

LAPORAN PRAKTIKUM
POSTTEST (6)
ALGORITMA PEMROGRAMAN LANJUT

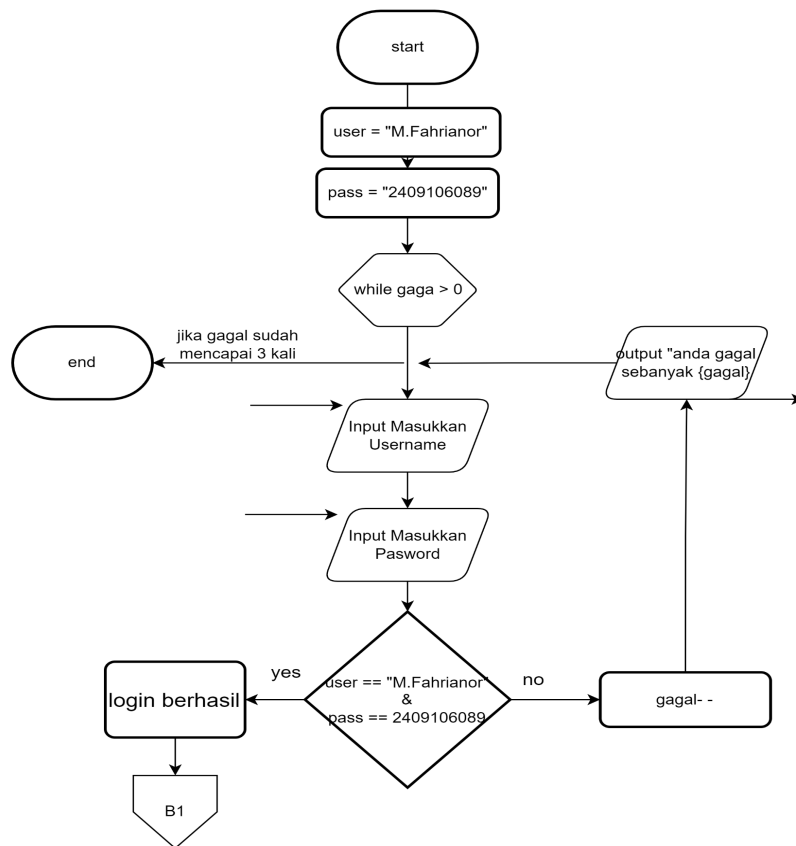


Disusun oleh:
M.Fahrianor (2409106089)
Kelas (B2'24)

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

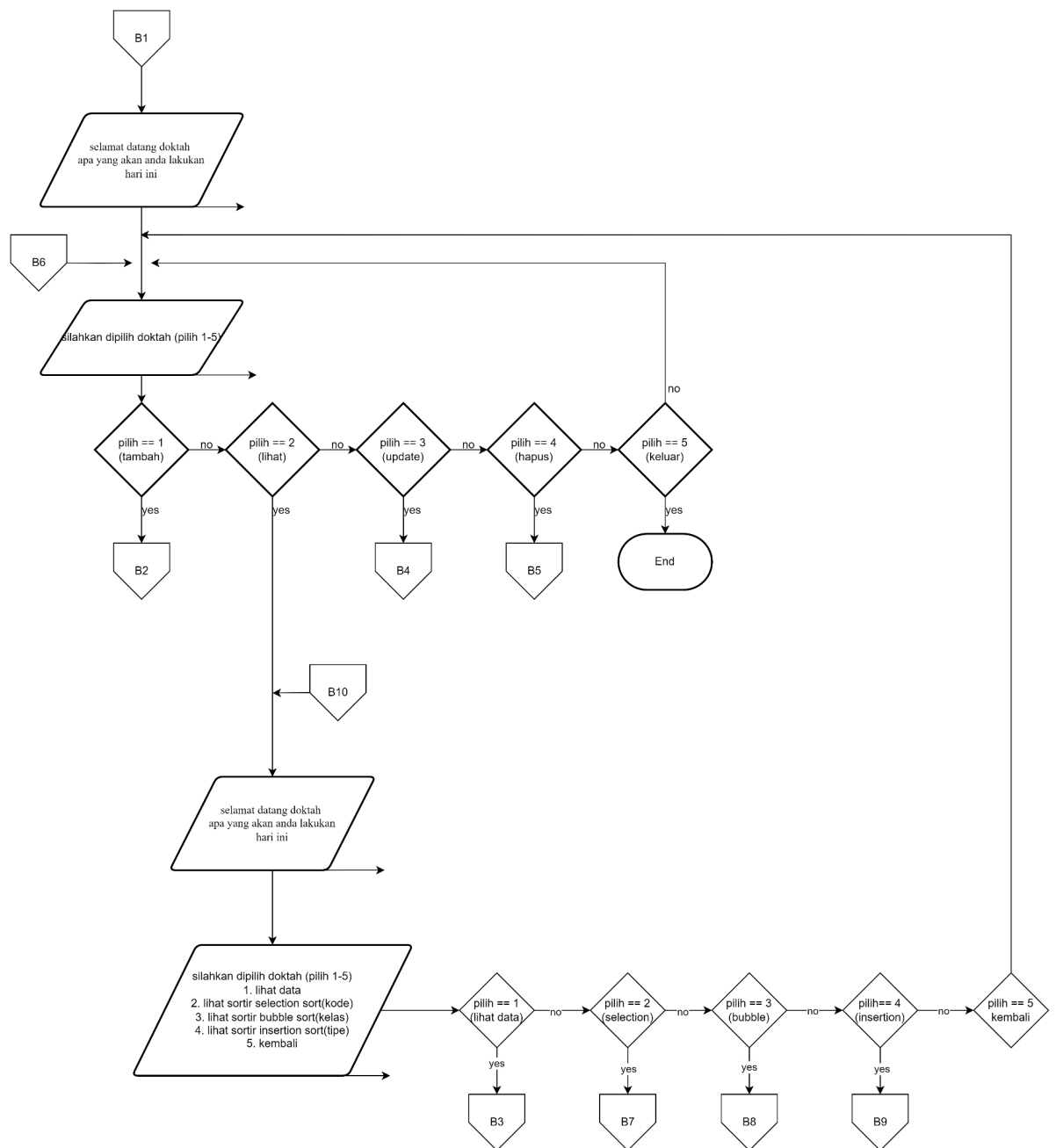
1. Flowchart

A. Login



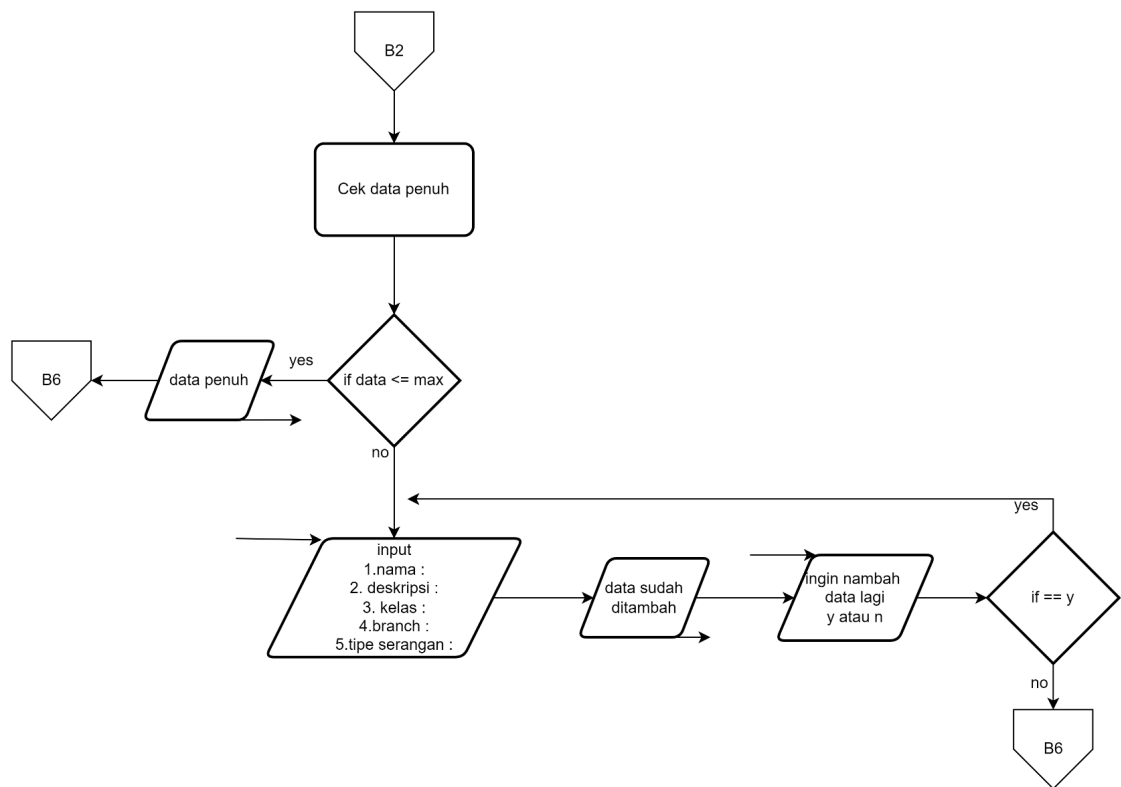
Gambar 1.1 Login

B. Menu Utama



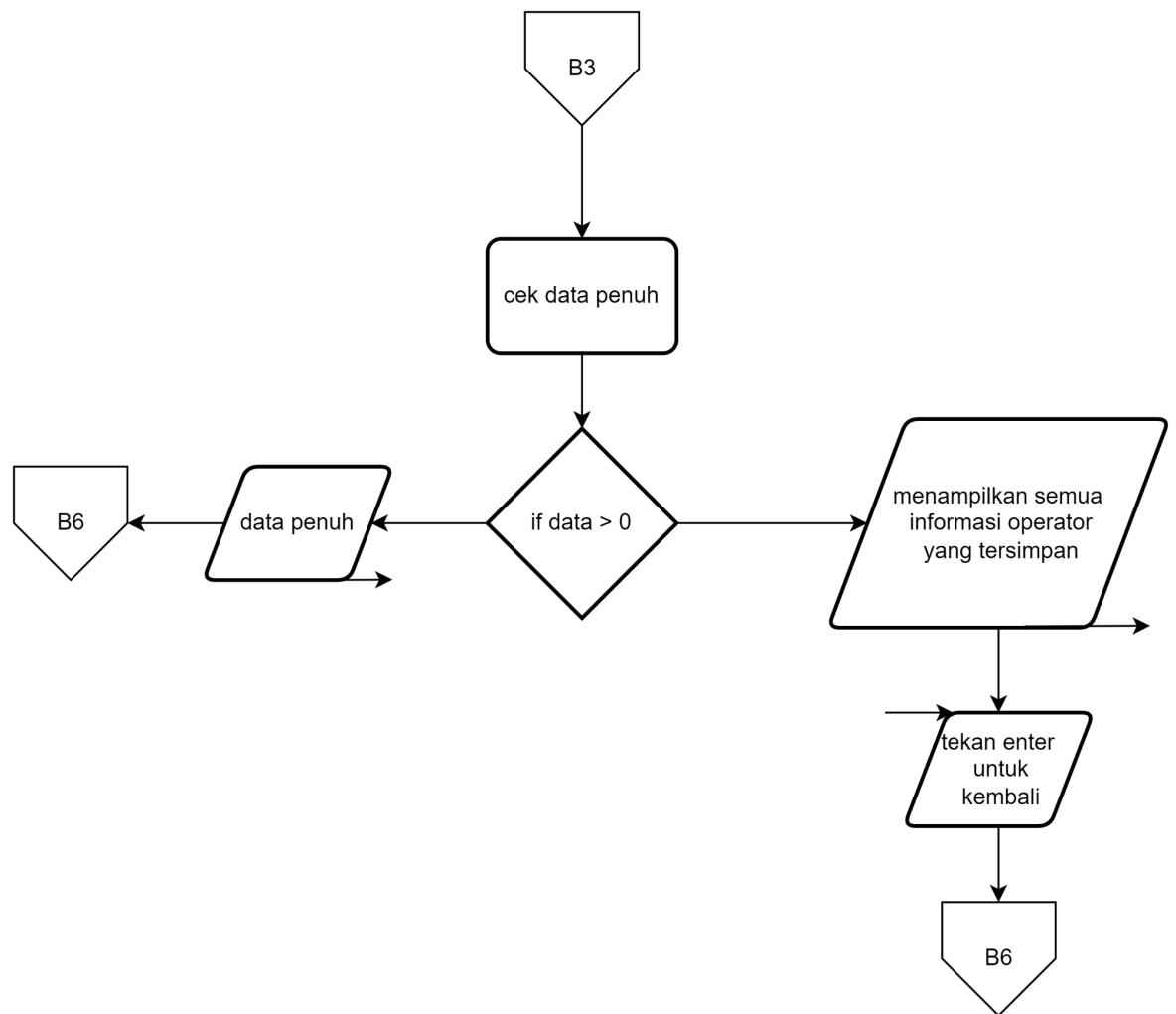
Gambar 1.2 Menu utama

C. Tambah data



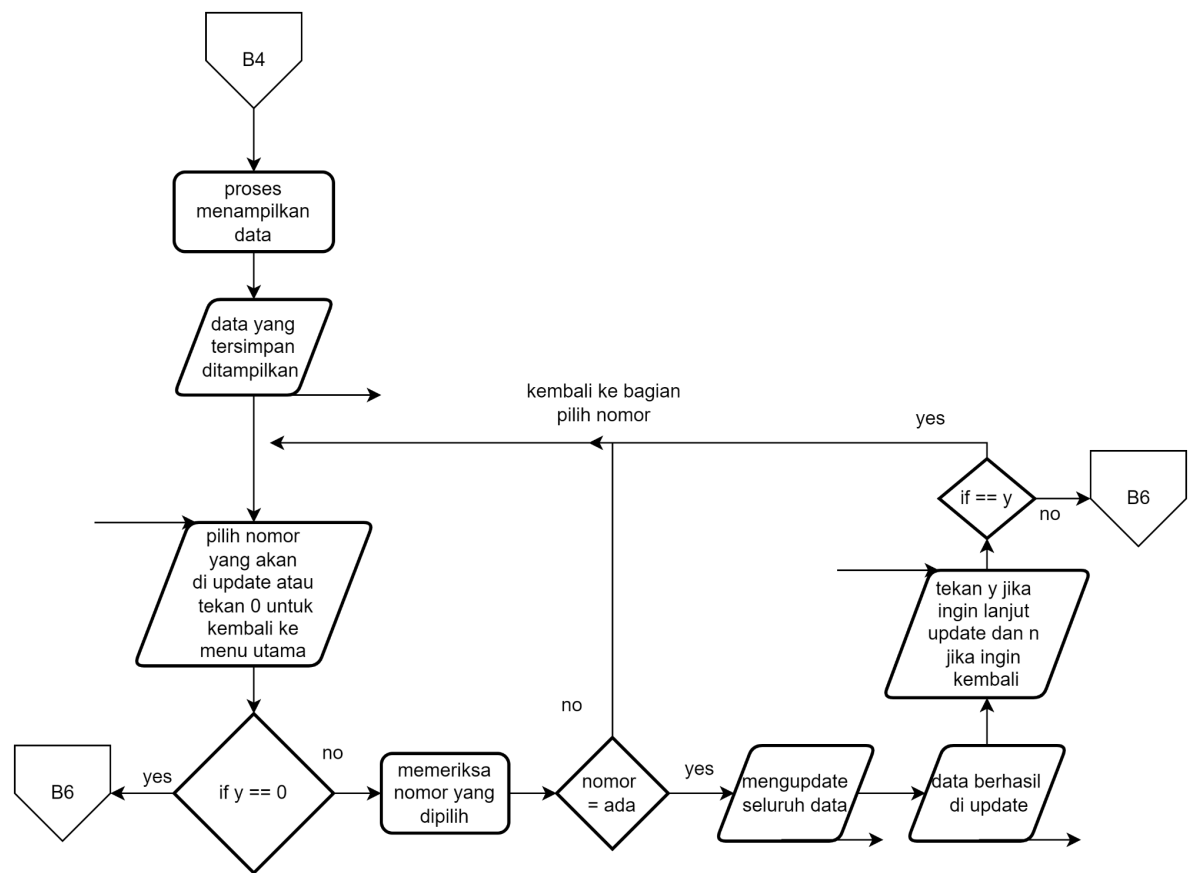
Gambar 1.3 Tambah data

D. Lihat data



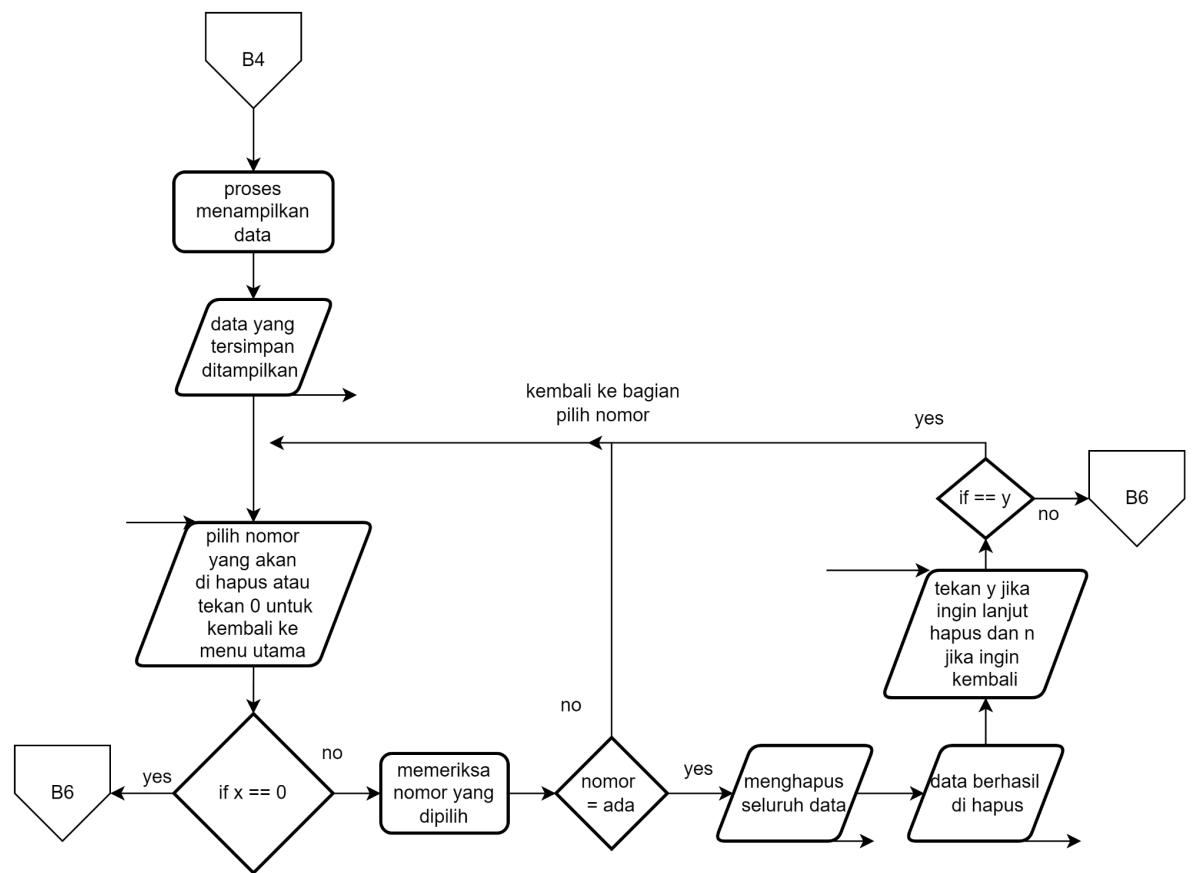
Gambar 1.4 Lihat data

E. Update data



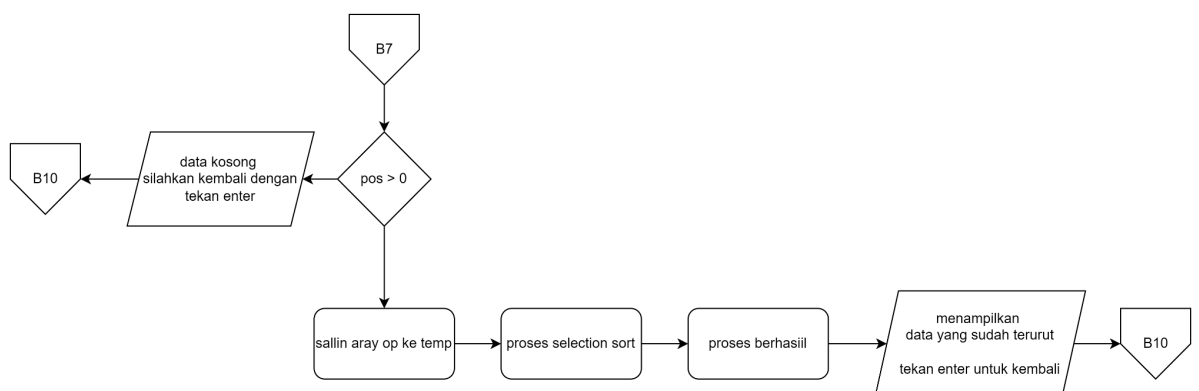
Gambar 1.5 Update data

F. Hapus data



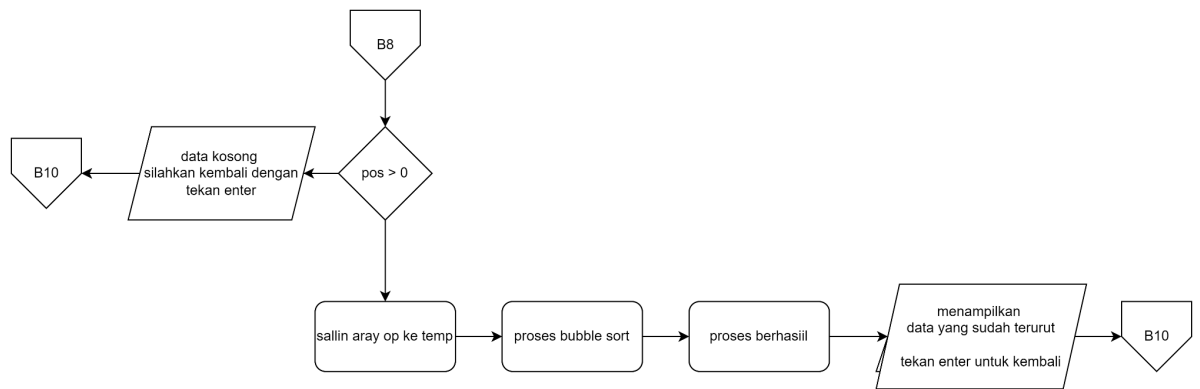
Gambar 1.6 Hapus data

G. Selection sort



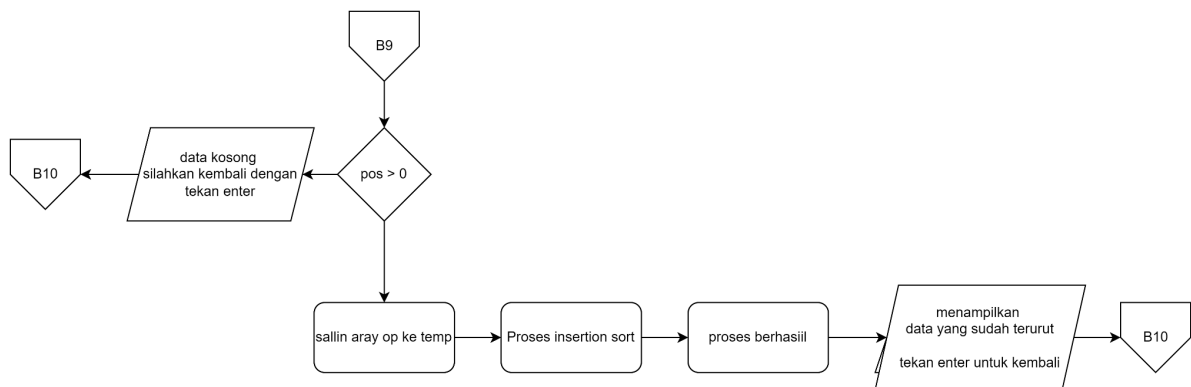
Gambar 1.7 Selection sort

H. Bubble sort



Gambar 1.8 Bubble sort

I. Insertion sort



Gambar 1.9 Insertion sort

2. Analisis Program

2.1 Deskripsi Singkat Program

Program ini merupakan program CRUD (create, read, update, delete) yang memiliki judul “Sistem informasi karakter game Arknights”, sistem ini dibuat dengan menggunakan fungsi, parameter, prosedur, dan juga menggunakan prinsip dari pointer yang dapat terlihat pada program terdapat fungsi dengan parameter address-of dan dereference yang dimana pada program ini akan terdapat 5 sub program yang dibuat menggunakan fungsi. Didalam program tepatnya sebelum masuk ke menu utama akan dilakukan login yang dimana pengguna akan diminta memasukkan username dan password, dan ketika pengguna salah sebanyak 3 kali maka program akan otomatis berhenti, tetapi jika berhasil pengguna akan masuk ke dalam menu utama, yang di dalam menu utama ada beberapa pilihan yaitu, tambah data operator, lihat data operator, update data operator, hapus data operator, dan keluar. pada program ini pengguna tidak akan bisa keluar kecuali pengguna memilih opsi keluar. selain menggunakan prinsip pointer program ini juga menggunakan sistem sorting, seperti yang terlihat di program ini menggunakan sorting selection sort, bubble sort, dan insertion sort baik ascending maupun descending

3. Source Code

A. Struct dan Data awal

Ini adalah program array of struct

Source Code:

```
// maksimal data 20
#define max 20
int pos = 8;

// struct
struct Operator
{
    int kode;
    string karakter;
    string deskripsi;
    string kelas;
```

```

    string branch;
    string tipe;
};
Operator op[max];
Operator temp[max];
void dataawal(){
    op[0].kode = 215;
    op[0].karakter = "Eyjafjalla";
    op[0].deskripsi = "Nama asli Adele Nauman";
    op[0].kelas = "Caster";
    op[0].branch = "Core caster";
    op[0].tipe = "Arts";
}

```

Gambar 3.1 Struct dan Data awal

B. Fitur Login

Ini adalah fitur login untuk pengguna

Source Code:

```

bool Login(string userbenar, string passbenar) {
    string username;
    string password;
    int gagal = 3;

    while (gagal > 0) {
        cout << "Masukkan username anda doktah: ";
        cin >> username;
        cout << "Masukkan password anda doktah: ";
        cin >> password;
        if (username == userbenar && password == passbenar) {
            return true;
        }else {
            gagal--;
            cout << "" << endl;
            cout << "Kesempatan kamu untuk masuk hanya tersisa " << gagal << ".  
Jika kamu gagal login maka kamu bukanlah doktah" << endl;
        }
        if (gagal == 0) {
            system("cls");
            cout << "=====" << endl;
            cout << "Kamu gagal. Kamu diblokir dari PRTS" << endl;
            cout << "=====" << endl;
            return 0;
        }
    }
}

```

```
    return false;
}
```

Gambar 3.2 Fitur Login

C. Menu Utama

Ini merupakan tampilan menu utama jika pengguna berhasil login

Source Code:

```
system("cls");
cout << "======" << endl;
cout << "Selamat datang doktah apa yang akan anda lakukan hari ini" << endl;
cout << "======" << endl;
cout << "" << endl;
cout << "======" << endl;
cout << "1. Tambah data operator\n";
cout << "2. Lihat data operator\n";
cout << "3. Update data operator\n";
cout << "4. Hapus data operator\n";
cout << "5. Keluar" << endl;
cout << "======" << endl;
cout << "" << endl;
cout << "Silahkan dipilih doktah: ";
cin >> pilih;
cin.ignore();
```

Gambar 3.3 Menu utama

D. Tambah Data Operator

Pada gambar ini merupakan tampilan tambah data informasi yang menggunakan prosedur dan parameter di dalamnya dan terdapat parameter dereference pada bagian (Operator *op,

int *pos), *pos >= max, dan op[*pos], yang dimana memiliki fungsi untuk mengakses dan memanipulasi data asli secara langsung dari memori.(Operator *op, int *pos) menunjukkan bahwa fungsi menerima alamat dari array Operator dan variabel int pos. Dengan menggunakan *pos >= max, program bisa membandingkan nilai posisi data saat ini dengan batas maksimum secara langsung, tanpa membuat salinan variabel. Sedangkan op[*pos] digunakan untuk menunjuk ke elemen ke berapa dalam array, proses penambahan data ini mencakup kode operator, nama operator, deskripsi operator, kelas operator, branch operator, dan tipe operator

Source Code:

```
void tambahdata(Operator *op, int *pos){
    string loop;
    system("cls");
    do
    {
        if (*pos >= max)
        {
            system("cls");
            cout << "=====" << endl;
            cout << "Data sudah max doktah\n";
            cout << "=====" << endl;
            cout << "" << endl;
            cout << "Tekan enter untuk kembali: ";
            cin.get();
            break;
        }else if (*pos < max){
            system("cls");
            cout << "=====" << endl;
            cout << "Masukkan kode operator: ";
            cin >> op[*pos].kode;
            cin.ignore();
            cout << "Masukkan nama operator: ";
            getline(cin, op[*pos].karakter);
            cout << "Masukkan deskripsi operator: ";
            getline(cin, op[*pos].deskripsi);
            cout << "Masukkan class operator: ";
            getline(cin, op[*pos].kelas);
            cout << "Masukkan branch operator: ";
            getline(cin, op[*pos].branch);
            cout << "Masukkan tipe serangan: ";
```

```

        getline(cin, op[*pos].tipe);
        cout << "=====" << endl;
        (*pos)++;
        cout << "" << endl;
        cout << "Apakah anda ingin menambah operator lagi doktah?
(y/n): ";

        cin >> loop;
        cin.ignore();
    }

```

Gambar 3.4 Tambah Data

E. Lihat Data Operator

Pada gambar dibawah ini merupakan program untuk melihat data, program ini dinuat menjadi bentuk tabel agar terlihat rapi dengan menggunakan command setw yang artinya setting width, dan program ini juga menggunakan prinsip pointer.

Source Code:

```

void Lihatope(Operator *op, int *pos) {
    system("cls");
    if (*pos > 0) {
        cout << "Data Operator Tersimpan:\n\n";

        cout << "+====+====+====+====+====+====+====+\n";
        cout << "| No|Kode| Nama| Deskripsi| Kelas | Branch| Tipe |\n";
        cout << "+====+====+====+====+====+====+====+\n";

        for (int a = 0; a < *pos; a++) {
            cout << "| " << setw(2) << a + 1 << " ";
            cout << "| " << setw(6) << left << to_string(op[a].kode).substr(0,
6);

            cout << "| " << setw(13) << left << op[a].karakter.substr(0,13);
            cout << "| " << setw(28) << left << op[a].deskripsi.substr(0,28);
            cout << "| " << setw(14) << left << op[a].kelas.substr(0,14);
            cout << "| " << setw(15) << left << op[a].branch.substr(0,15);
            cout << "| " << setw(14) << left << op[a].tipe.substr(0,14) <<
"/\n";
        }

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

```

```

    } else {
        cout << "Data kosong\n";
    }
    cout << "Tekan enter untuk kembali: ";
    cin.get();
}

```

Gambar 3.5 Lihat Data Operator

F. Update Data Operator

Pada gambar ini merupakan fitur update data yang dimana saat memilih fitur ini di dalam menu maka yang pertama akan dilakukan sistem adalah menampilkan data yang tersimpan dan kemudian pengguna dapat memilih data nomor berapa yang akan di-update. Fitur ini menggunakan prosedur dan parameter dengan dereference, yaitu pada bagian (Operator *op, int *pos), *pos > 0, dan op[a]. Bagian (Operator *op, int *pos) menunjukkan bahwa fungsi menerima alamat dari array Operator dan jumlah data yang tersimpan. Penggunaan *pos > 0 berfungsi untuk mengecek apakah ada data yang dapat di-update secara langsung dari memori. Sedangkan op[a] digunakan untuk menampilkan nama-nama operator yang tersimpan berdasarkan jumlah data yang telah dimasukkan. proses update data ini mencakup update kode operator, update nama operator, update deskripsi operator, update kelas operator, update branch operator, dan update tipe operator.

Source Code:

```

void updateope(Operator *op, int *pos){
    string loop;
    if (*pos > 0)
    {
        int y;
        do
        {
            system("cls");
            cout << "======" << endl;
            cout << "Daftar Operator yang Tersimpan:\n";
            cout << "======" << endl;
            for (int a = 0; a < *pos; a++) {
                cout << a + 1 << ". " << op[a].karakter << endl;
            }
        }
    }
}

```

```

cout << "=====" << endl;
cout << "Tekan 0 jika ingin kembali ke menu utama" << endl;
cout << "=====" << endl;
cout << "" << endl;
cout << "\nMasukkan nomor operator yang ingin diupdate: ";
cin >> y;
cin.ignore();
if (y == 0 )
{
    cout << "Kita kembali ke menu utama doktah\n";
    break;
} else if(y > 0 && y <= *pos)
{
    system("cls");
    cout << "=====" << endl;
    cout << "update kode operator: ";
    cin >> op[y-1].kode;
    cout << "Update nama operator menjadi: ";
    getline(cin, op[y-1].karakter);
    cout << "Update deskripsi operator menjadi: ";
    getline(cin, op[y-1].deskripsi);
    cout << "Update class operator menjadi: ";
    getline(cin, op[y-1].kelas);
    cout << "Update branch operator menjadi: ";
    getline(cin, op[y-1].branch);
    cout << "Update tipe serangan operator menjadi: ";
    getline(cin, op[y-1].tipe);
    cout << "Data operator berhasil di update silahkan lihat
pada bagian lihat data\n";
    cout << "=====" << endl;
} else {
    cout << "=====" << endl;
    cout << "Data operator tidak ada mohon untuk memasukkan
nomor yang benar" << endl;
    cout << "=====" << endl;
}
}

```

Gambar 3.6 Update Data Operator

G. Hapus Data Operator

pada gambar ini merupakan fitur hapus data yang dimana saat memilih fitur ini di dalam menu maka yang pertama akan dilakukan sistem adalah menampilkan data yang tersimpan dan kemudian pengguna dapat memilih data nomor berapa yang akan dihapus hal ini sama seperti penjelasan update data yang menjadi pembeda adalah pada update program akan

mengganti data lama pada array menjadi data baru sedangkan pada hapus program akan menghapus data dari array yang tersimpan

Source Code:

```
void hapusope(Operator *op, int *pos){
    string loop;
    if (*pos > 0)
    {
        int x;

        do
        {
            system("cls");
            cout << "======" << endl;
            cout << "Daftar Operator yang Tersimpan:\n";
            cout << "======" << endl;
            for (int a = 0; a < *pos; a++) {
                cout << a + 1 << ". " << op[a].karakter << endl;
            }

            cout << "======" << endl;
            cout << "Tekan 0 jika ingin kembali ke menu utama" << endl;
            cout << "======" << endl;
            cout << "" << endl;
            cout << "\nMasukkan nomor operator yang ingin dihapus: ";
            cin >> x;
            cin.ignore();

            if (x == 0)
            {
                cout << "Kita kembali ke menu utama doktah\n";
                break;
            }else if (x > 0 && x <= *pos)
            {
                for (int i = x; i < *pos; i++)
                {
                    op[i-1].kode = op[i].kode;
                    op[i-1].karakter = op[i].karakter;
                    op[i-1].deskripsi = op[i].deskripsi;
                    op[i-1].kelas = op[i].kelas;
                    op[i-1].branch = op[i].branch;
                    op[i-1].tipe = op[i].tipe;
                }
                (*pos)--;
            }
        }
    }
}
```

Gambar 3.7 Hapus Data Operator

H. Log Out

Gambar ini adalah fitur log out

Source Code:

```
void logout(){
    system("cls");
    cout << "======" << endl;
    cout << "Selamat tinggal doktah" << endl;
    cout << "======" << endl;
}
```

Gambar 3.8 Log Out

J. Menu pilih sorting

Pada gambar dibawah ini adalah menu pilihan jika kita memilih nomor 2 pada menu utama.

Source Code:

```
else if (pilih == 2)
{
    do
    {
        system("cls");
        cout << "======" << endl;
        cout << "Selamat datang doktah apa yang akan anda lakukan hari ini" << endl;

        cout << "======" << endl;
        cout << "" << endl;
        cout << "======" << endl;
        cout << "1. Lihat data asli\n";
        cout << "2. Lihat data berdasarkan kelas operator(selection sort)\n";
        cout << "3. Lihat data berdasarkan kode operator(bubble sort)\n";
        cout << "4. Lihat data berdasarkan tipe operator(insertion sort)\n";
        cout << "5. kembali\n";
        cout << "======" << endl;
        cout << "" << endl;
        cout << "Silahkan dipilih doktah: ";
        cin >> pilihlihat;
        cin.ignore();
        if (pilihlihat == 1)
        {
            lihatope(op, &pos);
        }
    }
}
```

```

    }else if (pilihlihat == 2)
    {
        lihatope_selectionsort(op, &pos);
    }else if (pilihlihat == 3)
    {
        lihatope_terurut(op, &pos);
    }else if (pilihlihat == 4)
    {
        lihatope_insertionSort(op, &pos);
    }
} while (pilihlihat != 5 );

```

Gambar 3.9 Menu pilih sorting

K. Selection sort

Pada gambar dibawah ini merupakan program sorting dengan menggunakan selection sort yang dimana lihatope_selectionsort digunakan untuk mengurutkan data array Operator berdasarkan kelas secara ascending menggunakan algoritma Selection Sort. Pertama, seluruh data operator disalin dari array op ke array temp untuk diproses tanpa mengubah data asli. Kemudian, untuk setiap elemen dalam array temp, program mencari data dengan nilai kelas terkecil di antara elemen yang tersisa dan menukarnya dengan elemen saat ini jika ditemukan nilai yang lebih kecil. Proses ini dilakukan berulang kali hingga seluruh data terurut dari nilai kelas terkecil ke terbesar. Fungsi ini hanya berjalan jika jumlah data operator (*pos) lebih dari 0

Source Code:

```

void lihatope_selectionsort(Operator *op, int *pos) {
    if (*pos > 0) {
        for (int i = 0; i < *pos; i++) {
            temp[i] = op[i];
        }

        for (int i = 0; i < *pos - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < *pos; j++) {
                if (temp[j].kelas < temp[minIndex].kelas) {
                    minIndex = j;
                }
            }
        }
    }
}

```

```

    }
    if (minIndex != i) {
        Operator t = temp[i];
        temp[i] = temp[minIndex];
        temp[minIndex] = t;
    }
}

```

Gambar 4.0 Selection sort

L. Bubble sort

Pada gambar dibawah ini merupakan program sorting dengan menggunakan Bubble sort yang dimana `lihatope_bubble` digunakan untuk mengurutkan array Operator berdasarkan kode secara descending menggunakan algoritma Bubble Sort. Pertama, semua data dari array `op` disalin ke array `temp` untuk diproses terpisah dari data aslinya. Kemudian, algoritma melakukan perbandingan antar elemen yang bersebelahan dan menukar posisinya jika nilai kode pada elemen kiri lebih kecil dari elemen kanan, sehingga nilai terbesar akan "mengambang" ke posisi akhir setiap kali satu iterasi selesai. Proses ini diulang sampai seluruh elemen dalam array `temp` tersusun dari kode terbesar ke terkecil. Fungsi ini hanya berjalan jika jumlah data operator (`*pos`) lebih dari 0.

Source Code:

```

void lihatope_bubble(Operator *op, int *pos){
    if (*pos > 0) {
        for (int i = 0; i < *pos; i++) {
            temp[i] = op[i];
        }

        for (int i = 0; i < *pos - 1; i++) {
            for (int j = 0; j < *pos - i - 1; j++) {
                if (temp[j].kode < temp[j+1].kode) {
                    Operator t = temp[j];
                    temp[j] = temp[j+1];
                    temp[j+1] = t;
                }
            }
        }
    }
}

```

Gambar 4.1 Bubble sort

M. Insertion sort

Pada gambar dibawah ini merupakan program sorting dengan menggunakan Insertion sort yang dimana lihatope_insertionSort berfungsi untuk mengurutkan array Operator berdasarkan tipe secara ascending menggunakan algoritma Insertion Sort. Pertama, semua data operator dari array op disalin ke array temp agar pengurutan tidak langsung mengubah data aslinya. Kemudian, fungsi memulai dari elemen kedua dan memilihnya sebagai key, lalu membandingkannya dengan elemen-elemen sebelumnya. Jika ditemukan elemen yang lebih besar dari key, elemen tersebut akan digeser satu posisi ke kanan. Setelah semua elemen yang lebih besar digeser, key akan ditempatkan di posisi yang tepat. Proses ini diulang hingga semua elemen tersusun dari tipe terkecil ke terbesar, dan fungsi hanya berjalan jika jumlah data operator (*pos) lebih dari 0.

Source Code:

```
void lihatope_insertionSort(Operator *op, int *pos) {
    if (*pos > 0) {
        for (int i = 0; i < *pos; i++) {
            temp[i] = op[i];
        }

        for (int i = 1; i < *pos; i++) {
            Operator key = temp[i];
            int j = i - 1;

            while (j >= 0 && temp[j].tipe > key.tipe) {
                temp[j + 1] = temp[j];
                j = j - 1;
            }
            temp[j + 1] = key;
        }
    }
}
```

Gambar 4.2 Insertion sort

4. Hasil Output

4.1 Hasil Output

```
PS C:\Users\ACER\Documents\praktikum-apl\post-t  
Masukkan username anda doktah: M.Fahrianor  
Masukkan password anda doktah: 2409106089
```

Gambar 4.1 Login

```
PS C:\Users\ACER\Documents\praktikum-apl\post-test\post-test-37 : (2409106089-M.Fahrianor-PRTS)  
Masukkan username anda doktah: M.Fahrianor  
Masukkan password anda doktah: 2409  
  
Kesempatan kamu untuk masuk hanya tersisa 2. Jika kamu gagal login maka kamu bukanlah doktah  
Masukkan username anda doktah: 
```

Gambar 4.2 Gagal kurang dari 3 kali

```
=====  
Kamu gagal. Kamu diblokir dari PRTS  
=====
```

Gambar 4.3 Gagal sudah mencapai 3 kali

```
=====  
Selamat datang doktah apa yang akan anda lakukan hari ini  
=====  
  
=====  
1. Tambah data operator  
2. Lihat data operator  
3. Update data operator  
4. Hapus data operator  
5. Keluar  
=====  
Siahhkan dipilih doktah: 
```

Gambar 4.4 Login berhasil dan menu utama

```

=====
Masukkan kode operator: 445
Masukkan nama operator: Mostima
Masukkan deskripsi operator: Seorang legatus dari laterano
Masukkan class operator: Caster
Masukkan branch operator: Splash
Masukkan tipe serangan: Arts
=====

Apakah anda ingin menambah operator lagi doktah? (y/n): █

```

Gambar 4.5 Tambah data

```

=====
Selamat datang doktah apa yang akan anda lakukan hari ini
=====

1. Lihat data asli
2. Lihat data berdasarkan kelas operator(selection sort)
3. Lihat data berdasarkan kode operator(bubble sort)
4. Lihat data berdasarkan tipe operator(insertion sort)
5. kembali
=====

Silahkan dipilih doktah: █

```

Gambar 4.6 Tampilan lihat data

Data Operator Tersimpan:

No	Kode	Nama	Deskripsi	Kelas	Branch	Tipe
1	215	Eyjafjalla	Nama asli Adele Nauman	Caster	Core caster	Arts
2	2	Ines	Ines adalah tentara bayaran	Vanguard	Agent	Physical
3	182	Nymph	Seorang wanga kazdel	Caster	Primal caster	Arts
4	3	Kal'tsit	Kepala medis rhodes	Medic	Medic	Arts
5	143	Cantabile	Seorang assassin bolivar	Vanguard	Agent	Physical
6	22	Virtuosa	Kriminal laterano	Supporter	Ritualist	Arts
7	20	Qiubai	Petualang dari Yan	Guard	Lord	Physical
8	117	Weedy	Operator original	Specialist	Push Stroker	Arts
9	445	Mostima	Seorang legatus dari lateran	Caster	Splash	Arts

Tekan enter untuk kembali: █

Gambar 4.7 Tampilan Tabel data

Data Operator Tersimpan (Sudah Diurutkan Berdasarkan Kelas - Selection Sort):

No	Kode	Nama	Deskripsi	Kelas	Branch	Tipe
1	215	Eyjafjalla	Nama asli Adele Nauman	Caster	Core caster	Arts
2	182	Nymph	Seorang warga kazdel	Caster	Primal caster	Arts
3	445	Mostima	Seorang legatus dari lateran	Caster	Splash	Arts
4	20	Qiubai	Petualang dari Yan	Guard	Lord	Physical
5	3	Kal'tsit	Kepala medis rhodes	Medic	Medic	Arts
6	117	Weedy	Operator original	Specialist	Push Stroker	Arts
7	22	Virtuosa	Kriminal laterano	Supporter	Ritualist	Arts
8	143	Cantabile	Seorang assassin bolivar	Vanguard	Agent	Physical
9	2	Ines	Ines adalah tentara bayaran	Vanguard	Agent	Physical

Tekan enter untuk kembali:

Gambar 4.8 Tampilan Tabel selection sort

Data Operator Tersimpan (Sudah Diurutkan Berdasarkan Kode):

No	Kode	Nama	Deskripsi	Kelas	Branch	Tipe
1	445	Mostima	Seorang legatus dari lateran	Caster	Splash	Arts
2	215	Eyjafjalla	Nama asli Adele Nauman	Caster	Core caster	Arts
3	182	Nymph	Seorang warga kazdel	Caster	Primal caster	Arts
4	143	Cantabile	Seorang assassin bolivar	Vanguard	Agent	Physical
5	117	Weedy	Operator original	Specialist	Push Stroker	Arts
6	22	Virtuosa	Kriminal laterano	Supporter	Ritualist	Arts
7	20	Qiubai	Petualang dari Yan	Guard	Lord	Physical
8	3	Kal'tsit	Kepala medis rhodes	Medic	Medic	Arts
9	2	Ines	Ines adalah tentara bayaran	Vanguard	Agent	Physical

Tekan enter untuk kembali:

Gambar 4.9 Tampilan Tabel Bubble sort

Data Operator Tersimpan (Sudah Diurutkan Berdasarkan Tipe - Insertion Sort):

No	Kode	Nama	Deskripsi	Kelas	Branch	Tipe
1	215	Eyjafjalla	Nama asli Adele Nauman	Caster	Core caster	Arts
2	182	Nymph	Seorang warga kazdel	Caster	Primal caster	Arts
3	3	Kal'tsit	Kepala medis rhodes	Medic	Medic	Arts
4	22	Virtuosa	Kriminal laterano	Supporter	Ritualist	Arts
5	117	Weedy	Operator original	Specialist	Push Stroker	Arts
6	2	Ines	Ines adalah tentara bayaran	Vanguard	Agent	Physical
7	143	Cantabile	Seorang assassin bolivar	Vanguard	Agent	Physical
8	20	Qiubai	Petualang dari Yan	Guard	Lord	Physical

Tekan enter untuk kembali:

Gambar 5.0 Tampilan Tabel Insertion sort

```

=====
Daftar Operator yang Tersimpan:
=====
1. Eyjafjalla
2. Ines
3. Nymph
4. Kal'tsit
5. Cantabile
6. Virtuosa
7. Qiubai
8. Weedy
9. Mostima
=====
Tekan 0 jika ingin kembali ke menu utama
=====

Masukkan nomor operator yang ingin diupdate: █

```

```

=====
update kode operator: 67
Update nama operator menjadi: Fiammetta
Update deskripsi operator menjadi: Agen spesial dari laterano
Update class operator menjadi: Sniper
Update branch operator menjadi: Artilleryman
Update tipe serangan operator menjadi: Arts
Data operator berhasil di update silahkan lihat pada bagian lihat data
=====
Apakah doktah ingin update operator lagi? (y/n): █

```

Gambar 5.1 Menu update dan hasil update

```

=====
Daftar Operator yang Tersimpan:
=====
1. Eyjafjalla
2. Ines
3. Nymph
4. Kal'tsit
5. Cantabile
6. Virtuosa
7. Qiubai
8. Fiammetta
=====
Tekan 0 jika ingin kembali ke menu utama
=====

Masukkan nomor operator yang ingin dihapus: █

```



```
=====
Data operator berhasil dihapus
=====

Apakah doktah ingin menghapus lagi operator lagi? (y/n):
```

Gambar 5.2 Menu hapus dan hasil hapus

```
=====
Selamat tinggal doktah
=====
```

Gambar 5.3 Log out

5. Langkah-langkah Git

A. Git add

Perintah git add digunakan untuk menambahkan file apa saja sebelum dilakukannya commit

```
PS C:\Users\ACER\Documents\praktikum-apl\post-test\post-test-6> git add .  
PS C:\Users\ACER\Documents\praktikum-apl\post-test\post-test-6>
```

Gambar 5.2 Git add

B. Git commit

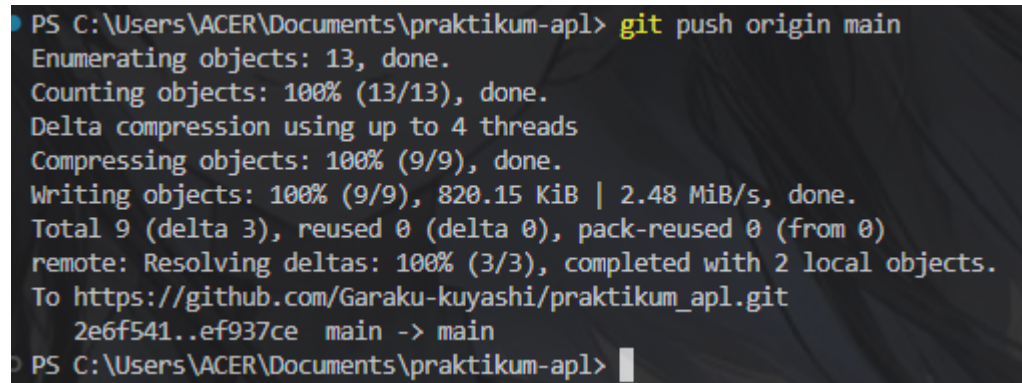
Perintah git commit digunakan untuk menyimpan perubahan yang telah di tambahkan ke dalam repository git

```
PS C:\Users\ACER\Documents\praktikum-apl\post-test\post-test-6> git commit -m 'finish posttest 6'  
[main cbee7c5] finish posttest 6  
3 files changed, 515 insertions(+)  
create mode 100644 post-test-6/2409106089-M.Fahrianor-PT-6  
create mode 100644 post-test-6/2409106089-M.Fahrianor-PT-6.cpp  
create mode 100644 post-test-6/2409106089-M.Fahrianor-PT-6.exe  
PS C:\Users\ACER\Documents\praktikum-apl\post-test\post-test-6>
```

Gambar 5.3 Git commit

C. Git push

perintah git push digunakan untuk mengunggah perubahan dari repository lokal ke github.



```
PS C:\Users\ACER\Documents\praktikum-apl> git push origin main
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 820.15 KiB | 2.48 MiB/s, done.
Total 9 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To https://github.com/Garaku-kuyashi/praktikum_apl.git
   2e6f541..ef937ce  main -> main
PS C:\Users\ACER\Documents\praktikum-apl>
```

Gambar 5.5 Git push