**LAPORAN PRAKTIKUM**

**POSTTEST 6**

**ALGORITMA PEMROGRAMAN LANJUT**



**Disusun oleh:**

**Devon Falen Pasae**

**(2409106055)**

# Kelas (B1‘24)

**PROGRAM STUDI INFORMATIKA**

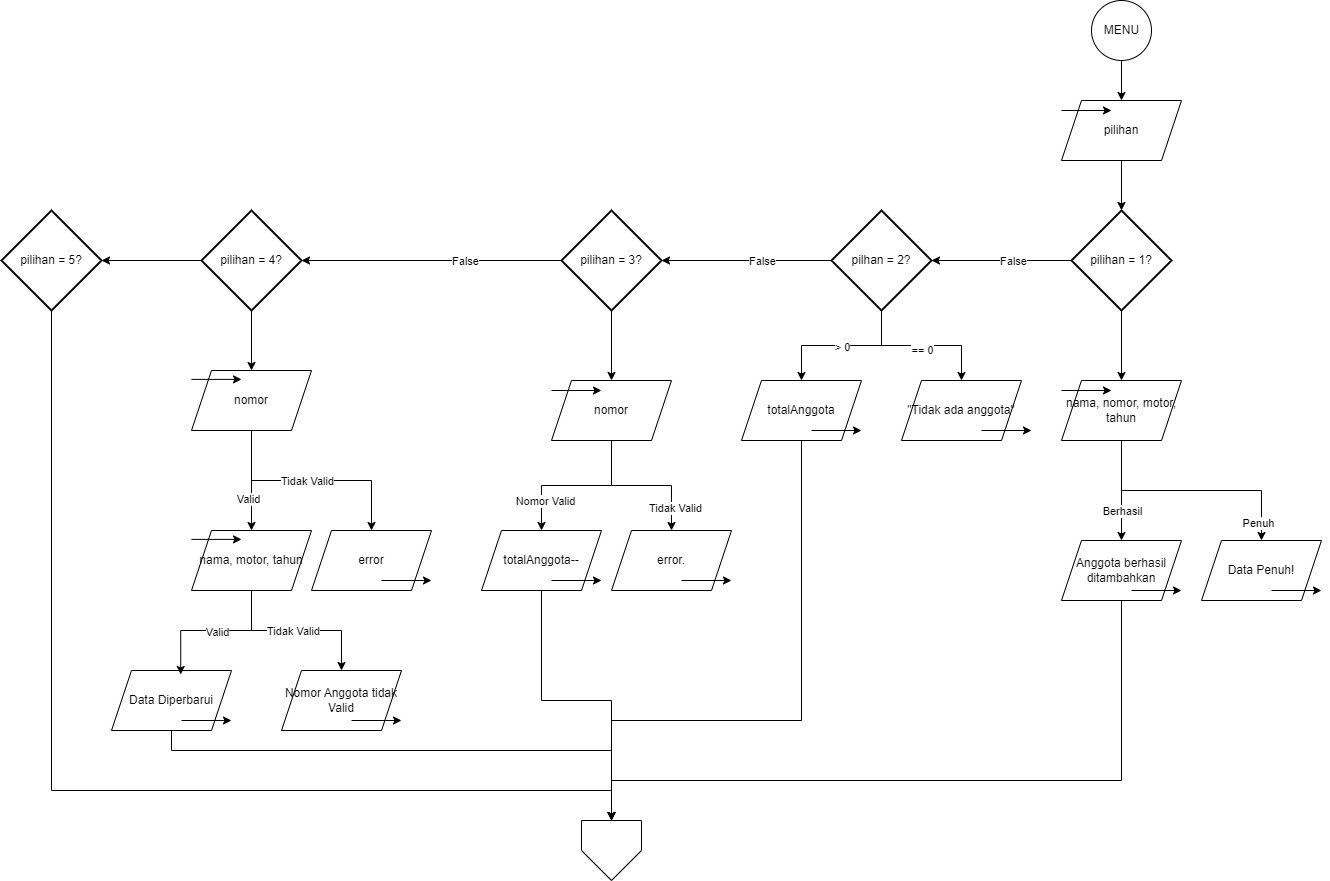
**UNIVERSITAS MULAWARMAN**

**SAMARINDA**

**2025**

# Flowchart





# Analisis Program

## Deskripsi Singkat Program

Program "Manajemen Data Anggota Klub Motor" yangberbasis C++ yang digunakan untuk mengelola data anggota dalam sebuah klub motor. Program ini memungkinkan pengguna untuk melakukan operasi CRUD (Create, Read, Update, Delete) terhadap data anggota, yang mencakup informasi Nama, Nomor Anggota, Jenis Motor, dan Tahun Bergabung. Dan juga, telah ditambahkan metode sorting di dalam program. Terdapat 3 metode sorting, yaitu Bubble Sort, Quick Sort, dan Merge Sort.

# Source Code

* **Autentikasi**

Fitur ini dibuat sebagai bentuk autentikasi program dengan cara pengguna input password yang benar dan diberi kesempatan 3 kali. Jika pengguna gagal menginput password benar, maka program akan otomatis berakhir.

|  |
| --- |
| *// Fungsi autentikasi* int autentikasi() {  string username\_benar = "Devon";  string nim\_benar = "6055";  int max\_percobaan = 3;  bool beri\_akses = false;   for (int percobaan = 1; percobaan <= max\_percobaan; percobaan++) {  string input\_username, input\_nim;  cout << "Masukkan Nama: ";  getline(cin >> std::ws, input\_username);  cout << "Masukkan NIM: ";  cin >> input\_nim;   if (input\_username == username\_benar && input\_nim == nim\_benar) {  beri\_akses = true;  cout << "== Autentikasi berhasil! ==\n";  break;  } else {  cout << "Nama atau NIM salah! Sisa percobaan: " << (max\_percobaan - percobaan) << endl;  }  }   if (!beri\_akses) {  cout << "== Autentikasi gagal. Program berhenti. ==\n";  return 0;  }  return 1; } |

=

=

* **Menu Utama**  =

Fitur ini berfung=si sebagai tampilan menu. Pengguna dapat memilih berbagai menu yang tersedia.

|  |
| --- |
| int main() {  if (autentikasi() == 0) return 0;   Anggota anggota[MAX\_ANGGOTA];  int totalAnggota = 0;  int pilihan;   do {  cout << "\nMENU:" << endl;  cout << "1. Tambah Anggota" << endl;  cout << "2. Tampilkan Anggota" << endl;  cout << "3. Hapus Anggota" << endl;  cout << "4. Edit Anggota" << endl;  cout << "5. Urutkan Data Anggota" << endl;  cout << "6. Keluar" << endl;  cout << "Pilih: ";  cin >> pilihan; |

* **Tambah Anggota**

Fitur ini berperan sebagai CREATE, yang bertugas untuk menambahkan data baru berupa informasi yang berkaitan dengan anggota klub motor.

*// Fungsi penambahan anggota*

|  |
| --- |
| void tambahAnggota(Anggota \*data, int \*total) {  if (\*total < MAX\_ANGGOTA) {  cout << "Masukkan Nama: ";  getline(cin >> std::ws, (data + \*total)->nama);  cout << "Masukkan Nomor Anggota: ";  cin >> (data + \*total)->nomor;  cin.ignore(numeric\_limits<streamsize>::max(), '\n');  cout << "Masukkan Jenis Motor: ";  getline(cin, (data + \*total)->motor);  cout << "Masukkan Tahun Bergabung: ";  cin >> (data + \*total)->tahun;  cin.ignore(numeric\_limits<streamsize>::max(), '\n');  (\*total)++;  cout << "Anggota berhasil ditambahkan!" << endl;  } else {  cout << "Data penuh!" << endl;  } } |

* **Tampilkan Anggota**

Fitur ini berperan sebagai READ, dimana pengguna dapat melihat data yang telah dibuat sebelumnya dan memeriksa data tersebut.

|  |
| --- |
| *// Fungsi menampilkan anggota* void tampilkanAnggota(Anggota \*data, int \*total) {  if (\*total == 0) {  cout << "Tidak ada anggota." << endl;  } else {  cout << "\n----------------------------------------------------------------------" << endl;  cout << "| No | Nama | No. Anggota | Jenis Motor | Tahun |" << endl;  cout << "----------------------------------------------------------------------" << endl;  for (int i = 0; i < \*total; i++) {  cout << "| " << i + 1 << " | "  << (data + i)->nama << string(18 - (data + i)->nama.length(), ' ') << "| "  << (data + i)->nomor << " | "  << (data + i)->motor << string(17 - (data + i)->motor.length(), ' ') << "| "  << (data + i)->tahun << " |" << endl;  }  cout << "----------------------------------------------------------------------" << endl;  } } |

* **Hapus Anggota**

Fitur ini berperan sebagai DELETE, dimana pengguna dapat menghapus data yang sekiranya ingin dihapus dari daftar anggota. */ Fungsi menghapus data anggota*

|  |
| --- |
| void hapusAnggota(Anggota \*data, int \*total) {  if (\*total == 0) {  cout << "Tidak ada anggota untuk dihapus." << endl;  } else {  int index;  cout << "Masukkan nomor anggota yang dihapus: ";  cin >> index;  if (index > 0 && index <= \*total) {  for (int i = index - 1; i < \*total - 1; i++) {  \*(data + i) = \*(data + i + 1);  }  (\*total)--;  cout << "Anggota dihapus!" << endl;  } else {  cout << "Nomor anggota tidak valid!" << endl;  }  } } |

* **Edit Anggota**

Berfungsi sebagai EDIT, dimana pengguna dapat menyunting data anggota yang sebelumnya sudah terdapat dalam daftar.

|  |
| --- |
| void editAnggota(Anggota \*data, int \*total) {  if (\*total == 0) {  cout << "Tidak ada anggota untuk diedit." << endl;  } else {  int index;  cout << "Masukkan nomor anggota yang diedit: ";  cin >> index;  cin.ignore(numeric\_limits<streamsize>::max(), '\n');  if (index > 0 && index <= \*total) {  cout << "Masukkan Nama Baru: ";  getline(cin, (data + index - 1)->nama);  cout << "Masukkan Jenis Motor Baru: ";  getline(cin, (data + index - 1)->motor);  cout << "Masukkan Tahun Bergabung Baru: ";  cin >> (data + index - 1)->tahun;  cin.ignore(numeric\_limits<streamsize>::max(), '\n');  cout << "Data diperbarui!" << endl;  } else {  cout << "Nomor anggota tidak valid!" << endl;  }  } } |

* **Metode Sorting**

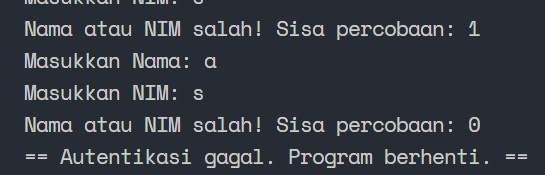
Berfungsi untuk mengurutkan data yang ditampilkan, dalam kasus ini menggunakan 3 metode, yaitu Quick Sort, Merge Sort, dan Bubble Sort.

|  |
| --- |
| *// Bubble Sort berdasarkan nama (A-Z)* void bubbleSortNamaAsc(Anggota data[], int total) {  for (int i = 0; i < total - 1; i++) {  for (int j = 0; j < total - i - 1; j++) {  if (data[j].nama > data[j + 1].nama) {  swap(data[j], data[j + 1]);  }  }  }  cout << "Data telah diurutkan berdasarkan nama (A-Z) menggunakan Bubble Sort.\n"; }  *// Merge Sort berdasarkan tahun (desc)* void merge(Anggota arr[], int l, int m, int r) {  Anggota temp[MAX\_ANGGOTA];  int i = l, j = m + 1, k = 0;  while (i <= m && j <= r) {  if (arr[i].tahun >= arr[j].tahun) {  temp[k++] = arr[i++];  } else {  temp[k++] = arr[j++];  }  }  while (i <= m) temp[k++] = arr[i++];  while (j <= r) temp[k++] = arr[j++];  for (int x = 0; x < k; x++) {  arr[l + x] = temp[x];  } }  void mergeSortTahunDesc(Anggota arr[], int l, int r) {  if (l < r) {  int m = (l + r) / 2;  mergeSortTahunDesc(arr, l, m);  mergeSortTahunDesc(arr, m + 1, r);  merge(arr, l, m, r);  } }  *// Quick Sort berdasarkan nomor (asc)* void quickSortNomor(Anggota arr[], int low, int high) {  if (low >= high) return;  int mid = low + (high - low) / 2;  int pivot = arr[mid].nomor;  int i = low, j = high;  while (i <= j) {  while (arr[i].nomor < pivot) i++;  while (arr[j].nomor > pivot) j--;  if (i <= j) {  swap(arr[i], arr[j]);  i++;  j--;  }  }  if (low < j) quickSortNomor(arr, low, j);  if (i < high) quickSortNomor(arr, i, high); } |

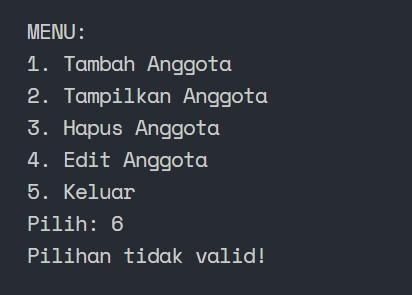
# Uji Coba dan Hasil Output

1. **Skenario 1** : Jika pengguna salah menginput Nama dan NIM/PIN 3 kali berturutturut.
2. **Skenario 2** : Jika pengguna memilih menu yang tidak terdapat dalam Menu Utama.
3. **Skenario 3** : JIka pengguna menginput nomor anggota yang tidak terdaftar.
4. **Skenario 4** : Jika pengguna menggunakan metode metode Merge Sort untuk tahun (descending)
5. **Skenario 5** : Jika pengguna menggunakan metode metode Quick Sort untuk nomor anggota (ascending)
6. **Skenario 6** : Jika pengguna menggunakan metode metode Bubble Sort untuk nama anggota (ascending)

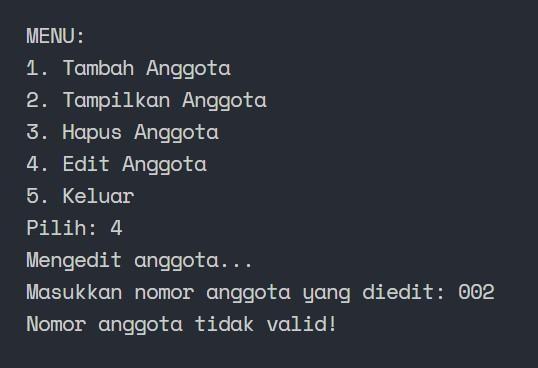
## Hasil Output



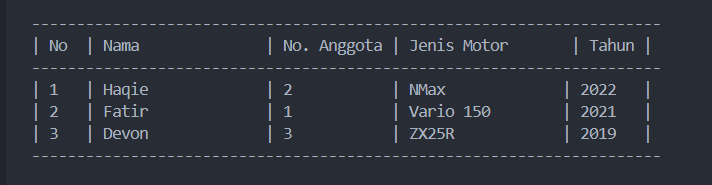
Gambar 4.1 Skenario 1

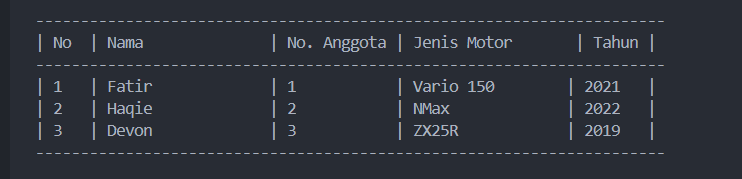


Gambar 4.2 Skenario 2

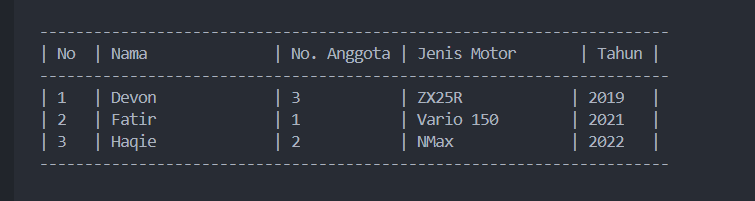


Gambar 4.3 Skenario 3

Skenario 4



Skenario 5



Skenario 6