LAPORAN PRAKTIKUM

POSTTEST 6

ALGORITMA PEMROGRAMAN

LANJUT

Disusun oleh:

Ajiva Alank Setiandra (2409106017)

Kelas (A1’24)

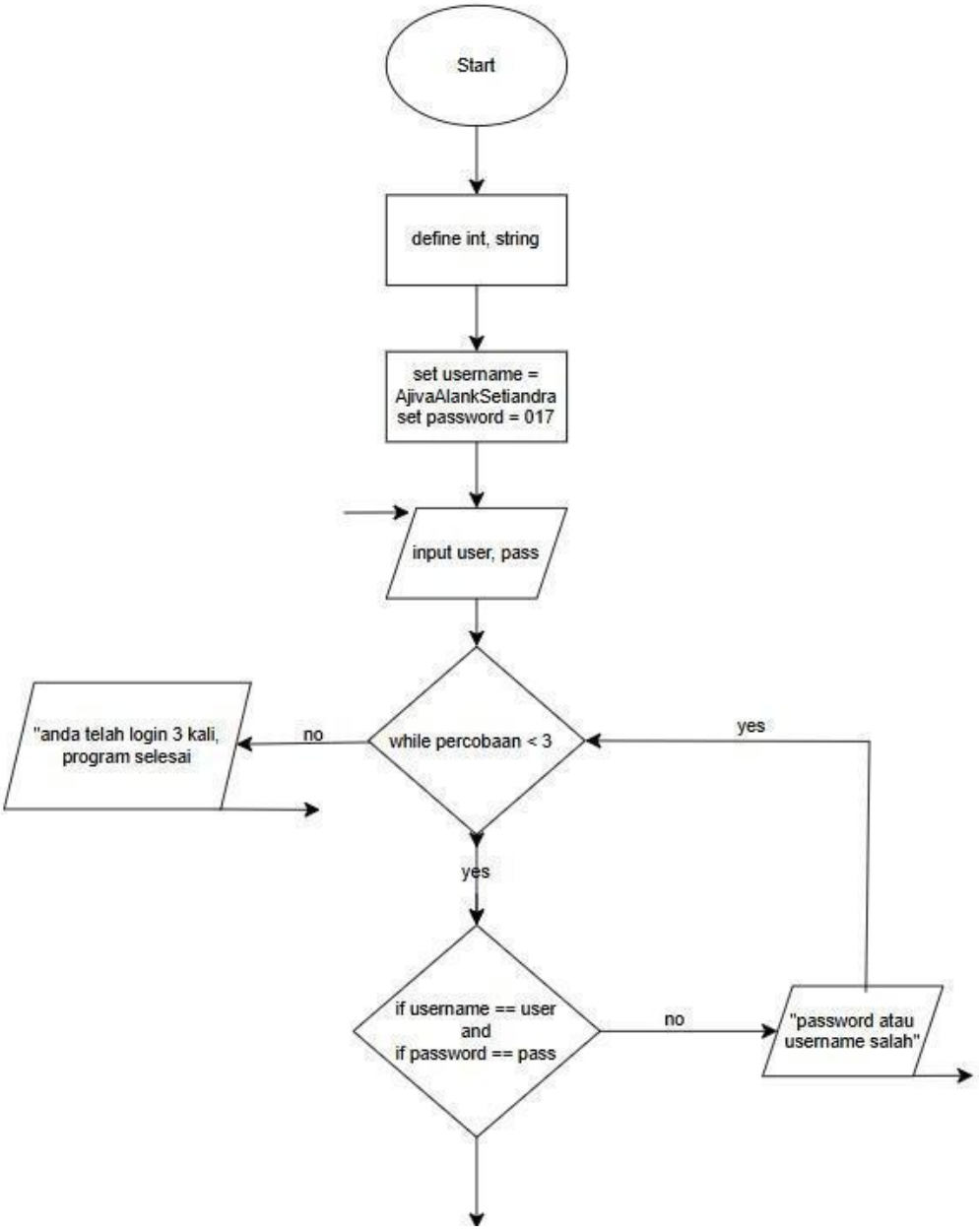
PROGRAM STUDI

INFORMATIKA UNIVERSITAS

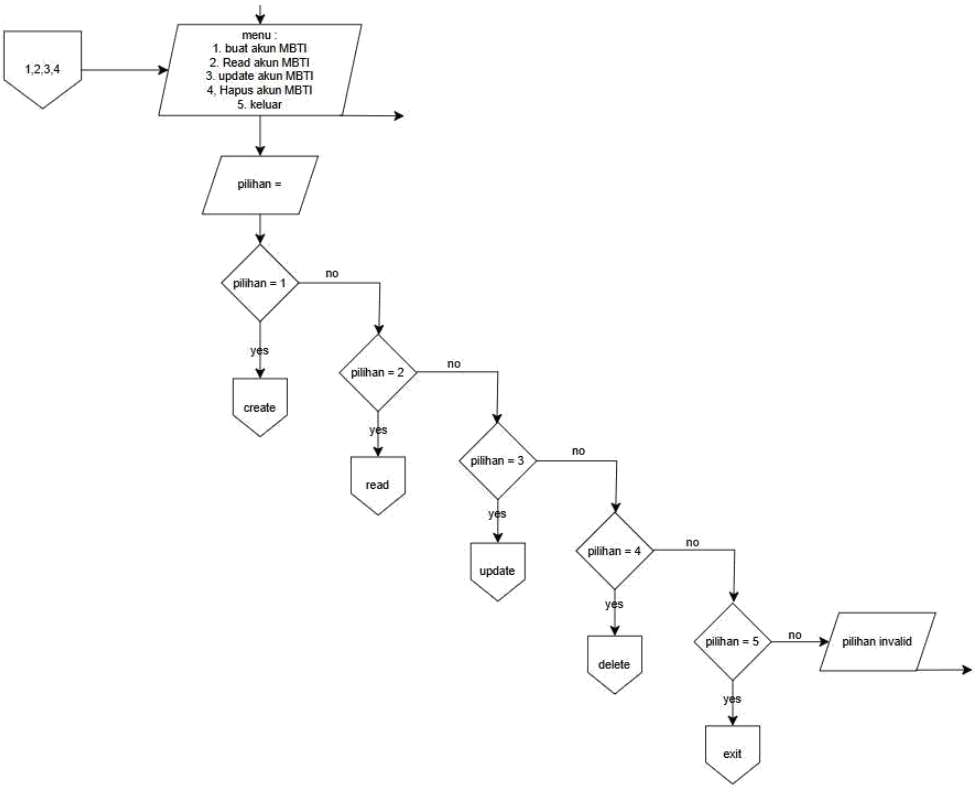
MULAWARMAN SAMARINDA

2025

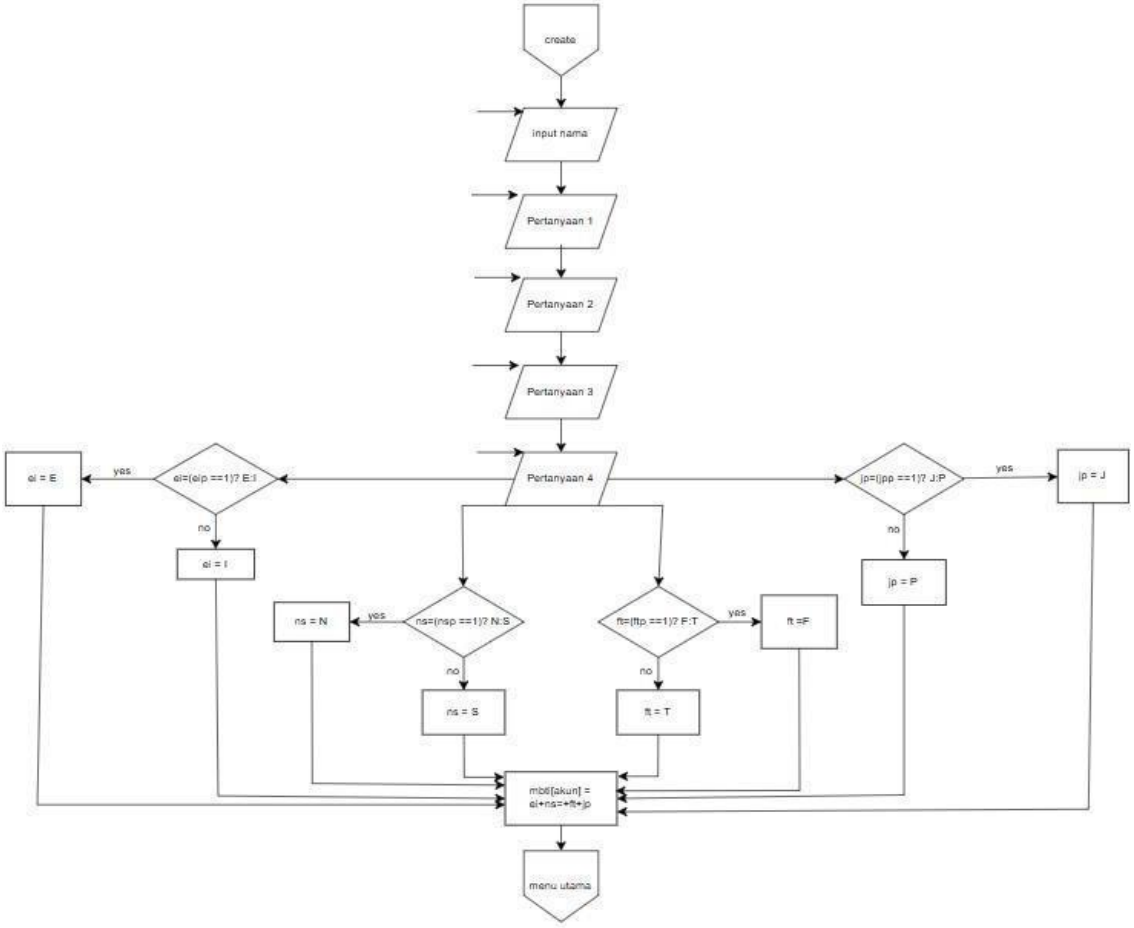
1. Flowchart



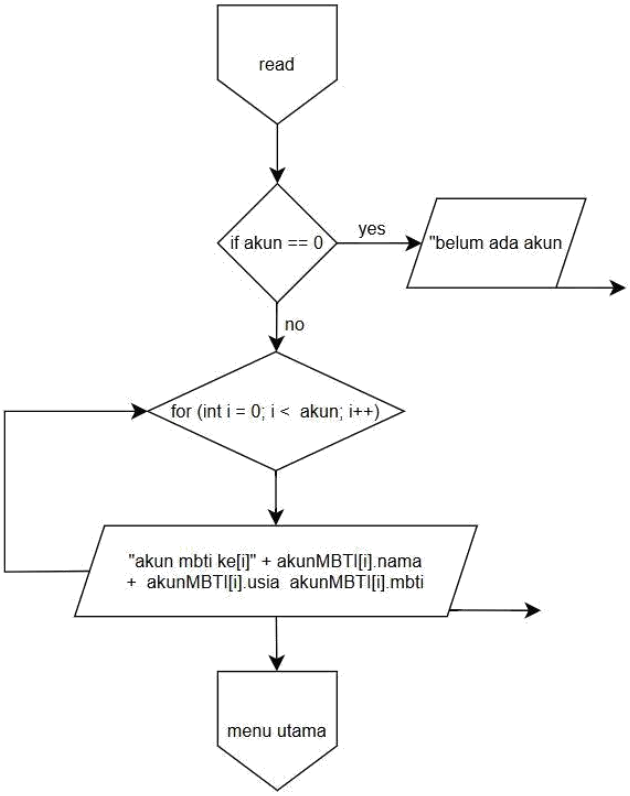
Gambar 1.1 flowchart bagian 1



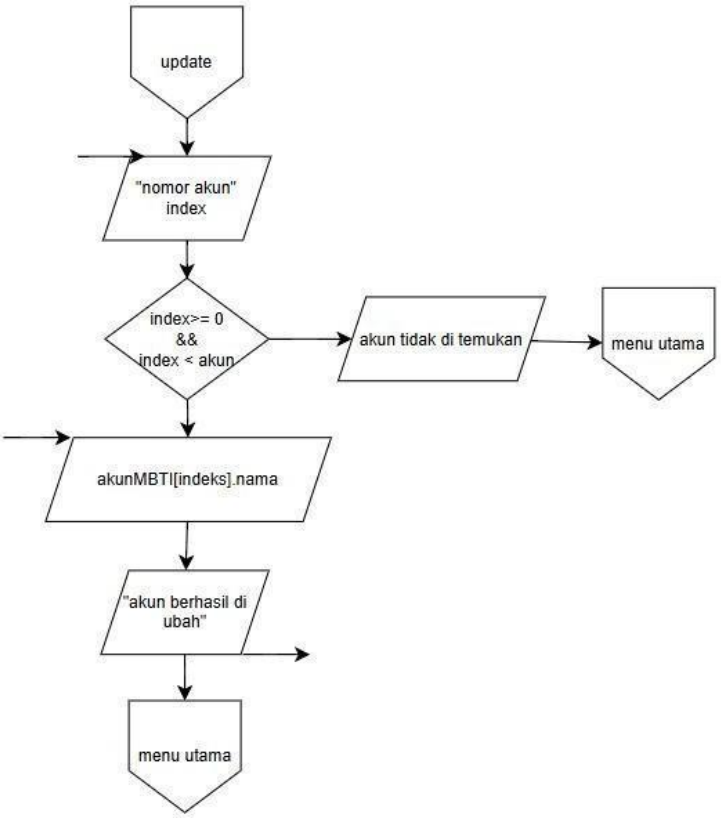
Gambar 1.2 flowchart bagian 2



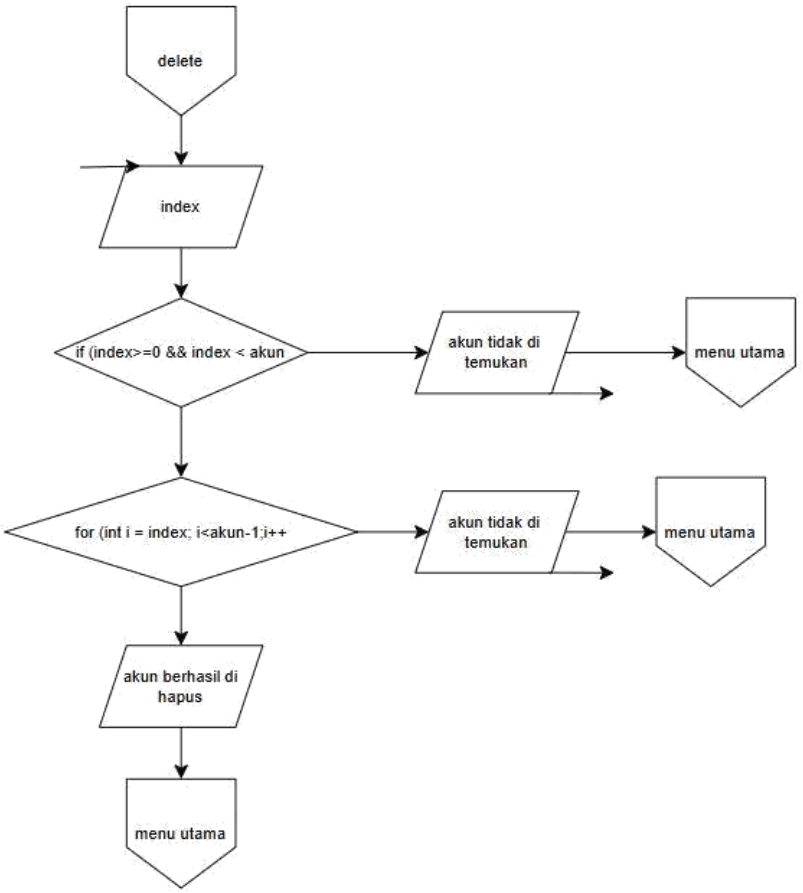
Gambar 1.3 flowchart fitur create



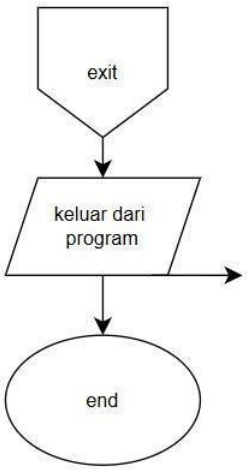
Gambar 1.4 flowchart fitur read



Gambar 1.5 flowchart fitur update



Gambar 1.6 flowchart fitur delete



Gambar 1.7 flowchart fitur exit

2. Analisis Program

Progrram memiliki tujuan berupa untuk mengikuti suatu tes kecil untuk menentukan apa mbti sang pengguna dengan cara menjawab beberapa soal, pengguna dapat menggunakan multiple akun untuk sekali pakai namun maksimal 5 akun. Selain itu pengguna dapat update. Read dan delete akun dan MBTI yang terpaparkan berdasarkan hasil dari coding

3. Source Code

A. Deklarasi tipe data

Disini tempat dimana setiap bentuk data akan di tampilkan disini, entah itu string, maupun interger, terdapat juga deklarasi array disini.

Source Code:

```
int main() {
    string username = "AjivaAlankSetiandra";
    string password = "017";
    string user, pass;
    int percobaan = 0;
    #define MBTI 5
    int akun = 0;
    AkunMBTI akunMBTI[MBTI];
    int pilihan;
```

Gambar 3.1 source code deklarasi tipe data

B. Fitur login

Fitur ini adalah fitur dimana pengguna harus mengisi password dan username dengan sesuai agar dapat lanjut kedalam program, jika salah sebanyak 3 kali, programa kan tutup dengan sendirinya.

Source Code:

```
    cout << "Selamat datang di program Manajemen MBTI" << endl;

    while (percobaan < 3) {
        cout << "Masukkan Username: ";
        cin >> user;

        if (user == username) {
            cout << "Masukkan Password (NIM): ";
            cin >> pass;

            if (pass == password) {
                cout << "Login sukses" << endl;
```

Gambar 3.1 source code deklarasi tipe data

C. Fitur create

Fitur berisi suatu pertanyaan perntanyaan yang dimana pengguna harus menjawab agar pengguna dapat melihat hasil dari MBTI mereka. Berikut berupa salah satu source codenya (tidak memasuki semuanya karena akan terlalu banyak, dan ke 4 pertanyaan memiliki bentuk yang sama namun variabel yang berbeda saja).

Terdapat variabel asing yang bernama (ei,ns,ft,jp) dan (eip, nsp, ftp, jpp), variabel ini berupa singkatan dari Extrovert/Introvert, Intuation/Sensing, Feeling/Thinking, Judgement/perceiving. Untuk yang berakhiran p sama saja, tapi itu menandakan pilihan.

Terdapat juga pointer disini yang berguna untuk menunjuk alamat dari MBTI yang di create

Source Code:

```
switch (pilihan) {
    case 1:
        if (akun < MBTI) {
            string ei, ns, ft, jp;
            int eip, nsp, ftp, jpp;

            AkunMBTI* ptrAkun = &akunMBTI[akun]; // pointer ke akunMBTI

            cout << "Masukkan nama Anda: ";
            cin >> ptrAkun->nama;
            cout << "Masukkan usia Anda: ";
            cin >> ptrAkun->usia;
            cout << " " << endl;
            cout << "<------------------------------------------------------------>"<< endl;

            cout << "1. Apakah anda suka keluar atau sendiri di kamar? (1/2)" << endl;
            cout << "a. Keluar (1)\nb. Di kamar (2)" << endl;
            cout << "Jawaban anda = ";
            cin >> eip;
            ei = (eip == 1) ? "E" : "I";
            cout << "<------------------------------------------------------------>"<< endl;

            cout << "2. Apakah anda suka melakukan hal berdasarkan tata cara yang ada atau tidak? (1/2)" << endl;
            cout << "a. Tidak (1)\nb. Iya (2)" << endl;
            cout << "Jawaban anda = ";
            cin >> nsp;
            ns = (nsp == 1) ? "N" : "S";
            cout << "<------------------------------------------------------------>"<< endl;
```

Gambar 3.3 source code fitur create

D. Fitur read

Fitur ini berisikan pembacaan akun akun yang telah di ciptakan di fitur create Pointer disini berguna untuk menarik memori atau data dari alamat yang disimpan didalam fitur create sebelumnya.

Source Code:

```
case 2:
    if (akun == 0) {
        cout << "Belum ada akun" << endl;
    } else {
        for (int i = 0; i < akun; i++) {
            AkunMBTI* ptrTampil = &akunMBTI[i];
            cout << "Akun MBTI ke-" << i + 1 << " (" << ptrTampil->nama << ", Usia: " << ptrTampil->usia << ") : " << ptrTampil->mbti << endl;
        }
    }
    break;
```

Gambar 3.4 source code fitur read

E. Fitur update

Fitur hanya berisikan fitur yang bertujuan untuk mengubah MBTI dari suatu akun yang ada, jika seandainya pengguna salah mengisi jawaban. Guna dari pointer dalam fitur update ini berguna untuk menarik atau mengabil data dari create sebelumnya lalu di replace/ di ganti dengan data yang akan terperbarui.

Source Code:

```
case 3: {
    int index;
    cout << "Masukkan nomor akun yang ingin diupdate: ";
    cin >> index;
    index--;

    if (index >= 0 && index < akun) {
        AkunMBTI* ptrUpdate = &akunMBTI[index];
        cout << "Masukkan MBTI baru untuk " << ptrUpdate->nama << ": ";
        cin >> ptrUpdate->mbti;
        cout << "MBTI berhasil diperbarui!" << endl;
    } else {
        cout << "Akun tidak ditemukan" << endl;
    }
    break;
}
```

Gambar 3.5 source code fitur update

F. Fitur delete

Fitur yang berguna untuk menghapus akun akun yang telah di buat, jika tidak ada akun yang terbuat maka program akan kembali ke menu utama.
Sama hal seperti pada bagian fitur sebelumya, pointer disini berguna untul mengambilmdata memori yang tersimpan lalu dihapus, tapi di bagian ini lebih ke di hapus.

Source Code:

```
case 4: {
    int index;
    cout << "Masukkan nomor akun yang ingin dihapus: ";
    cin >> index;
    index--;

    if (index >= 0 && index < akun) {
        AkunMBTI* ptrHapus = akunMBTI;
        for (int i = index; i < akun - 1; i++) {
            *(ptrHapus + i) = *(ptrHapus + i + 1);
        }
        akun--;
        cout << "Akun berhasil dihapus" << endl;
    } else {
        cout << "Akun tidak ditemukan" << endl;
    }
    break;
}
```

Gambar 3.6 source code fitur update

G. Fitur sorting nama ascending

Fitur ini berguna untuk merngurutkan data dari hasil test (yang berasal dari fitur create) berdasarkan urutan nama (A-Z) berdasarkan dari huruf terkecil (A) hinnga ke huruf terbesar (Z). (terurut dari capslock ke huruf kecil)

```
// Sorting nama ascending
void sortNamaAscending(AkunMBTI* akunMBTI, int akun) {
    for (int i = 0; i < akun - 1; i++) {
        for (int j = 0; j < akun - i - 1; j++) {
            if ((akunMBTI + j)->nama > (akunMBTI + j + 1)->nama) {
                AkunMBTI temp = *(akunMBTI + j);
                *(akunMBTI + j) = *(akunMBTI + j + 1);
                *(akunMBTI + j + 1) = temp;
            }
        }
    }
    cout << "Data berhasil diurutkan berdasarkan Nama (ascending)" << endl;
}
```

Gambar 3.7 source code fitur sorting ascending

I. Fitur sorting angka descending

Fitur ini berguna untuk merngurutkan data dari hasil test (yang berasal dari fitur create) berdasarkan urutan angka (1-9 dan seterusnya) berdasarkan dari agnka terbesar hingga angka terkecil.

```
// Sorting usia descending
void sortUsiaDescending(AkunMBTI* akunMBTI, int akun) {
    for (int i = 0; i < akun - 1; i++) {
        for (int j = 0; j < akun - i - 1; j++) {
            if ((akunMBTI + j)->usia < (akunMBTI + j + 1)->usia) {
                AkunMBTI temp = *(akunMBTI + j);
                *(akunMBTI + j) = *(akunMBTI + j + 1);
                *(akunMBTI + j + 1) = temp;
            }
        }
    }
    cout << "Data berhasil diurutkan berdasarkan Usia (descending)" << endl;
}
```

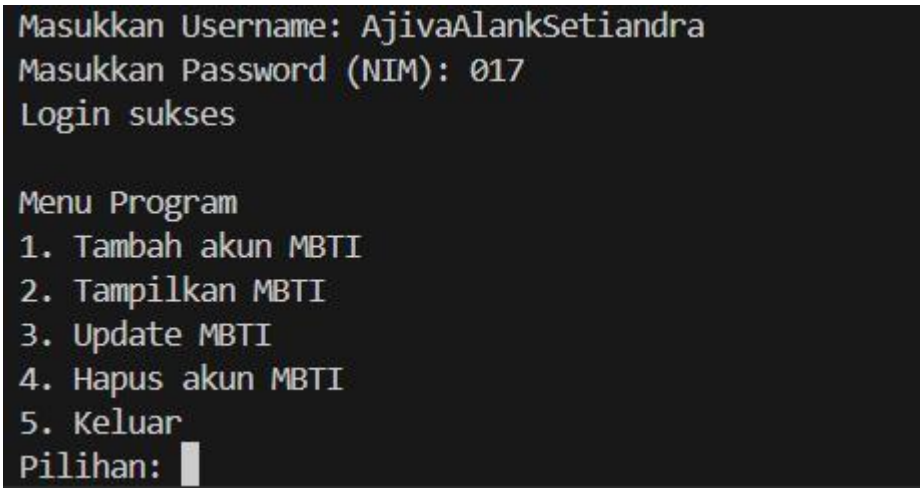
Gambar 3.7 source code fitur sorting descending

I. Fitur sorting MBTI Bubble sort

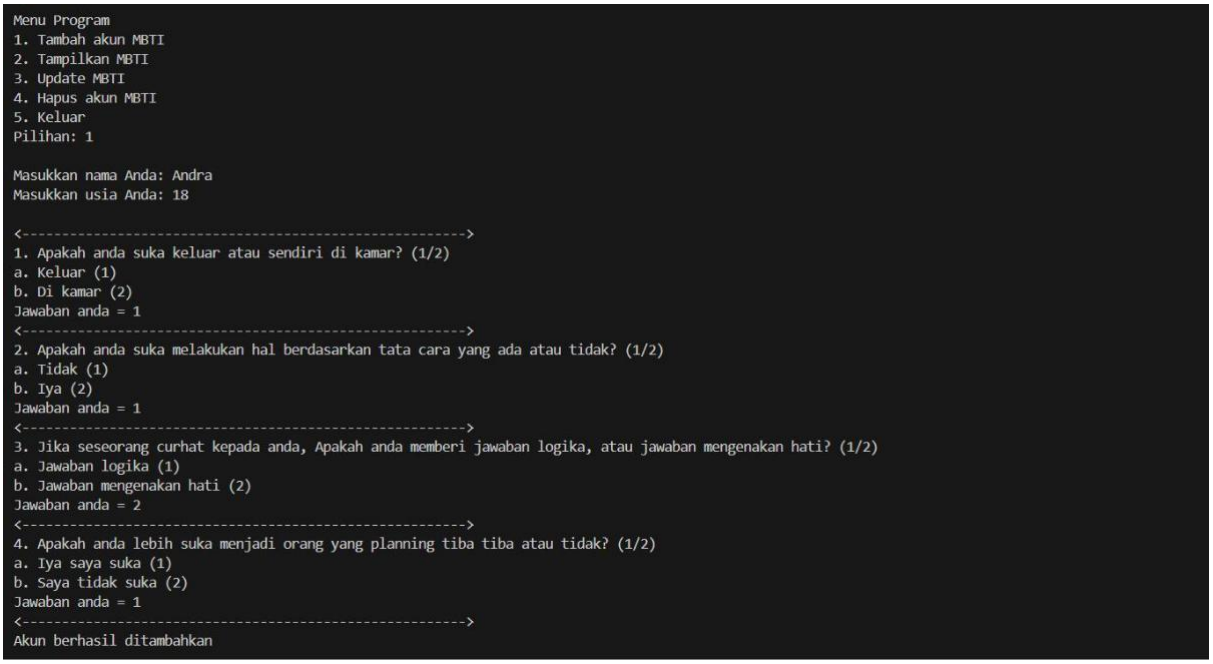
Fitur ini berguna untuk merngurutkan data dari hasil test (yang berasal dari fitur create) berdasarkan urutan MBTI berdasarkan dari elemen sebelumnya (kayak menggelumbung).

```
// Bubble sort MBTI
void bubbleSortMBTI(AkunMBTI* akunMBTI, int akun) {
    for (int i = 0; i < akun - 1; i++) {
        for (int j = 0; j < akun - i - 1; j++) {
            if ((akunMBTI + j)->mbti > (akunMBTI + j + 1)->mbti) {
                AkunMBTI temp = *(akunMBTI + j);
                *(akunMBTI + j) = *(akunMBTI + j + 1);
                *(akunMBTI + j + 1) = temp;
            }
        }
    }
    cout << "Data berhasil diurutkan berdasarkan MBTI (bubble sort)" << endl;
}
```

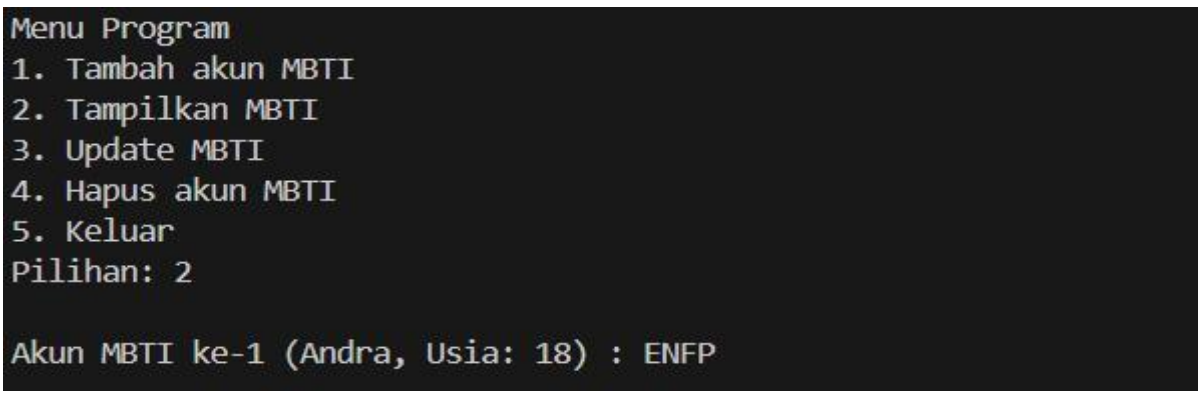
4. Uji Coba dan Hasil Output



Gambar 4.1 output login dan menu utama



Gambar 4.2 output fitur create



Gambar 4.3 output fitur read

```
Menu Program
1. Tambah akun MBTI
2. Tampilkan MBTI
3. Update MBTI
4. Hapus akun MBTI
5. Keluar
Pilihan: 3

Masukkan nomor akun yang ingin diupdate: 1
Masukkan MBTI baru untuk Andra: ISTJ
MBTI berhasil diperbarui!

Menu Program
1. Tambah akun MBTI
2. Tampilkan MBTI
3. Update MBTI
4. Hapus akun MBTI
5. Keluar
Pilihan: 2

Akun MBTI ke-1 (Andra, Usia: 18) : ISTJ
```

Gambar 4.4 output fitur update

```
Menu Program
1. Tambah akun MBTI
2. Tampilkan MBTI
3. Update MBTI
4. Hapus akun MBTI
5. Keluar
Pilihan: 4

Masukkan nomor akun yang ingin dihapus: 1
Akun berhasil dihapus

Menu Program
1. Tambah akun MBTI
2. Tampilkan MBTI
3. Update MBTI
4. Hapus akun MBTI
5. Keluar
Pilihan: 2

Belum ada akun
```

Gambar 4.4 output fitur delete

```
Menu Program
1. Tambah akun MBTI
2. Tampilkan MBTI
3. Update MBTI
4. Hapus akun MBTI
5. Keluar
Pilihan: 5

Keluar dari program
```

Gambar 4.4 output exit

```
Akun MBTI ke-1 (Aiden, Usia: 18) : INFJ
Akun MBTI ke-2 (Andra, Usia: 18) : ENFP
Akun MBTI ke-3 (Kyle, Usia: 25) : ISFP
Akun MBTI ke-4 (Wind, Usia: 22) : ENFP
Akun MBTI ke-5 (val, Usia: 19) : ESFP
```

Gambar 4.5 output sorting ascending nama

```
Akun MBTI ke-1 (Kyle, Usia: 25) : ISFP
Akun MBTI ke-2 (Wind, Usia: 22) : ENFP
Akun MBTI ke-3 (val, Usia: 19) : ESFP
Akun MBTI ke-4 (Aiden, Usia: 18) : INFJ
Akun MBTI ke-5 (Andra, Usia: 18) : ENFP
```

Gambar 4.6 output sorting descending umur

```
Akun MBTI ke-1 (Wind, Usia: 22) : ENFP
Akun MBTI ke-2 (Andra, Usia: 18) : ENFP
Akun MBTI ke-3 (val, Usia: 19) : ESFP
Akun MBTI ke-4 (Aiden, Usia: 18) : INFJ
Akun MBTI ke-5 (Kyle, Usia: 25) : ISFP
```

Gambar 4.7 output sorting bubble MBTI

5. Langkah-Langkah Git pada VSCode

```
PS C:\Users\HP\Downloads\praktikum-apl> git init
Reinitialized existing Git repository in C:/Users/HP/Downloads/praktikum-apl/.git/
```

Gambar 5.1.1 Git init

Pergi terminal dari vscode lalu ketik git init (git init adalah suatu command yang dapat di lakukan di terminal untuk menginisiasi reprisotory git.)

```
PS C:\Users\HP\Downloads\praktikum-apl> git add .
PS C:\Users\HP\Downloads\praktikum-apl> git commit -m "Yayyyy selesai #6"
[main a653334] Yayyyy selesai #6
 4 files changed, 236 insertions(+), 5 deletions(-)
 delete mode 100644 .vscode/settings.json
 create mode 100644 post-test/PT-6/2409106017-AjivaAlankSetiandra-PT-6.cpp
 create mode 100644 post-test/PT-6/2409106017-AjivaAlankSetiandra-PT-6.exe
 create mode 100644 post-test/PT-6/2409106017-AjivaAlankSetiandra-PT-6.pdf
```

Gambar 5.1.2 Git add dan commit

lalu ketikan git add (Git add adalah command di terminal yang digunakan untuk menambahkan file apa saja yang akan di commit nantinya), lalu ketik git commit untuk memastikan sebagai checkpoint jika program akan di tandai selesai maupn tidak.

```
PS C:\Users\HP\Downloads\praktikum-apl> git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 1.33 MiB | 489.00 KiB/s, done.
Total 7 (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/Ajiva-Alank-Setiandra/praktikum-apl.git
   bcd0131..a653334  main -> main
PS C:\Users\HP\Downloads\praktikum-apl>
```

Gambar 5.1.2 Git push

Git push berisikan suatu command yang dapat membuat semua file apapun yanf ada di folder akan masuk ke reprisotory yang sudah disediakan pada awalan. Ada beberapa langkah yang dilompati dikarenakan sudah terdaftarnya beberapa hal pada git, seperti git remote.