\n===== REGISTER =====\n";
    cout << "Masukkan Nama: ";
    cin.ignore();
    getline(cin, nama);
    cout << "Masukkan Password: ";
    getline(cin, password);

```

```

    users[jumlahUser].login.nama = nama;
    users[jumlahUser].login.password = password;
    jumlahUser++;
    cout << "Registrasi berhasil!\n";
}

```

### G. Fungsi Main dan tampilan menu utama serta CRUD

```

int main() {
    users[0].login.nama = "dzaki";
    users[0].login.password = "056";

    int pilih;
    do {
        system("cls");
        cout << "=====\n";
        cout << "|          MENU UTAMA          |\n";
        cout << "=====\n";
        cout << "| 1. Login                      |\n";
        cout << "| 2. Register                   |\n";
        cout << "| 3. Keluar                     |\n";
        cout << "=====\n";
        cout << "Pilih: ";
        cin >> pilih;

        if (pilih == 1) {
            int userIndex = login(users, jumlahUser);
            if (userIndex != -1) {
                int menu;
                do {
                    system("cls");

```

```

        cout << "=====\n";
        cout << "|          MENU UTAMA          |\n";
        cout << "=====\n";
        cout << "| 1. Tambah Barang          |\n";
        cout << "| 2. Tampilkan Barang      |\n";
        cout << "| 3. Hapus Barang          |\n";
        cout << "| 4. Edit Barang           |\n";
        cout << "| 5. Logout                |\n";
        cout << "=====\n";
        cout << "Pilih: ";
        cin >> menu;

        switch (menu) {
            case 1:
                tambahBarang(dataBarang, jumlahData);
                cin.ignore(); cin.get();
                break;
            case 2:
                tampilkanBarang(dataBarang, jumlahData);
                cin.ignore(); cin.get();
                break;
            case 3:
                hapusBarang(dataBarang, jumlahData);
                cin.ignore(); cin.get();
                break;
            case 4:
                editBarang(dataBarang, jumlahData);
                cin.ignore(); cin.get();
                break;
        }
    } while (menu != 5);
} else {
    cout << "Login gagal 3 kali silahkan keluar.\n";
    cin.ignore(); cin.get();
}

```

```

        exit(0);
    }
    } else if (pilih == 2) {
        registrasi(users, jumlahUser);
        cin.ignore(); cin.get();
    }
} while (pilih != 3);

return 0;
}

```

#### 4.Hasil output

```

=====
|              MENU UTAMA              |
=====
| 1. Login                             |
| 2. Register                          |
| 3. Keluar                            |
=====
Pilih: 3
PS C:\Users\ASUS\Praktikum-APL\Post-test\Post-test-4>

```

Gambar 4. 1 Menu utama

```

===== LOGIN =====
Nama: dzaki
Password: 056

```

Gambar 4. 2 Menu login

```

===== REGISTER =====
Masukkan Nama: jakjajk
Masukkan Password: 123

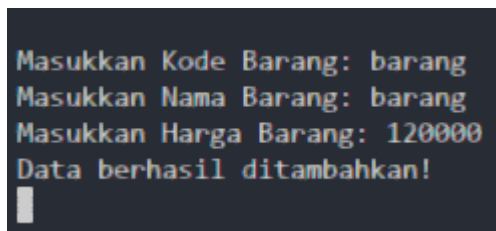
```

Gambar 4. 3 Menu Register

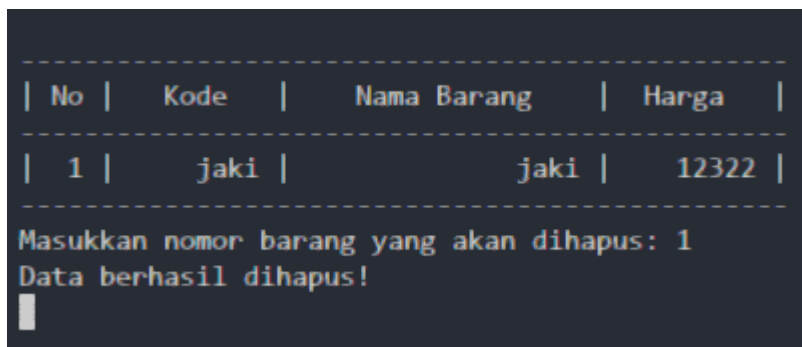




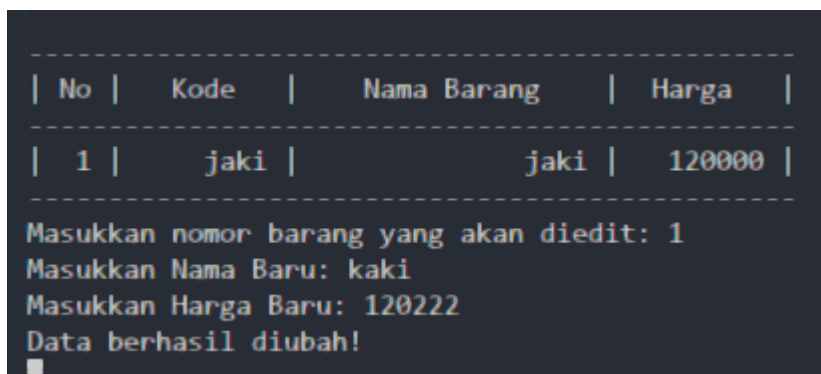
Gambar 4. 4 Menu CRUD



Gambar 4. 5 Menu Tambah barang



Gambar 4. 6 Hapus barang



Gambar 4. 7 Edit barang

```
=====
|                               |
|          MENU UTAMA         |
|                               |
| 1. Login                    |
| 2. Register                 |
| 3. Keluar                   |
|                               |
Pilih: 3
PS C:\Users\ASUS\Praktikum-APL\Post-test\Post-test-4>
```

Gambar 4. 8 Menu Keluar

```
=====
|                               |
|      Pilih metode sorting:   |
|                               |
| 1. Huruf (Nama Barang) Ascending
| 2. Harga Descending
| 3. Panjang Nama Descending
| 4. Tanpa Sorting (Tampilan Asli)
|                               |
Pilihan: 1

-----
| No | Kode | Nama Barang | Harga |
-----
| 1 | kalo1234 |      kalo | 1300 |
| 2 |      kalo |     kalo12 | 1200 |
-----
```

Gambar 4. 9 Sorting ascending huruf

```
=====
|                               |
|      Pilih metode sorting:   |
|                               |
| 1. Huruf (Nama Barang) Ascending
| 2. Harga Descending
| 3. Panjang Nama Descending
| 4. Tanpa Sorting (Tampilan Asli)
|                               |
Pilihan: 2

-----
| No | Kode | Nama Barang | Harga |
-----
| 1 | kalo1234 |      kalo | 1300 |
| 2 |      kalo |     kalo12 | 1200 |
-----
```

Gambar 4. 10 Sorting descending harga

```
|=====|
|      Pilih metode sorting:      |
|=====|
|1. Huruf (Nama Barang) Ascending|
|2. Harga Descending             |
|3. Panjang Nama Descending      |
|4. Tanpa Sorting (Tampilan Asli)|
|=====|
Pilihan: 3

-----
| No | Kode | Nama Barang | Harga |
-----
| 1 |  kalo |      kalo12 | 1200 |
| 2 | kalo1234 |      kalo | 1300 |
-----
|
```

Gambar 4. 11 Sorting banyaknya char

```
|=====|
|      Pilih metode sorting:      |
|=====|
|1. Huruf (Nama Barang) Ascending|
|2. Harga Descending             |
|3. Panjang Nama Descending      |
|4. Tanpa Sorting (Tampilan Asli)|
|=====|
Pilihan: 4

-----
| No | Kode | Nama Barang | Harga |
-----
| 1 |  kalo |      kalo12 | 1200 |
| 2 | kalo1234 |      kalo | 1300 |
-----
|
```

Gambar 4. 12 tabel tanpa sorting

## 5. Langkah-Langkah Git pada VSCode

### 1. Git add .

```
PS C:\Users\ASUS\Praktikum-APL\Post-test\Post-test-6> git add .
```

Git add . berfungsi untuk menambahkan file-file yang ada di dalam suatu folder atau repositori karena . sendiri berarti semua

### 2. Git commit -m "done"

```
PS C:\Users\ASUS\Praktikum-APL\Post-test\Post-test-6> git commit -m "done"
[main 33fd6db] done
3 files changed, 326 insertions(+)
create mode 100644 Post-test/Post-test-6/.vscode/settings.json
create mode 100644 Post-test/Post-test-6/2409106056-MuhammadDzakiRifai-PT5.cpp
create mode 100644 Post-test/Post-test-6/2409106056-MuhammadDzakiRifai-PT5.exe
```

Git commit -m "done" berfungsi untuk mencommit pekerjaan kita sebelum di push ke github

-m sendiri berarti message karena saat mencommit kita harus meninggalkan pesan dan pesan yang ditinggalkan adalah "done"

### 3. Git push origin main

```
PS C:\Users\ASUS\Praktikum-APL\Post-test\Post-test-6> git push origin main
info: please complete authentication in your browser...
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 20 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 680.68 KiB | 8.01 MiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/dzak2323/Praktikum-APL.git
8c0c815..33fd6db main -> main
```

Git push origin main berfungsi untuk mengupload file yang sudah kita add dan commit ke github kita.