**LAPORAN PRAKTIKUM POSTTEST 2**

**ALGORITMA PEMROGRAMAN LANJUT**



# Disusun oleh:

**Rusdiana (2409106021)**

**Kelas A1’24**

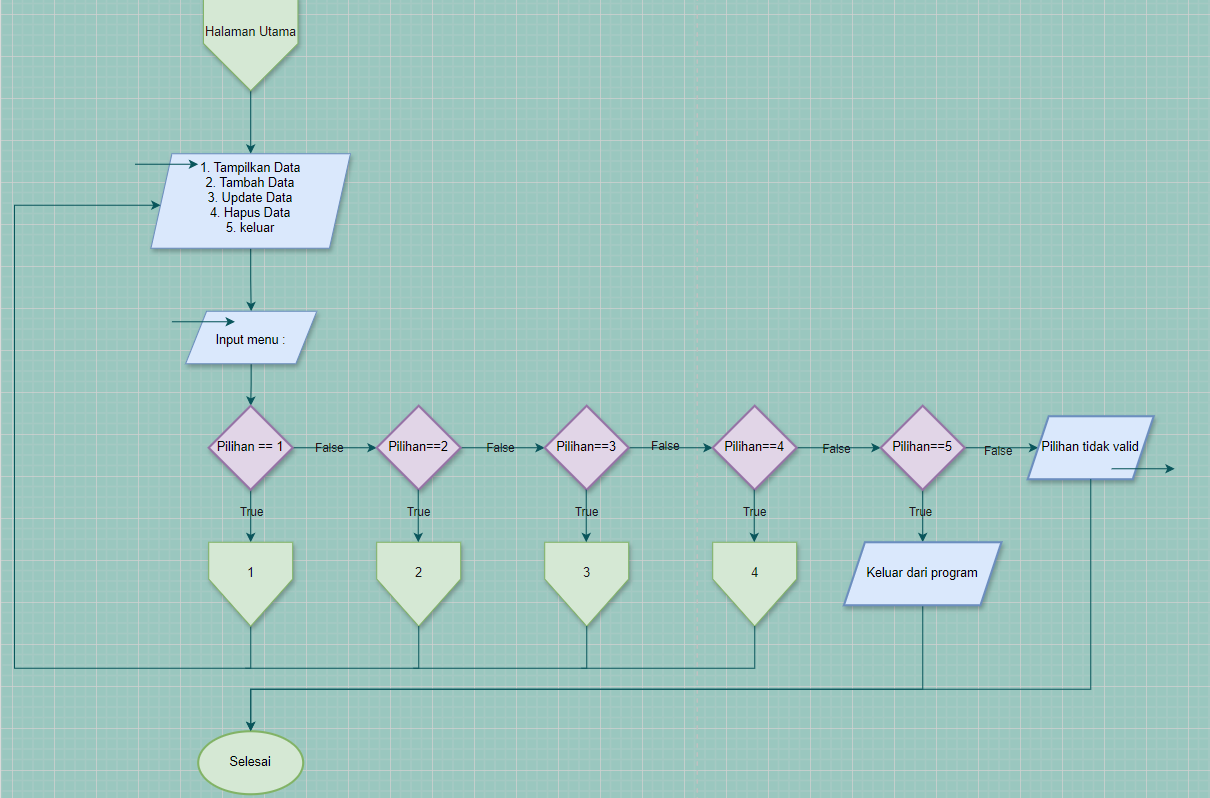
**PROGRAM STUDI INFORMATIKA UNIVERSITAS MULAWARMAN SAMARINDA**

**2025**

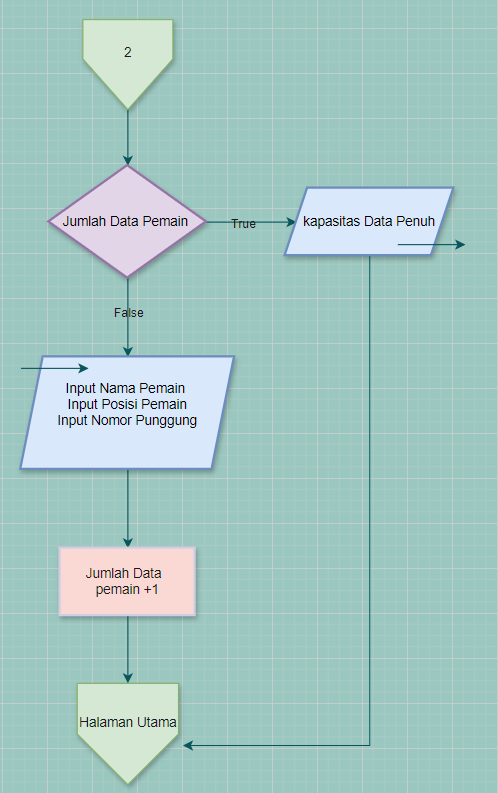
# Flowchart

# 

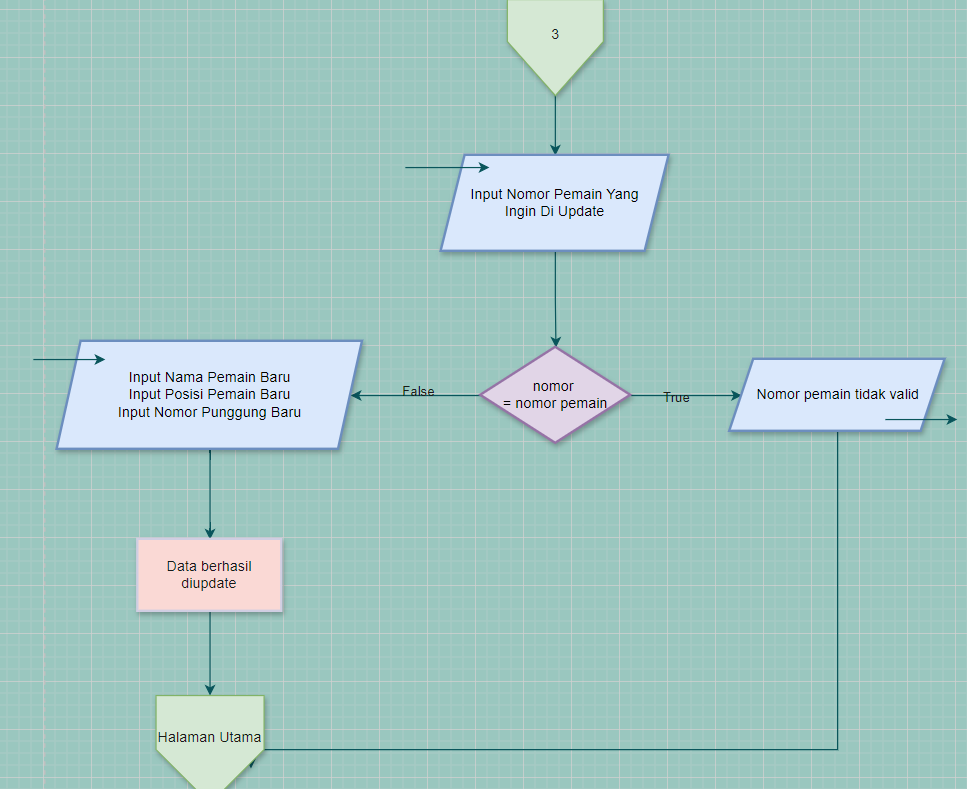
Gambar 1. 1 Login Gambar 1.3Menampilkan data



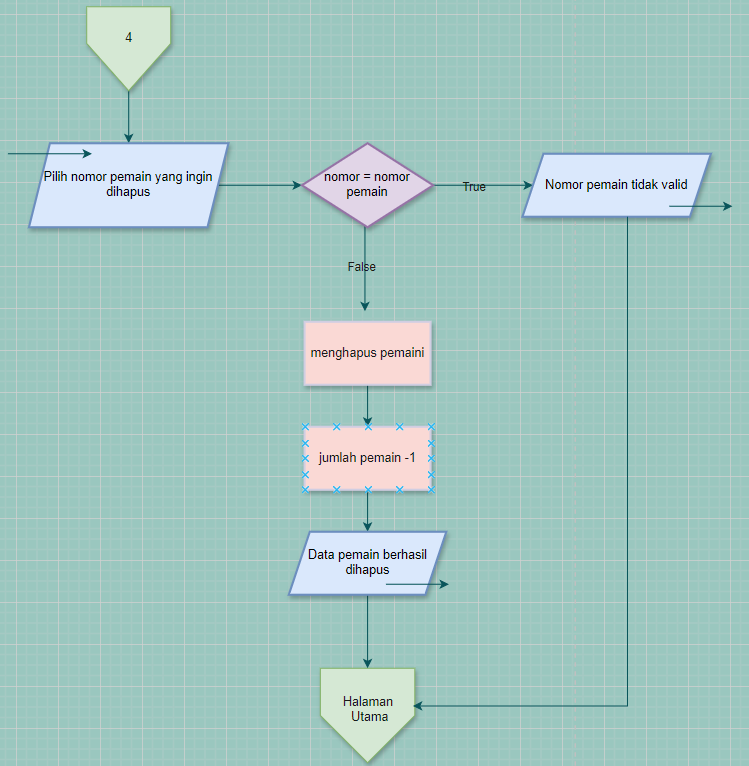
Gambar 1. 2 Halaman Utama



Gambar 1. 4 Penambahan Data



Gambar 1.5 Update Pemain



Gambar 1. 6 Penghapusan Data

# Analisis Program

* 1. **Deskripsi Singkat Program**

Program ini adalah sebuah manajemen “Data Pemain Bola” yang ditulis dalam bahasa C++. Program ini memungkinkan pengguna untuk melakukan berbagai operasi CRUD (Create, Read, Update, Delete) pada data pemain bola, seperti menambahkan, menampilkan, mengupdate, dan menghapus data pemain. Selain itu, program ini juga dilengkapi dengan sistem login untuk mengamankan akses ke aplikasi..

# Penjelasan Alur & Algoritma

Program ini terdiri dari beberapa tahap utama, yang mencakup proses login, navigasi menu, serta operasi CRUD (Create, Read, Update, Delete) pada data produk. Berikut adalah alur kerja program :

### Login Pengguna

* + - * Pengguna harus memasukkan username dan 3 digit terakhir NIM untuk masuk ke sistem.
      * Jika pengguna salah memasukkan data login sebanyak tiga kali, program akan berhenti.
      * Jika login berhasil, pengguna diarahkan ke menu utama.

### Menampilkan Menu Utama

* + - * Program menampilkan opsi yang bisa dipilih oleh pengguna:
        1. Tampilkan Data Pemain
        2. Tambah Data Pemain
        3. Update Data Pemain
        4. Hapus Data Pemain
        5. Keluar
      * Pengguna memasukkan nomor pilihan menu, yang akan menentukan langkah selanjutnya.

### CRUD Data Produk

* + - * **Tampilan Data Pemain**: Program mencetak daftar produk dalam format tabel agar lebih rapi dan mudah dibaca.
      * **Tambah Data Pemain**: Pengguna menginputkan nama pemain, posisi pemain, dan nomor punggung pemain, yang kemudian disimpan dalam array dua dimensi.
      * **Update Data Pemain**: Pengguna memilih Pemain berdasarkan nomor urutnya, lalu memasukkan data baru untuk mengganti informasi yang ada.
      * **Hapus Data Pemain** : Pengguna memilih nomor pemain yang ingin dihapus, lalu program menggeser data dalam array agar tetap terstruktur.

### Keluar dari Program

* + - * Jika pengguna memilih opsi Keluar, program akan menampilkan pesan perpisahan dan berhenti.

### Penjelasan Fungsi dalam Program:

* **TampilkanData( )** Menampilkan daftar pilihan menu utama.
* **TambahData( )** Menambahkan data pemain baru ke dalam array jika kapasitas belum penuh.
* **TampilkanData( )** Menampilkan semua produk yang telah disimpan dalam format tabel.
* **UbahData( )** Mengubah informasi data pemain tertentu berdasarkan input pengguna.
* **HapusData( )** Menghapus data pemain yang dipilih dan menyesuaikan susunan array.
* **Main( )** Mengontrol seluruh proses eksekusi program dari login hingga interaksi dengan menu.

# Source Code

1. **Login**

Pada Fitur ini menu yang digunakan untuk menvalidasi user yang ingin menggunakan aplikasi

### Source Code :

|  |
| --- |
| int main() {  *// Inisialisasi variabel array satu dimensi*  string nama\_pemain[100];  string posisi\_pemain[100];  int nomor\_punggung[100];  int jumlah\_data = 0;   const string username = "Rusdiana";  const string password = "2409106021";  const int max\_kesempatan = 3;  int kesempatan = 0;    while (kesempatan < max\_kesempatan) {  string input\_username, input\_password;  cout << "Username: ";  getline(cin, input\_username);  cout << "Password: ";  getline(cin, input\_password);    if (input\_username == username && input\_password == password) {  cout << "Login berhasil!\n" << endl;  break;  } else {  kesempatan++;  cout << "Login gagal. Sisa percobaan: " << max\_kesempatan - kesempatan << "\n" << endl;  }  }    if (kesempatan == max\_kesempatan) {  cout << "Anda telah mencoba login " << max\_kesempatan << " kali. Program berhenti." << endl;  return 0;  } |

1. **Menu**

Terdapat beberapa menu untuk memilih fitur pada aplikasi

### Source Code

|  |
| --- |
| *// Main program*  while (true) {  cout << "\n==== Manajemen Data Pemain Bola =====" << endl;  cout << "1. Tampilkan Data" << endl;  cout << "2. Tambah Data" << endl;  cout << "3. Update Data" << endl;  cout << "4. Hapus Data" << endl;  cout << "5. Keluar" << endl;  int pilihan;  cout << "Pilih menu (1-5): ";  cin >> pilihan; |

**C. CRUD Manajemen Data Pemain Bola**

Fitur CRUD dalam pemrograman ini mengacu pada empat operasi utama yaitu Create (membuat/menambahkan data baru), Read (Mengambil dan menampilkan data yang tersimpan), Update (Mengubah atau memperbarui data yang sudah ada.), Delet (Menghapus data dari Array.) yang digunakan untuk mengelola data produk dalam Array

**Source Code:**

|  |
| --- |
| if (pilihan == 1) {  *// Tampilkan Data*  cout << "\n========================= Data Pemain Bola =========================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(20) << "Nomor Punggung" << endl;  cout << string(66, '-') << endl;  for (int i = 0; i < jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << nama\_pemain[i] << setw(20) << posisi\_pemain[i] << setw(15) << nomor\_punggung[i] << endl;  }  cout << endl;  } else if (pilihan == 2) {  *// Tambah Data*  if (jumlah\_data < 100) {  cout << "Masukkan Nama Pemain: ";  cin.ignore();  getline(cin, nama\_pemain[jumlah\_data]);  cout << "Masukkan Posisi Pemain: ";  getline(cin, posisi\_pemain[jumlah\_data]);  cout << "Masukkan Nomor Punggung: ";  cin >> nomor\_punggung[jumlah\_data];  jumlah\_data++;  cout << "Data berhasil ditambahkan!\n";  } else {  cout << "Kapasitas data penuh!\n";  }  } else if (pilihan == 3) {  *// Update Data*  if (jumlah\_data > 0) {  cout << "\n===== Data Pemain Bola =====" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(15) << "Nomor Punggung" << endl;  cout << string(60, '-') << endl;  for (int i = 0; i < jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << nama\_pemain[i] << setw(20) << posisi\_pemain[i] << setw(15) << nomor\_punggung[i] << endl;  }  cout << endl;   int index;  cout << "Masukkan nomor pemain yang ingin diupdate: ";  cin >> index;  if (index >= 1 && index <= jumlah\_data) {  cout << "Masukkan Nama Pemain Baru: ";  cin >> nama\_pemain[index - 1];  cout << "Masukkan Posisi Pemain Baru: ";  cin >> posisi\_pemain[index - 1];  cout << "Masukkan Nomor Punggung Baru: ";  cin >> nomor\_punggung[index - 1];  cout << "Data berhasil diupdate!\n";  } else {  cout << "Nomor pemain tidak valid!\n";  }  } else {  cout << "Tidak ada data untuk diupdate!\n";  }  } else if (pilihan == 4) {  *// Hapus Data*  if (jumlah\_data > 0) {  cout << "\n=== Data Pemain Bola ===" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(15) << "Nomor Punggung" << endl;  cout << string(60, '-') << endl;  for (int i = 0; i < jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << nama\_pemain[i] << setw(20) << posisi\_pemain[i] << setw(15) << nomor\_punggung[i] << endl;  }  cout << endl;   int index;  cout << "Masukkan nomor pemain yang ingin dihapus: ";  cin >> index;  if (index >= 1 && index <= jumlah\_data) {  for (int i = index - 1; i < jumlah\_data - 1; i++) {  nama\_pemain[i] = nama\_pemain[i + 1];  posisi\_pemain[i] = posisi\_pemain[i + 1];  nomor\_punggung[i] = nomor\_punggung[i + 1];  }  jumlah\_data--;  cout << "Data berhasil dihapus!\n";  } else {  cout << "Nomor pemain tidak valid!\n";  }  } else {  cout << "Tidak ada data untuk dihapus!\n";  } |

**4. Uji Coba dan Hasil Output**

**4.1 Uji Coba**

Untuk memastikan bahwa program berjalan dengan baik dilakukan beberapa skenario uji coba dengan berbagai jenis input. Berikut adalah dua skenario utama yang diuji:

1. Skenario 1: Login dengan Input yang Salah

* **Langkah-langkah:**

1. Menjalankan program.
2. Memasukkan username atau NIM yang salah sebanyak tiga kali.
3. Program menampilkan pesan error dan berhenti setelah tiga kali percobaan gagal.

* Hasil yang diharapkan:
* Jika pengguna salah memasukkan login tiga kali, program akan berhenti secara otomatis.
* Jika pengguna memasukkan data yang benar sebelum tiga kali percobaan, mereka dapat mengakses menu utama.

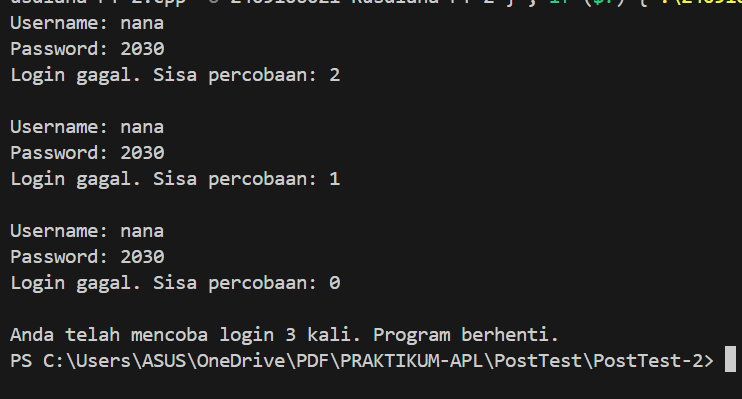
2. Skenario 2: Menambahkan dan Menampilkan Produk

* Langkah-langkah:

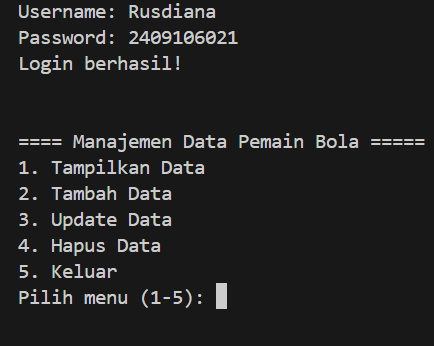
1. Login dengan username dan NIM yang benar.
2. Memilih menu "Tampilkan Data" dan memasukkan data produk (nama, stok, harga).
3. Memilih menu "Tambah Data" untuk memastikan data yang telah dimasukkan muncul dengan format tabel yang rapi.

* Hasil yang diharapkan:
* Data yang baru ditambahkan muncul dalam daftar dengan format tabel.
* Data pemain tersimpan dengan benar dalam array.

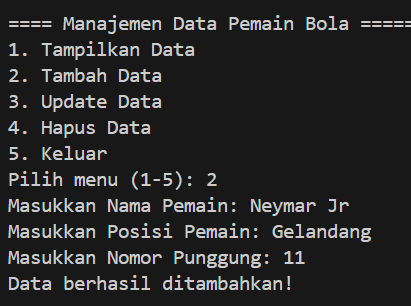
**4.2 Hasil Output**



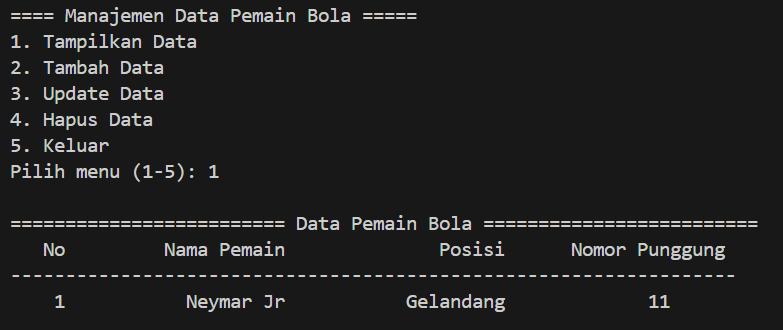
Gambar 4. 2 Gagal Login



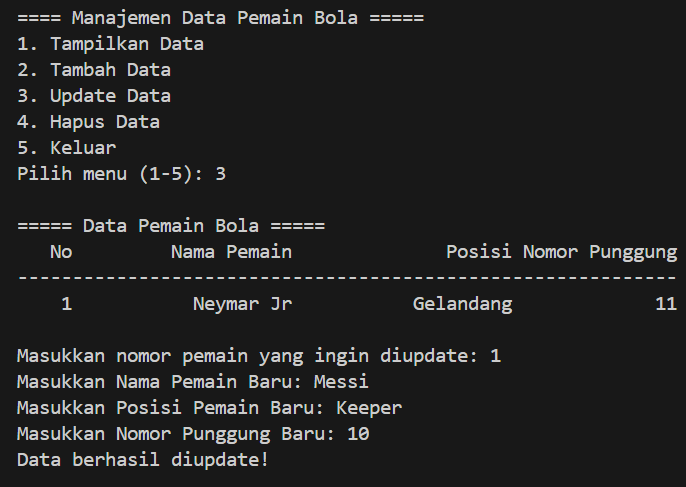
Gambar 4. 1 Berhasil Login



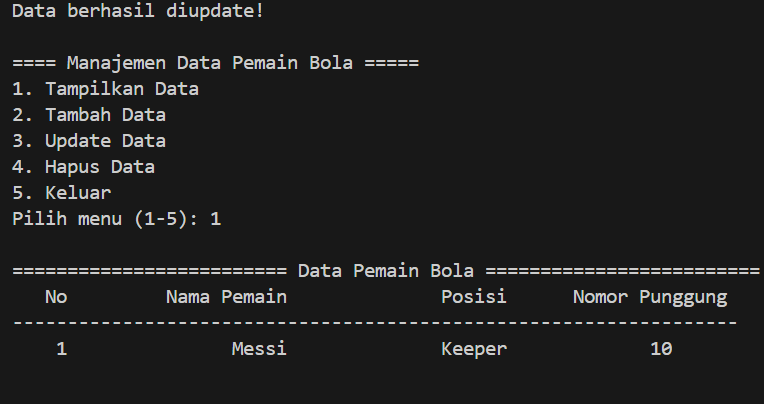
Gambar 4. 4 Tambah Data



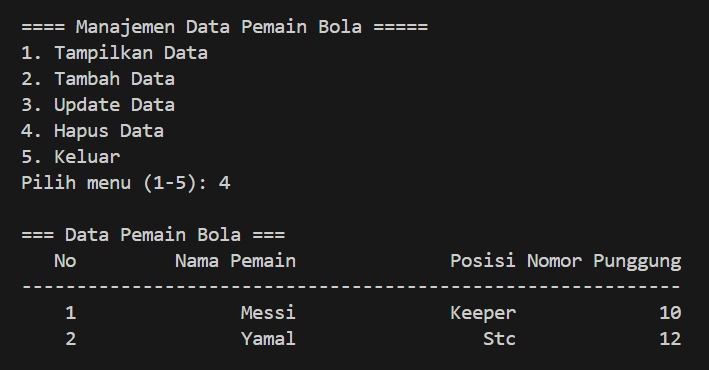
Gambar 4. 3 Tampilkan Data Pemain



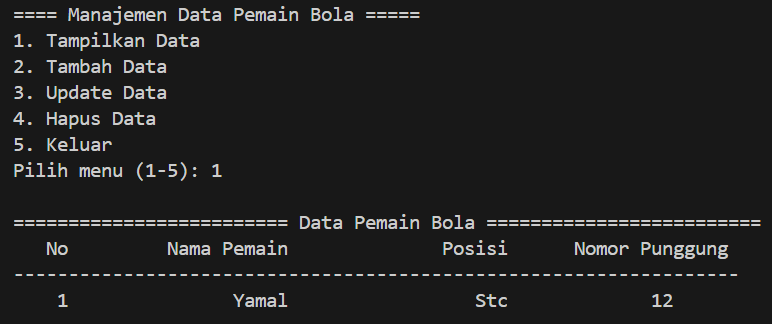
Gambar 4. 6 Update Data Pemain



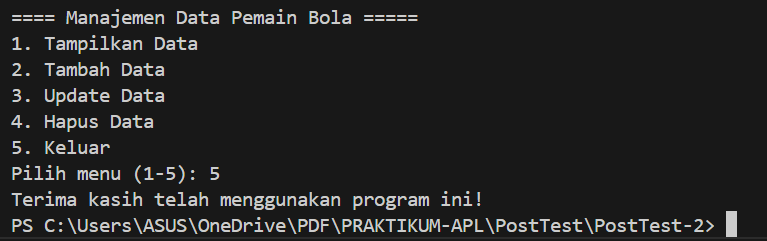
Gambar 4. 5 daftar setelah diubah



Gambar 4. 8 Hapus Data Pemain



Gambar 4. 7 Daftar Setelah Dihapus

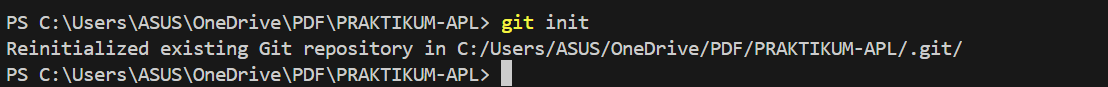


Gambar 4. 9 Keluar Dari Program

**5. Langkah-Langkah GIT**

1. **Git Init**

## Git init adalah perintah Git yang digunakan untuk menginisialisasi repository Git baru dalam sebuah folder. Cukup ketik “git init” pada terminal vscode



Gambar 5. 1 git\_init

# Git add

Git add adalah perintah Git yang digunakan untuk menambahkan perubahan pada file ke staging area sebelum dilakukan commit. Perintah ini memungkinkan Git mengetahui file mana saja yang akan dimasukkan dalam commit berikutnya