LAPORAN PRAKTIKUM

POSTTEST 2

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Elfin Sinaga (2409106024)

Kelas (A2 ‘24)

PROGRAM STUDI INFORMATIKA

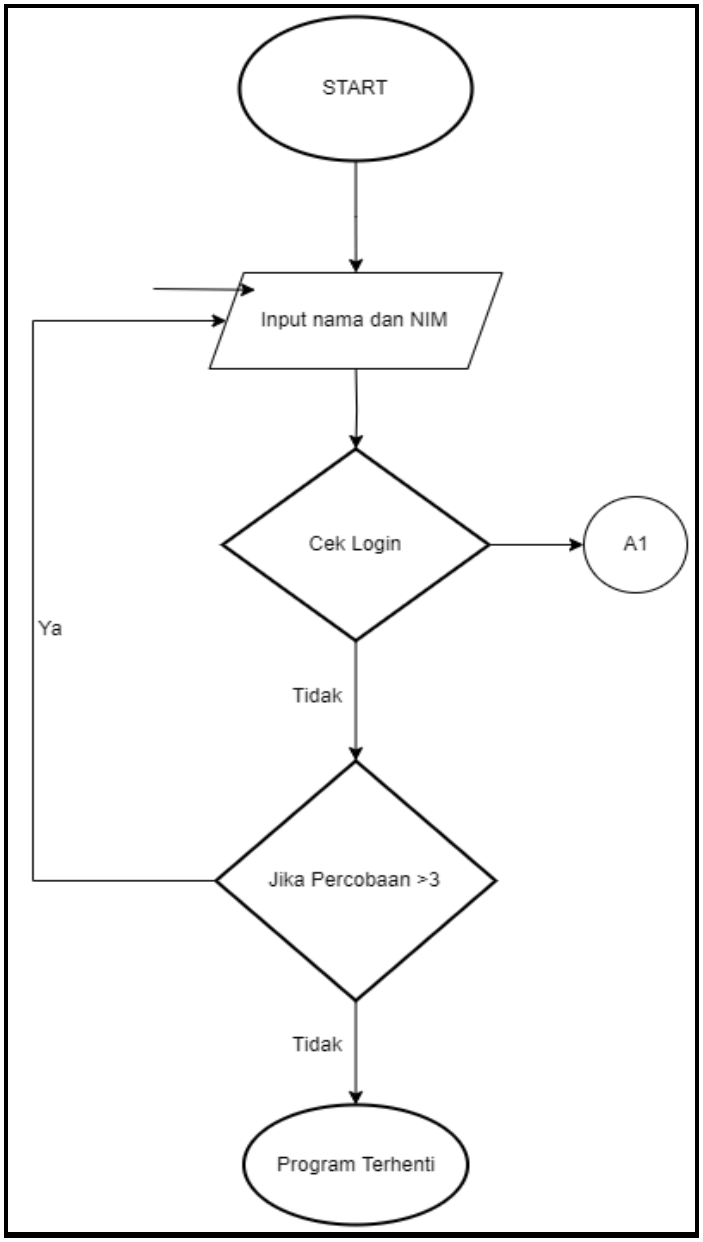
UNIVERSITAS MULAWARMAN

SAMARINDA

2025

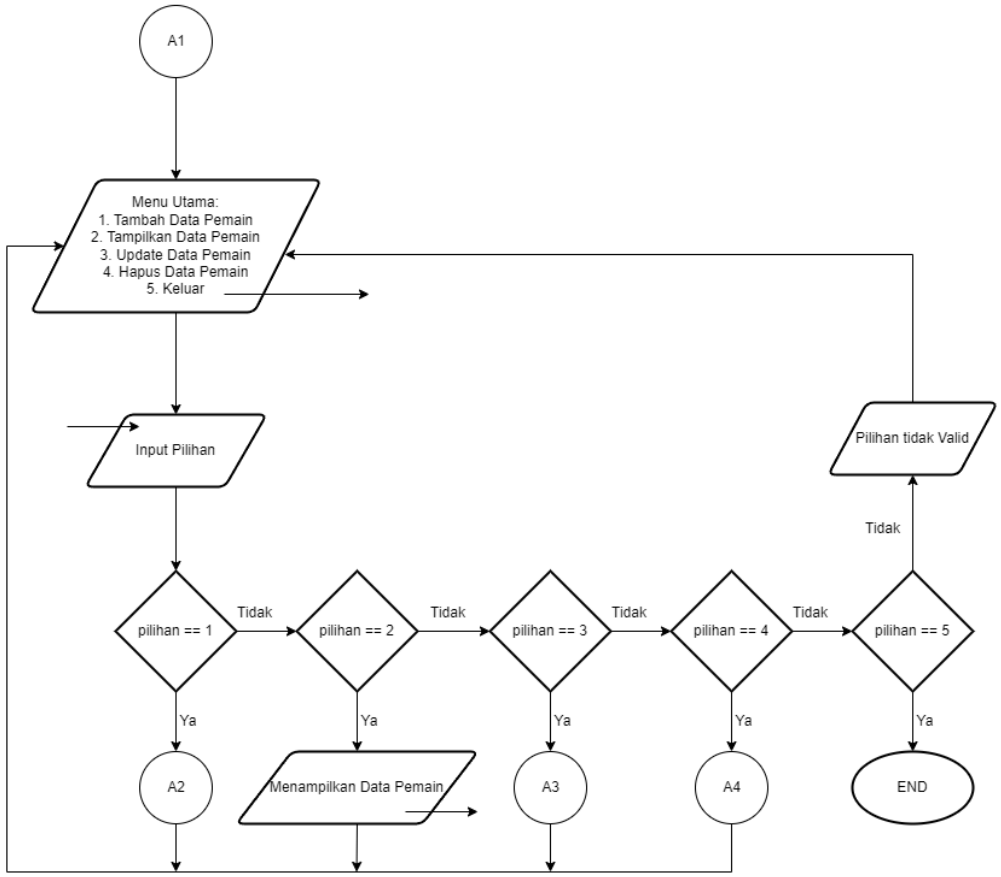
1. Flowchart

1.1. Proses Login



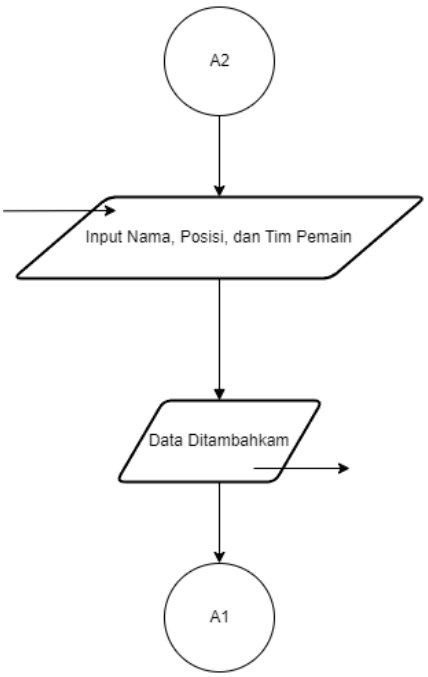
Gambar 1.1

1.2. Proses Menu



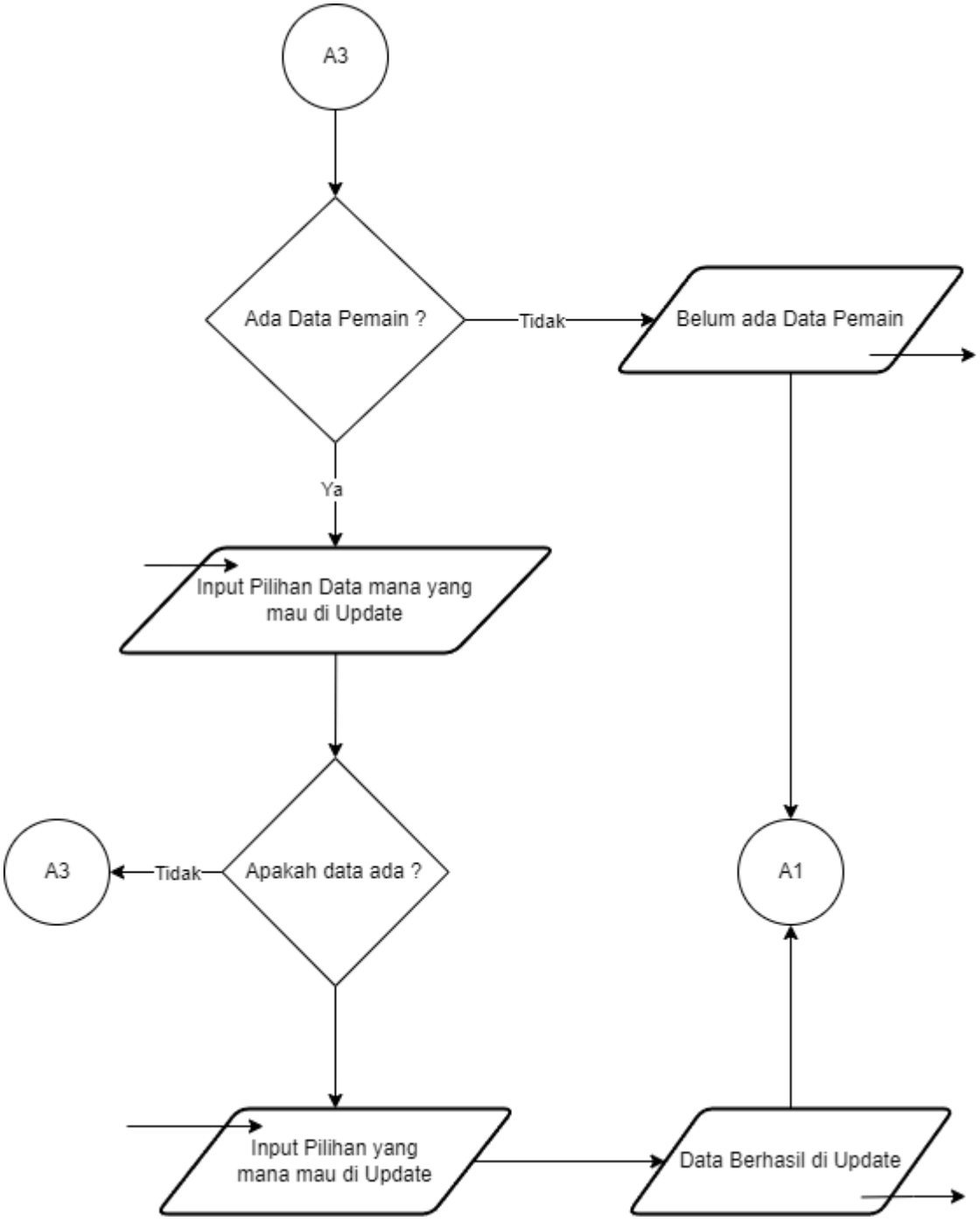
Gambar 1.2

1.3. Pilihan 1



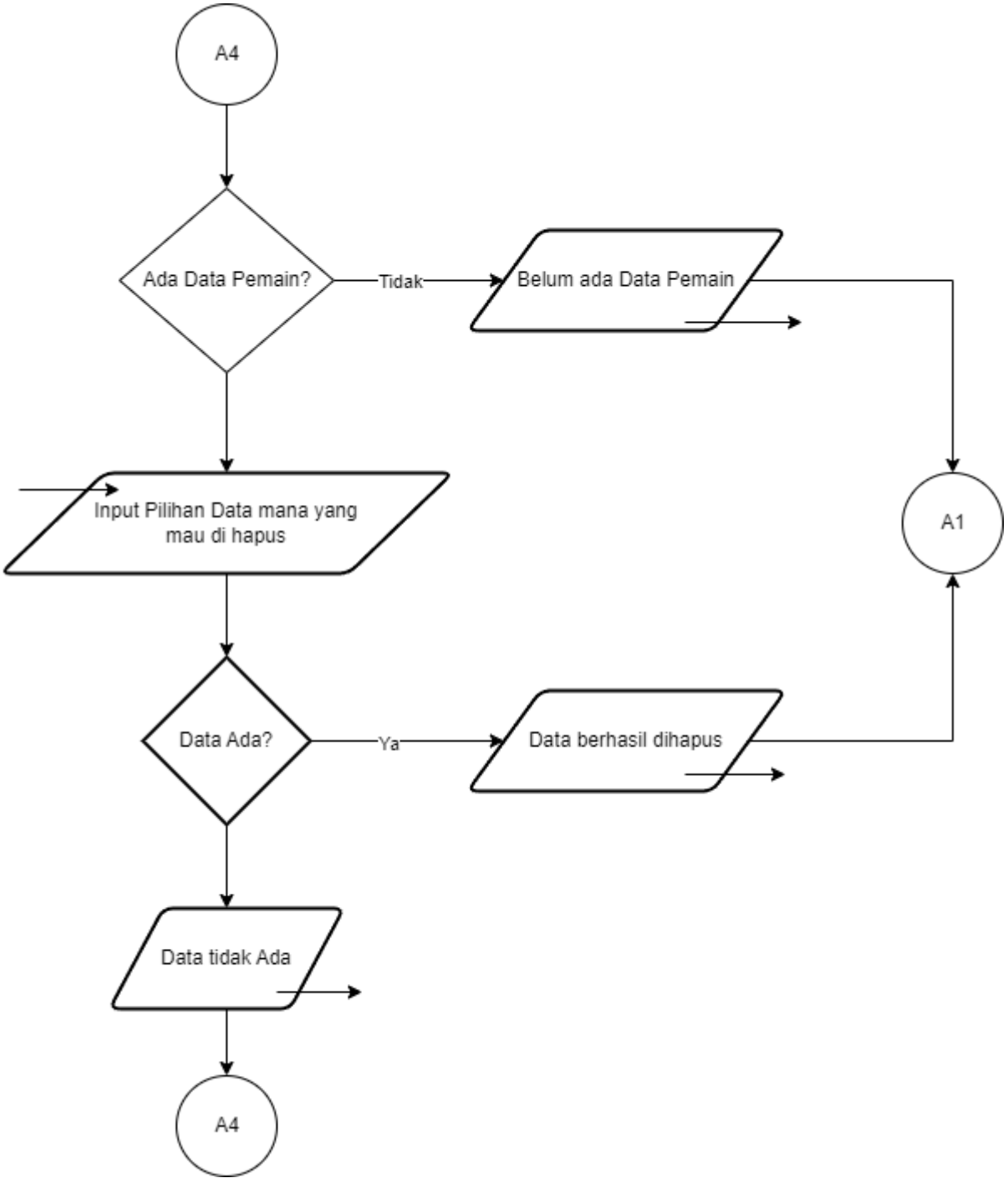
Gambar 1.3

1.4. Pilihan 2



Gambar 1.4

1.5 Pilihan 4



Gambar 1.5

2. Analisis Program

2.1. Deskripsi Singkat Program

Program ini adalah aplikasi manajemen data pemain NBA yang dikembangkan menggunakan C++. Pengguna dapat login dengan nama ("elfin") dan NIM ("024"), dengan batasan 3 kali percobaan. Jika gagal, program akan berhenti. Setelah login berhasil, pengguna dapat mengelola data pemain melalui operasi CRUD: menambah data pemain baru (nama, posisi, tim), menampilkan seluruh data yang tersimpan, mengubah data pemain yang ada, atau menghapus data berdasarkan nomor urut. Data disimpan dalam variabel array satu dimensi dan multidimensi. Program menampilkan menu interaktif yang memungkinkan pengguna memilih operasi hingga memutuskan untuk keluar.

2.2. Penjelasan Alur & Algoritma

Alur Program

1. Login:
    - Program meminta pengguna memasukkan nama dan NIM.
    - Jika nama dan NIM sesuai ("elfin" dan "024"), login berhasil.
    - Jika salah, pengguna diberi 3 kesempatan. Jika gagal setelah 3 kali, program berhenti.

2. Menu Utama:
    - Setelah login berhasil, program menampilkan menu:
    - Tambah Data Pemain: Menambahkan data pemain baru (nama, posisi, tim).
    - Tampilkan Data Pemain: Menampilkan semua data pemain yang tersimpan.
    - Update Data Pemain: Mengubah data pemain yang sudah ada.
    - Hapus Data Pemain: Menghapus data pemain berdasarkan nomor urut.
    - Keluar: Mengakhiri program.

3. Operasi CRUD:

- Tambah Data:
    - Program meminta input nama, posisi, dan tim pemain.
    - Data disimpan dalam array 2D (dataPemain).

- Tampilkan Data:
    - Program menampilkan semua data pemain yang tersimpan dalam array.

- Update Data:
    - Program meminta nomor pemain yang ingin diupdate.
    - Pengguna memilih data yang ingin diubah (nama, posisi, atau tim).
    - Data diperbarui di dalam array.

- Hapus Data:
    - Program meminta nomor pemain yang ingin dihapus.
    - Data dihapus dari array, dan data setelahnya digeser ke depan.

- Keluar:
    - Jika pengguna memilih opsi Keluar, program berhenti.

Algoritma Program

1. Login:

- Input: Nama dan NIM.
- Proses:
    - Bandingkan input dengan nama dan NIM yang benar.
    - Jika benar, lanjut ke menu utama.
    - Jika salah, tambah percobaan login.
    - Jika percobaan mencapai 3 kali, hentikan program.
    - Output: Status login (berhasil/gagal).

2. Menu Utama:

- Tampilkan pilihan menu (1-5).
- Input: Pilihan pengguna.
- Proses:
    - Jalankan operasi sesuai pilihan pengguna.
- Output: Hasil operasi (tambah, tampilkan, update, hapus, atau keluar).

3. Tambah Data:

- Input: Nama, posisi, dan tim pemain.
- Proses:
    - Simpan data ke dalam array dataPemain.
    - Tambah jumlah pemain (jumlahPemain).
- Output: Konfirmasi data berhasil ditambahkan.

4. Tampilkan Data:

- Proses:
    - Loop melalui array dataPemain dan tampilkan data pemain.
    - Output: Daftar pemain beserta nama, posisi, dan tim.

5. Update Data:

- Input: Nomor pemain dan data yang ingin diupdate (nama, posisi, atau tim).
- Proses:
    - Perbarui data pemain di array dataPemain.
- Output: Konfirmasi data berhasil diupdate.

6. Hapus Data:

- Input: Nomor pemain yang ingin dihapus.
- Proses:
    - Geser data setelahnya ke depan untuk mengisi kekosongan.
    - Kurangi jumlah pemain (jumlahPemain).

- Output: Konfirmasi data berhasil dihapus.

7. Keluar:
- Proses:
    - Hentikan program.
- Output: Pesan terima kasih.

3. Source Code

A. Fitur Login

Fitur ini Memastikan pengguna memasukkan Nama dan NIM terkahir yang benar sebelum mengakses Menu Manajemen Pemain NBA.

Source Code :

```
    while (percobaan < MAX_LOGIN_ATTEMPTS) {
        cout << "Masukkan Nama: ";
        cin >> nama;
        cout << "Masukkan NIM: ";
        cin >> nim;

        if (nama == namaBenar && nim == nimBenar) {
            cout << "Login berhasil!\n";
            loginBerhasil = true;
            break;
        } else {
            percobaan++;
            cout << "Login gagal. Percobaan " << percobaan << " dari " <<
MAX_LOGIN_ATTEMPTS << ".\n";
        }
    }

    if (!loginBerhasil) {
        cout << "Anda telah melebihi batas percobaan login. Program
berhenti.\n";
        return 0;
    }
```

Fitur:

- User diberi 3 Kali percobaan untuk memasukkan Nama dan NIM yang benar.
- Jika gagal setelah 3 kali, maka program akan berhenti.

B. Fitur Menu Manajemen Pemain NBA

Fitur ini akan memunculkan Menu yang bisa dipilih dan memudahkan pengguna untuk mengaksesnya.

Source Code :

```
    do {
        cout << "\n=== Menu Manajemen Data Pemain NBA ===\n";
        cout << "=====================================\n";
        cout << "1. Tambah Data Pemain\n";
        cout << "2. Tampilkan Data Pemain\n";
        cout << "3. Update Data Pemain\n";
        cout << "4. Hapus Data Pemain\n";
        cout << "5. Keluar\n";
        cout << "=====================================\n";
        cout << "Pilih menu: ";
        cin >> pilihan;
```

Fitur:

- Pengguna bisa memilih Menu dari 1 Sampai 5.
- Jika pengguna salah menginput angka yang tersedia pada Menu, maka program akan mengulang program kembali ke menu dan menginput ulang.

C. CREATE

Fitur ini bisa menambahkan data Pemain NBA

Source Code :

```
            case 1:
                if (jumlahPemain < MAX_PEMAIN) {
                    cout << "\n=== Tambah Data Pemain ===\n";
                    cin.ignore();
                    cout << "Masukkan Nama Pemain: ";
                    getline(cin, dataPemain[jumlahPemain][0]);
                    cout << "Masukkan Posisi Pemain: ";
                    getline(cin, dataPemain[jumlahPemain][1]);
                    cout << "Masukkan Tim Pemain: ";
                    getline(cin, dataPemain[jumlahPemain][2]);
                    jumlahPemain++;
                    cout << "Data pemain berhasil ditambahkan.\n";
                } else {
                        cout << "Kapasitas pemain penuh. Tidak bisa menambah data
lagi.\n";
                }
                break;
```

Fitur:

- Jika melebihi kapasitas penyimpanan data yaitu sebesar 100 data, maka akan memunculkan output “Kapasitas pemain penuh”.

D. READ

Fitur ini bisa Menampilkan data apa saja yang telah ditambahkan.

Source Code :

```
            case 2:
                if (jumlahPemain == 0) {
                    cout << "\nBelum ada data pemain.\n";
                } else {
                    cout << "\n=== Data Pemain NBA ===\n";
                    cout << "--------------------------------------\n";
                    for (int i = 0; i < jumlahPemain; i++) {
                        cout << "Pemain ke-" << i + 1 << ":\n";
                        cout << "Nama: " << dataPemain[i][0] << "\n";
                        cout << "Posisi: " << dataPemain[i][1] << "\n";
                        cout << "Tim: " << dataPemain[i][2] << "\n";
                        cout << "--------------------------------------\n";
                    }
                }
                break;
```

Fitur:

- Jika data tidak ada atau belum ada ditambahkan, maka mengeluarkan Output "Belum ada data pemain"

E. UPDATE

Fitur ini dapat memperbarui data.

Source Code :

```
            case 3:
                if (jumlahPemain == 0) {
                    cout << "\nBelum ada data pemain untuk diupdate.\n";
                } else {
                    cout << "\n=== Update Data Pemain ===\n";
                    int index;
                       cout << "Masukkan nomor pemain yang ingin diupdate (1-" <<
jumlahPemain << "): ";
                    cin >> index;

                    if (index > 0 && index <= jumlahPemain) {
                        int pilihanUpdate;
                        cout << "\nPilih data yang ingin diupdate:\n";
                        cout << "1. Nama\n";
                        cout << "2. Posisi\n";
                        cout << "3. Tim\n";
                        cout << "Pilihan: ";
                        cin >> pilihanUpdate;
```

```
                    cin.ignore();
                    switch (pilihanUpdate) {
                        case 1:
                            cout << "Masukkan Nama Pemain Baru: ";
                            getline(cin, dataPemain[index - 1][0]);
                            cout << "Nama pemain berhasil diupdate.\n";
                            break;
                        case 2:
                            cout << "Masukkan Posisi Pemain Baru: ";
                            getline(cin, dataPemain[index - 1][1]);
                            cout << "Posisi pemain berhasil diupdate.\n";
                            break;
                        case 3:
                            cout << "Masukkan Tim Pemain Baru: ";
                            getline(cin, dataPemain[index - 1][2]);
                            cout << "Tim pemain berhasil diupdate.\n";
                            break;
                        default:
                            cout << "Pilihan tidak valid.\n";
                            break;
                    }
                } else {
                    cout << "Nomor pemain tidak valid.\n";
                }
            }
            break;
```

Fitur:

- Pengguna dapat menginput pilihan dari 1 sampai 3 untuk data mana yang mau diperbarui.
- Jika pengguna tidak menginput pilihan yang ada, maka mengeluarkan output “Nomor Pemain tidak valid”

F. DELETE

Fitur ini dapat menghapus data yang ada.

Source Code :

```
        case 4:
            if (jumlahPemain == 0) {
                cout << "\nBelum ada data pemain untuk dihapus.\n";
            } else {
                cout << "\n=== Hapus Data Pemain ===\n";
                int index;
```

```
                    cout << "Masukkan nomor pemain yang ingin dihapus (1-" <<
jumlahPemain << "): ";
                    cin >> index;

                    if (index > 0 && index <= jumlahPemain) {
                        for (int i = index - 1; i < jumlahPemain - 1; i++) {
                            dataPemain[i][0] = dataPemain[i + 1][0];
                            dataPemain[i][1] = dataPemain[i + 1][1];
                            dataPemain[i][2] = dataPemain[i + 1][2];
                        }
                        jumlahPemain--;
                        cout << "Data pemain berhasil dihapus.\n";
                    } else {
                        cout << "Nomor pemain tidak valid.\n";
                    }
                }
                break;
```

Fitur:

- Jika data belum ada, maka data tidak dapat dihapus dan mengeluarkan Output “Belum ada data pemain untuk dihapus”
- Jika Pengguna salah menginput data mana yang ingin dihapus, maka mengeluarkan Output “Nomor pemain tidak valid”

4. Uji Coba dan Hasil Output

4.1 Uji Coba

Untuk menguji program saya berjalan, saya akan menggunakan Skenario yaitu:

1. Skenario 1: Salah menginput nama dan NIM sebanyak 3 kali lalu keluar dari program
2. Skenario 2: Berhasil Login tetapi salah menginput menu yang ada dan kembali ke menu
3. Skenario 3: Memilih menu tampilkan data, update data, dan hapus data lalu belum ada data yang diinput
4. Skenario 4: Mengulang operasi inputan dengan benar dan menampilkan cara kerja C.R.U.D.

4.2 Hasil Output

Skenario 1 :

```
Masukkan Nama: asd
Masukkan NIM: asd
Login gagal. Percobaan 1 dari 3.
Masukkan Nama: asd
Masukkan NIM: asd
Login gagal. Percobaan 2 dari 3.
Masukkan Nama: asd
Masukkan NIM: asd
Login gagal. Percobaan 3 dari 3.
Anda telah melebihi batas percobaan login. Program berhenti.
```

Gambar 2.1

Skenario 2 :

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 6

Pilihan tidak valid. Silakan coba lagi.
```

Gambar 2.2

Skenario 3 :

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 2

Belum ada data pemain.
```

Gambar 2.3

```
=== Menu Manajemen Data Pemain NBA ===
========================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
========================================
Pilih menu: 3

Belum ada data pemain untuk diupdate.
```

Gambar 2.4

```
=== Menu Manajemen Data Pemain NBA ===
========================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
========================================
Pilih menu: 4

Belum ada data pemain untuk dihapus.
```

Gambar 2.5

Skenario 4 :

1. Tambah Data Pemain (CREATE)

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 1

=== Tambah Data Pemain ===
Masukkan Nama Pemain: Luka Doncic
Masukkan Posisi Pemain: Small Guard
Masukkan Tim Pemain: Los Angles Lakers
Data pemain berhasil ditambahkan.
```

Gambar 2.6

2. Menampilkan Data Pemain (READ)

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 2

=== Data Pemain NBA ===
---------------------------------------
Pemain ke-1:
Nama: Luka Doncic
Posisi: Small Guard
Tim: Los Angles Lakers
---------------------------------------
Pemain ke-2:
Nama: Stephen Curry
Posisi: Small Guard
Tim: Golden State Warriors
---------------------------------------
```

Gambar 2.7

3. Memperbarui Data (UPDATE)

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 3

=== Update Data Pemain ===
Masukkan nomor pemain yang ingin diupdate (1-2): 2

Pilih data yang ingin diupdate:
1. Nama
2. Posisi
3. Tim
Pilihan: 3
Masukkan Tim Pemain Baru: Boston Celtics
Tim pemain berhasil diupdate.
```

Gambar 2.8

```
----------------------------------------
Pemain ke-2:
Nama: Stephen Curry
Posisi: Small Guard
Tim: Boston Celtics
----------------------------------------
```

Gambar 2.9

4. Menghapus Data (DELETE)

```
=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 4

=== Hapus Data Pemain ===
Masukkan nomor pemain yang ingin dihapus (1-2): 1
Data pemain berhasil dihapus.

=== Menu Manajemen Data Pemain NBA ===
=======================================
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Keluar
=======================================
Pilih menu: 2

=== Data Pemain NBA ===
---------------------------------------
Pemain ke-1:
Nama: Stephen Curry
Posisi: Small Guard
Tim: Boston Celtics
---------------------------------------
```

Gambar 2.10

5. **Langkah Langkah Git**

```
PS C:\Kuliah\praktikum-apl> git add .
PS C:\Kuliah\praktikum-apl> git commit -m "test"
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
PS C:\Kuliah\praktikum-apl> git push origin main
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 16 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 3.17 KiB | 1.58 MiB/s, done.
Total 14 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Symthi/praktikum-apl.git
   8b1abdb..5f1d327  main -> main
PS C:\Kuliah\praktikum-apl>
```

1. Menginput (git add .)
   Perintah ini digunakan untuk menambahkan semua perubahan yang ada di direktori kerja ke staging area. Ini berarti semua file yang telah dimodifikasi, dihapus, atau ditambahkan akan siap untuk di-commit.
2. Menginput (git commit -m “test”)
   Perintah ini membuat commit baru dengan pesan "test". Commit adalah cara untuk menyimpan perubahan yang telah dilakukan ke dalam repositori lokal.
3. Menginput (git push origin main)
   Perintah ini digunakan untuk mengirim commit yang ada di branch lokal main ke repositori remote (origin).