**LAPORAN FINAL PROJECT**

**PROGRAM KAS KELAS**

**PRAKTIKUM ALGORITMA DAN PEMROGRAMAN**



**OLEH :**

**KELOMPOK 2**

Jeremi Herodian Abednigo (2008561042 / C)

Daniel Surya Wijaya (2008561044 / C)

I Made Alit Darma Putra (2008561045 / C)

I Made Arthya Andika Putra (2008561052 / C)

Gede Sudimahendra (2008561057 / C)

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA & ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**2021**

**KATA PENGANTAR**

Puji syukur kami panjatkan kehadirat Tuhan Yang Maha Esa karena berkat limpahan rahmat dan karunia-Nya kami dapat menyusun laporan *final* project pada mata kuliah Praktikum Algoritma & Pemrograman dengan tepat waktu.

Tidak lupa juga saya ucapkan terima kasih kepada asisten dosen mata kuliah Praktikum Algoritma dan Pemrograman kami, yang telah membimbing kami dalam mata kuliah yang bersangkutan. Kami juga mengucapkan terima kasih pada semua pihak yang telah membantu proses penyusunan laporan ini.

Dalam tugas ini kami dapat menyelesaikan laporan dengan judul “Program Kas Kelas”. Tugas ini dibuat dalam rangka memenuhi tugas *final project* mata kuliah Praktikum Algoritma & Pemrograman. Semoga laporan buat ini dapat bermanfaat bagi saya dan semua pihak yang membacanya.

Demikian kata pengantar ini saya buat. Kami menyadari bahwa laporan ini masih sangat jauh dari kata sempurna, untuk itu kami mohon maaf bila ada kesalahan kata dalam pembuatan laporan ini maupun kata pengantar ini, juga kami meminta kritik dan saran yang membangun agar dapat dibuat laporan yang lebih baik. Semoga laporan ini bermanfaat bagi para pembaca dan dapat menambah cakrawala pengetahuan kita.

Denpasar, 19 Mei 2021

Kelompok 2

**DAFTAR ISI**

**KATA PENGANTARI**

**DAFTAR ISIII**

**BAB I. LATAR BELAKANG1**

**BAB II. LANDASAN TEORI2**

* 1. Struktur Dasar Bahasa C2
  2. Deklarasi di Bahasa C2
  3. Operator dalam Bahasa C3
  4. Input Output dalam Bahasa C3
  5. Perulangan While dan For4
     1. Perulangan While dan Do-While4
     2. Perulangan For5
  6. Percabangan If dan Switch6
     1. If Branching6
     2. Switch Branching7
  7. Array8
     1. Definisi Array8
     2. Deklarasi Array8
     3. Deklarasi Array of Char dalam C8
  8. File Handling9
     1. Membuka dan Menutup file (fopen dan fclose)9
     2. Menulis data ke file (fprintf)10
     3. Membaca data di file (fscanf)10

**BAB III. DESAIN DAN METODE11**

* 1. Tampilan Awal11
  2. Admin Mode12
  3. User Mode13
  4. Metode13

**BAB IV. HASIL DAN IMPLEMENTASI21**

* 1. Program Utama Tampilan Awal21
  2. Fungsi main()26
  3. Admin Mode28
  4. User Mode35
  5. Output Program40

**BAB V. PENUTUP46**

* 1. Kesimpulan46
  2. Saran46

**DAFTAR PUSTAKA47**

**BAB I**

**LATAR BELAKANG**

Universitas Udayana merupakan salah satu universitas yang terletak di Kabupaten Badung sudah barang tentu dalam pengelolaannya memiliki faktor-faktor pedukung hingga saat ini masih berdiri dengan baik. Di sebuah universitas pastinya terdapat berbagai macam fakultas dan jurusan, yang didalamnya terdiri dari berbagai kelas, salah satunya yaitu kelas C Program Studi Informatika, Fakultas MIPA. Tentunya faktor pengelolaan keuangan, yaitu pengelolaan dana kas menjadi sangat penting di dalam suatu kelas karena pengelolaan keuangan yang baik dapat menunjang keberlangsungan kelas tersebut. Sistem pengelolaan kas kelas pada Kelas C Jurusan Informatika masih menggunakan sistem manual yang masih kurang efektif dan efisien, karena pencatatan hanya dilakukan oleh satu orang yaitu bendahara, sehingga masih ada terjadi kesalahan saat pencatatan data penerimaan kas dan data pengeluaran kas, sehingga memerlukan pengecekan ulang dari kesalahan pada pencatatan kas dengan waktu yang sedikit lama dalam pencatatan pada buku kas dan pembuatan laporan. Aktivitas pencatatan kas yang baik adalah pencatatan kas yang rinci mengenai kondisi keuangan. Hal tersebut tidak akan didapatkan manakala menggunakan cara manual. Kemampuan untuk penyedia laporan keuangan secara detil juga harus ditunjang dalam pengelola yang diperlukan oleh bagian dan pihak manajemen dan keuangan.

Dari hasil identifikasi permasalahan yang ditemukan diperlukan suatu program manajemen kas kelas. Pada program ini, kami menggunakan bahasa pemrograman C. Bahasa pemrograman C digunakan karena bahasa ini adalah bahasa yang cukup efektif karna bersifat *compiler* sehingga cepat dalam implementasi. Program ini dibuat dan dikembangkan dengan harapan dapat memudahkan pengelolaan kas pada kelas C Program Studi Informatika dan memudahkan proses pemeliharaan program karena cukup mengupdate source code pada program. Selain itu, program ini juga dibuat untuk memenuhi penugasan *final project* mata kuliah Praktikum Algoritma & Pemrograman.

Berdasarkan latar belakang yang telah diuraikan, penelitian berikut ini membahas terkait dengan bagaimana cara membangun program pengelolaan kas kelas yaitu kas masuk dan kas keluar yang efektif dan efisien. Disini kami membuat suatu program pengelolaan kas kelas yang diharapakan dapat membantu mengatasi permasalahan keuangan kas yang biasa dihadapi pada kelas C Program Studi Informatika, dalam pencatatan pengelolaan kas kelas dan dapat memenuhi penugasan *final project* mata kuliah Praktikum Algoritma & Pemrograman. Program ini dibuat dengan menggunakan bahasa pemrograman C.

**BAB II**

**LANDASAN TEORI**

* 1. **Struktur Dasar Bahasa C**

Setiap kali ingin membuat program C, kita harus menggunakan line berikut ini:

#include <stdio.h>

int main(){

return 0

}

Kalau kita perhatikan, struktur program tersebut dibagi menjadi dua, yaitu bagian Include, dan fungsi main.Fungsi main() adalah fungsi utama dalam program. Fungsi ini akan dieksekusi pertama kali saat program dijalankan. Karena itu, kita harus menuliskan logika program di dalam fungsi ini. Pada contoh di atas kita membuat fungsi main dengan deklarasi seperti ini.Pada program C, #include berfungsi untuk mengimpor fungsi-fungsi yang sudah didefinisikan pada header file. Header file adalah file yang berisi definisi fungsi yang sudah dibuat. Tujuannya agar bisa digunakan pada program C yang lainnya. File ini berekstensi .h.

* 1. **Deklarasi di Bahasa C**

Variabel, merupakan suatu tempat di memori untuk menyimpan data sementara. Data yang ada dalam variabel dapat diubah sesuai kemauan programmer. Deklarasi pada Bahasa C dilakukan dengan mengetik , “tipe data <spasi> nama variabel” contoh, “int a” untuk mendeklarasikan variabel dengan tipe data integer atau bilangan bulat. Constant, hampir sama dengan variable constant merupakan suatu tempat di dalam memori yang dibuat untuk menyimpan data. Namun bedanya dalam constant datanya tidak dapat diubah. Deklarasi pada Bahasa C dilakukan dengan mengetik “const <spasi> nama constant <spasi> nilai konstanta”, contoh “const pp 1,5”. Function, merupakan kelompok statement yang bersama-sama menyelesaikan suatu masalah atau tugas. Dalam c kita memiliki satu function default, yaitu main(), yang selalu akan dijalankan pertama saat program di-compile. Untuk mendeklarasikan function kita menggunakan “return\_type <spasi> function\_name(parameter list);”. Command ini akan memberi tahu compiler tentang nama fungsi dan cara untuk memanggil fungsi tersebut. Body dari function akan dijelaskan secara terpisa

Macro, merupakan cara untuk mendefinisikan nilai yang konstan tanpa bisa diubah dalam kode. Hampir sama dengan Constant, namun Macro tidak memerlukan tempat di memori, serta tidak memerlukan inisialisasi di setiap fungsi yang ada. Yang menyebabkan macro lebih hemat memori dibanding Constant. Cara untuk mendefinisikan constant adalah dengan

“#define <spasi> nama <spasi> nilai” contoh,

“#define dimapp 3,5”

* 1. **Operator dalam Bahasa C**

Arithmetic Operator, merupakan operator dasar yang kita kenal sehari-hari, berhubungan dengan operasi aritmatika seperti penjumlahan, pengurangan, perkalian, dan pembagian. Contoh dari operatornya adalah, +, -, \*, dll Relational Operator, merupakan operator yang membandingkan relasi antara dua operand. Misalnya, mengecek apakah dua operand nilainya sama, berbeda, lebih besar dari, kurang dari, dan lainnya. Contoh dari operatornya adalah, ==, !=, >, <

Logical Operator, merupakan operator yang digunakan untuk menghubungkan dua atau lebih ekspresi yang memiliki nilai kebenaran, kemudian menghasilkan kesimpulan dari kumpulan ekspresi tersebut. Misalnya, Logical AND, OR, dan NOT, contoh operatornya, &&, ||, dan !

Bitwise Operator, merupakan operator yang melakukan manipulasi data pada tingkat bit. Merupakan operator yang bisa juga digunakan untuk shifting bits. Bitwise operator tidak bisa diaplikasikan pada floar atau double. Contohnya Bitwise AND(&) , Bitwise OR(|), dll Assignment Operator, merupakan operato yang memanipulasi operand sekaligus meng-assign hasil dari manipulasi ke operand tersebut. Hampir mirip dengan Arithmetic Operator, mungkin bisa dibilang merupakan pemanjangannya. Contoh dari operator ini adalah \*=(Multiply left operant assign to left operand), =, assign values from right side operands to left side operand. Conditional Operator, merupakan operator yang mengecek apakah kondisi ekspresi terpenuhi, kemudian mengeksekusi ekspresi lain dengan syarat apakah ekspresi True atau False. Contoh penulisan, ekspresi 1? Ekspresi 2: ekspresi 3. Saat ekspresi satu terpenuhi maka ekspresi 2 akan dieksekusi, saat tidak terpenuhi maka ekspresi 3 akan dieksekusi.

* 1. **Input Output dalam Bahasa C**

Dalam Bahasa C, kita bisa menggunakan printf(), untuk meng-outputkan nilai ke console. Data di-ouput dengan menggunakan juga beberapa format specifier. Seperti %f, %d, %c, dll.

Untuk menginputkan data kita bisa menggunakan fungsi scanf(), dengan tambahan beberapa type specifier. Namun sebelumnya kita harus sudah mendeklarasikan variable yang ingin kita gunakan sebagai tempat dan menggunakan type specifier yang cocok dengan variabel tersebut.

* 1. **Perulangan While dan For**
     1. **Perulangan While dan Do-While**

Perulangan while secara intuitif, akan bekerja dengan cara mengulang *statement* yang ada di dalam bracket while. Hingga kondisi while tersebut tidak terpenuhi. Jika secara intuitif saya terjemahkan, (semasih *statement*), maka lakukan {*statement*}. Saat statement sudah tidak terpenuhi lagi maka looping akan berhenti. While biasa digunakan saat kita tidak tahu berapa kali looping akan terjadi. Contoh penggunaan while adalah,

1. Pada pembuatan menu,

While dapat digunakan untuk looping sebuah menu, hingga user keluar dari program. While biasa digunakan karena kita tidak tahu berapa kali user akan mengulang menu yang dia inginkan. Contoh implementasi,

….

while(i != 1){

scanf(“%d”, menu);

….

if(i == 3){

i == 1;

}

….

}

Program ini akan mengulang terus menu hingga user menginput 3, untuk keluar dari program tersebut.

1. Perulangan yang banyaknya tidak diketahui,

Fungsi utama while adalah sebagai cara untuk melakukan perulangan saat kita tidak tau berapa kali harus terjadi perulangan tersebut, Contoh implementasinya,

…

while(n > 0) {

system.out.println(n + "...");

waitASecond ();

n -= 1;

}

Perulangan while memiliki alternatif lain yaitu do-while. Jika while akan mengecek condition terlebih dahulu sebelum melaksanakan statement. Do while akan melaksanakan statement terlebih dahulu setelah itu baru mengecek.

* + 1. **Perulangan For**

Perulangan For digunakan saat kita sudah tahu pasti berapa kali perulangan tersebut akan terjadi. For loop juga digunakan saat kita memerlukan berapa kali atau informasi tentang ini loop ke-berapa, baik untuk di print atau untuk operasi. Perulangan For bisa di set dengan tiga statement, statement yang menyatakan variabel yang akan di increment, lalu Batasan increment tersebut, dan jumlah increment setiap loop. Berikut contohnya dalam Bahasa c.

…..

for(i=0;i<5;i++){

……

}

Statement for loop diatas akan melakukan looping sebanyak 4 kali, dimulai dari I = 0, hingga I = 4. For loop biasa digunakan untuk looping dengan jumlah yang sudah kita ketahui, atau, jumlah looping tersebut bisa dicari dengan apa-apa saja yang sudah kita tahu sekarang. Contohnya,

1. Looping Benda di dalam Array

Kita bisa mencari jumlah benda yang ada di dalam array dengan menggunakan fungsi len(), karena kita sudah tahu berapa kali kita harus melakukan looping, kita bisa melakukan operasi pada array.

…..

char Array[15]

for(i=0; i<=len(array); i++){

print(“%s”, array);

}

…..

1. Looping Saat kita memerlukan Informasi loop

Kita bisa menggunakan for loop untuk mendapatkan informasi tentang ini looping ke berapa, dan ingin mencetak atau melakukan operasi sejumlah tertentu. Contoh,

…..

for(i=0; i<= 5; i++){

print(“%d”, i)

}

* 1. **Percabangan If dan Switch**
     1. **If Branching**

If branching digunakan saat kita ingin melakukan percabangan yang tidak terlalu banyak dan kita ingin hanya mengeksekusi satu case saja. Atau saat kita memerlukan percabangan yang mengecek banyak kondisi dari lebih dari satu variabel. Contoh,

….

if(n == a && c == b){

printf(“Keqing Wangy”)

}

else if(n == c && c == a){

printf(“Ganyu Wangy”)

}

else{

printf(“Klee Wangy Wangy Wangy!”)

}

…..

* + 1. **Switch Branching**

Percabangan switch digunakan saat kita memerlukan banyak percabangan yang tidak menggunakan terlalu banyak kondisi, switch case biasanya hanya support percabangan satu variabel. Switch merupakan sintaks untuk menentukan variabel apa yang akan di cek, dan case merupakan condition yang akan dicocokkan. Sementara default merupakan line yang akan dijalankan jika tidak ada case yang terpenuhi. Selanjutnya di beberapa Bahasa pemrograman lain seperti python switch dihilangkan dan diganti dengan tipe data dictionary. Berikut contohnya pada Bahasa c,

…

Switch(n){

Case “keqing”:

Printf(“Keqing wangy”);

Break;

Case “ganyu”:

Prinf(“Ganyu wangy”);

Break;

Case “klee”:

Printf(“Moshi-moshi keisatsu desu?”);

Break;

Default:

Printf(“Rip Gacha, try again next time);

Break;

…..

* 1. **Array**
     1. **Definisi Array**

Array adalah tempat untuk menyimpan kumpulan data. Dalam pengertian saya, array bisa diibaratkan sebagai lemari yang berisi banyak lubang. Namun setiap lemari tersebut hanya bisa menyimpan satu jenis barang saja. Char, int, atau float saja tidak bisa digabungkan dengan tipe data yang lain dalam satu lemari atau satu array.

* + 1. **Deklarasi Array**

Array dapat dideklarasikan dengan mengetik, tipe\_data nama\_array[ukuran],tipe data meliputi, char, int, float, Boolean, dan double. Deklarasi array ini merupakan contoh untuk deklarasi array dalam 1 dimensi. Berikut merupakan cara deklarasi array dan contohnya jika ingin membuat array lebih dari satu dimensi.

Deklarasi Array 2 Dimensi:

tipe\_data nama\_array[ukuran\_x][ukuran\_y]

Seperti jumlah dimensinya array membutuhkan dua ukuran yaitu ukuran y dan ukuran x. Dan akan menghasilkan array 2 dimensi seperti di bawah.

[0, 0, 0, 0]

[0, 0, 0, 0]

* + 1. **Deklarasi Array of Char dalam C**

#include <stdio.h>

#Declare Global Variable AOC

char string[10] = “Dima”;

void main(){

printf(“%s”, string);

}

Kode diatas akan mendeklarasikan array of char atau string, dan memberi nilai dima pada array tersebut, kemudian mencetaknya. Bisa kita lihat penggunaan format.

tipe\_data nama\_array[ukuran\_x]

Kita menggunakan char sebagai tipe data karena char atau huruf merupakan penyusun string atau kata. Selanjutnya nama array saya gunakan string karena itu merupakan nama paling intuitif yang bisa kita pakai. Selanjutnya ukuran array saya gunakan 10 karena cukup untuk menampung data yang saya inginkan.

* 1. **File Handling**

File handling dalam Bahasa c dibentuk oleh kesatuan dari beberapa fungsi file handling yang saling bekerja sama. Dalam program ini File Handling merupakan hal yang utama, karena merupakan inti dari yang program ini ingin capai. Fungsi ini memiliki fungsi masing-masing. Beberapa diantaranya :

* + 1. **Membuka dan Menutup file (fopen dan fclose)**

Dapat digunakan dengan cara:

Nama\_pointer = fopen(“nama\_file”, “r”);

|  |  |
| --- | --- |
| r | opens a text file in read mode |
| w | opens a text file in write mode |
| a | opens a text file in append mode |
| r+ | opens a text file in read and write mode |
| w+ | opens a text file in read and write mode |
| a+ | opens a text file in read and write mode |
| rb | opens a binary file in read mode |
| wb | opens a binary file in write mode |
| ab | opens a binary file in append mode |
| rb+ | opens a binary file in read and write mode |
| wb+ | opens a binary file in read and write mode |
| ab+ | opens a binary file in read and write mode |

Ada beberapa mode dalam fopen yang bisa digunakan meliputi:

fclose digunakan untuk menutup file dengan cara:

fclose(pointer\_name);

* + 1. **Menulis data ke file (fprintf)**

Data dapat ditulis ke file dengan pertama dibuka dengan fopen lalu selanjutnya melakukan operasi yang kita inginkan, salah satunya fprintf.

Fungsi ini dapat digunakan dengan cara,

fprintf(fp, “Hello\n”);

Data yang dituliskan dengan fprintf akan diformat dengan formatting dan tidak akan memberikan newline secara otomatis. Alternatifnya ada fputs() dan fputc(), perbedaannya fputs akan menghasilkan newline secara otomatis dan fputc akan menuliskan single character.

* + 1. **Membaca data di file (fscanf)**

Fungsi ini dapat digunakan dengan cara,

fscanf(fp, "%s", buff)

Biasanya digunakan dalam loop contoh,

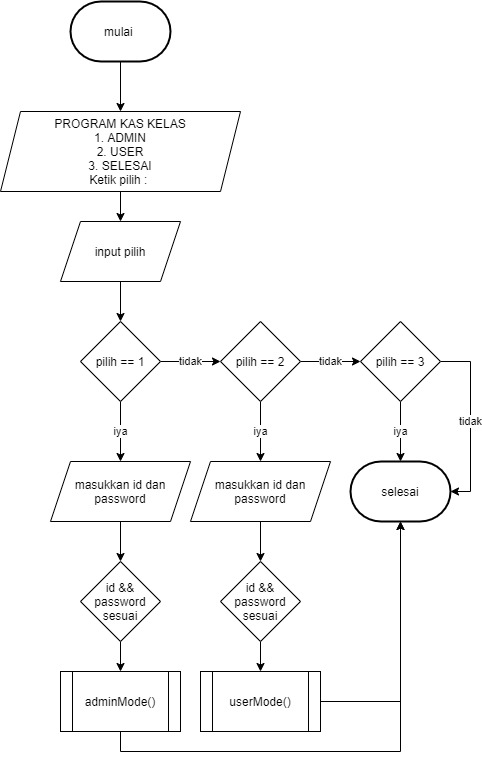
while(fscanf(fp, "%s", buff)!=EOF)

Data yang dibaca dalam fscanf dapat disimpan dalam array of char yang disini saya namai buff. Data ini akan disimpan termasuk dalam new linenya. Kita bisa menambah syarat pada looping, contoh, jika kita tidak ingin newline kita bisa menambahkan != “\n”. Alternatifnya ada fgets() dan fgetc(), perbedaannya fputs akan menghasilkan newline secara otomatis dan fputc akan menuliskan single character. Perbedaannya fungsi fgets akan membaca hingga newline.

**BAB III**

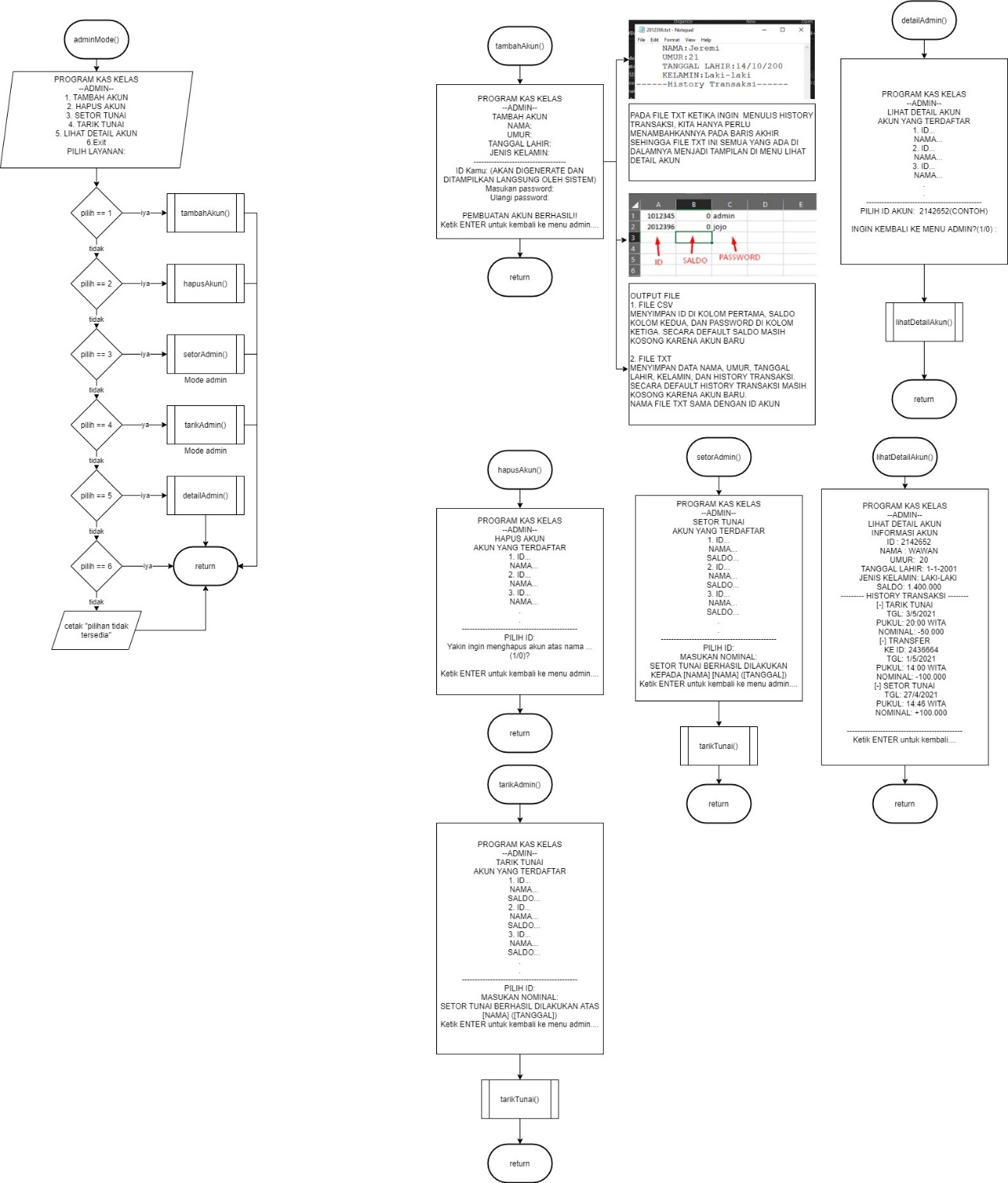
**DESAIN DAN METODE**

* 1. **Tampilan Awal**



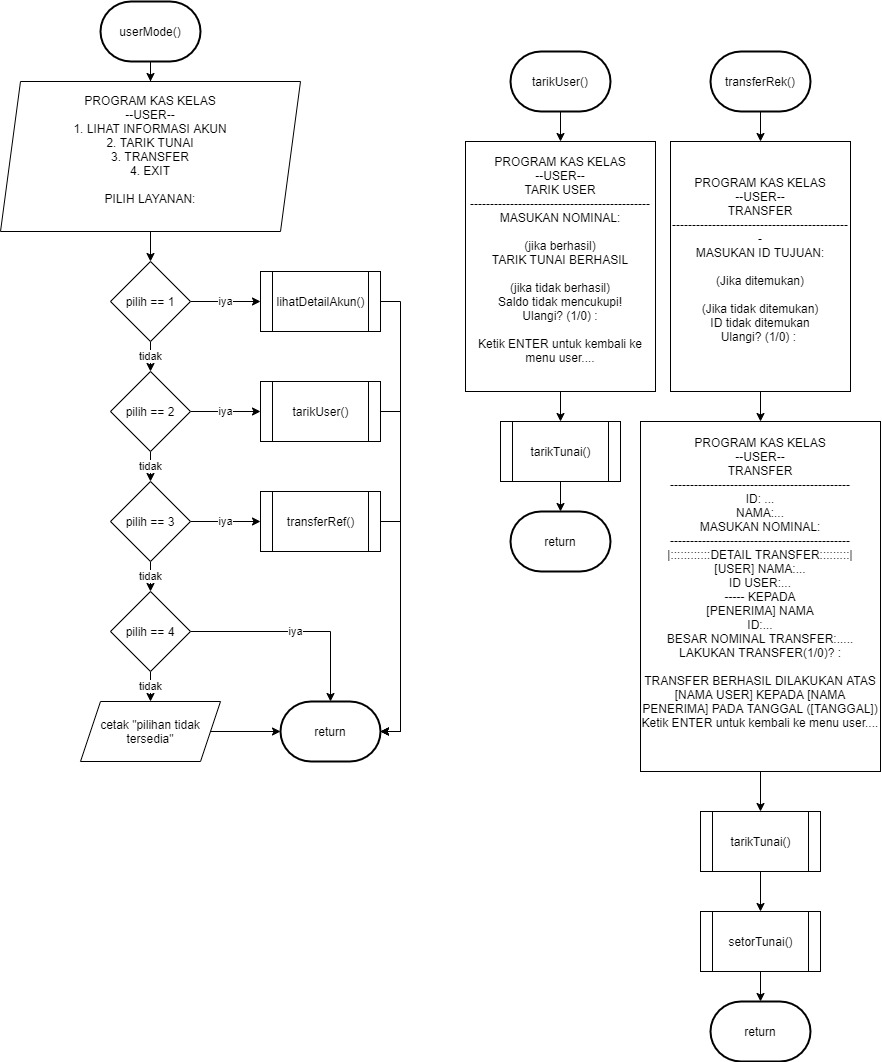
Flowchart di atas adalah desain bagaimana sebuah tampilan awal program, akan dibuat dua pilihan login, yaitu user dan admin.

* 1. **Admin Mode**



Ini adalah flowchart untuk tampilan admin. Fitur-fitur yang dimiliki mode admin lebih banyak dibandingkan dengan mode User. Setiap pilihan user akan memanggil fungsinya masing-masing.

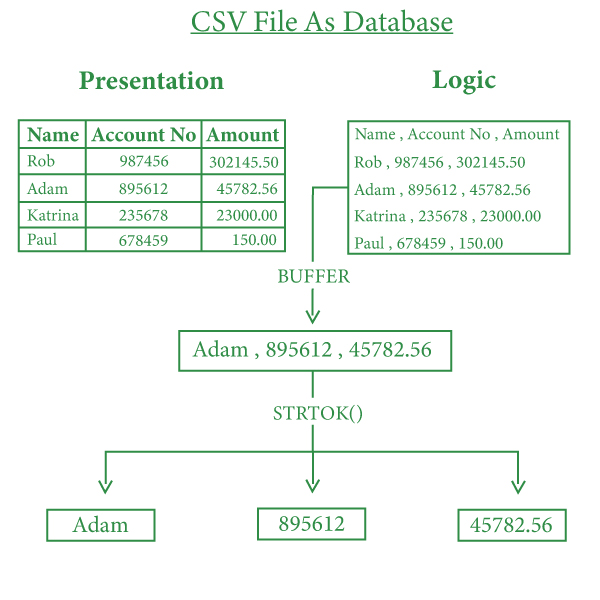
* 1. **User Mode**



Ini adalah flowchart untuk mode user. Mode user hanya memiliki 3 menu, yaitu lihat informasi akun, Tarik tunai, dan transfer ke sesame user.

* 1. **Metode**

Metode utama yang digunakan pada program ini adalah sebuah operasi file csv dan txt. File csv digunakan untuk menyimpan informasi akun seperti ID, saldo dan password. Untuk penerapan bagaimana operasi file pada sebuah file csv mungkin sedikir berbeda dengan file txt. Karena pada dasarnya file csv adalah file yang digunakan untuk menyimpan dan mengola data seperti file excel pada umumya.



File csv itu sendiri secara logika berbeda dengan file excel (.xls). file csv tidak bisa memuat formula dan bersifat text yang di mana tiap kolom dipisahkan dengan tanda koma, seperti nama csv itu sendiri adalah **C**omma **S**eparated **V**alues. Dalam pembacaannya kita hanya membacanya secara sederhana seperti biasa kita membaca file txt, hanya yang membedakan kita gunakan sebuah fungsi dari library string.h yaitu strtok untuk membagi tiap-tiap kolom. Dan ini dilakukan secara perulangan menggunakan while.

Ketika dalam pembacan filenya kita memasukannya dalam buffer. Buat buffer dari array karakter (dapat disebut sebagai string) yang mengambil semua data yang ada dalam file dan dengan menggunakan File Pointer dan fgets () data dapat diekstraksi. Gunakan dua variabel baris dan kolom yang akan mempertahankan identifikasi unik dari setiap entri.

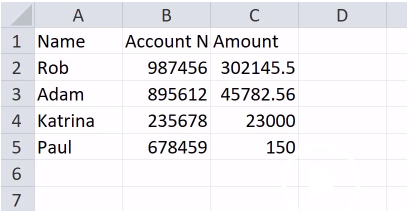
Karena string berisi koma ',' untuk memisahkan nilai, jadi idenya adalah menggunakan fungsi strtok () untuk memisahkan nilai. Fungsi ini membagi string menggunakan pembatas di sini kita menggunakan ','.

**Cara Ekstrasi datanya :**

Ekstraksi Data berkaitan dengan Membuka file CSV yang ada dan mengekstrak serta mencetak seluruh data di konsol.

1. Buka File CSV menggunakan File Pointer.
2. Ekstrak seluruh data file ke dalam array buffer karakter.
3. Sekarang inisialisasi variabel baris dan kolom dengan nilai 0.
4. Cetak data yang dipisahkan dengan koma dan tingkatkan variabel kolom.
5. Saat mencapai akhir entri baris, inisialisasi variabel kolom ke 0 dan tingkatkan variabel baris.
6. Ulangi langkah 4 dan 5, hingga penunjuk mencapai akhir file.
7. Tutup file.

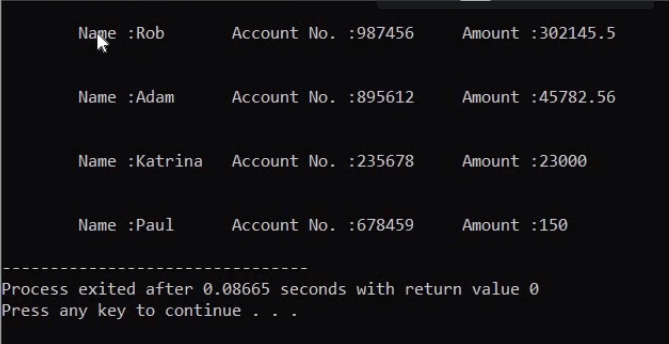
Contoh :



Berikut contoh implementasi dalam program

*// C program for the above approach*#include <conio.h>  
#include <stdio.h>  
#include <string.h>  
  
*// Driver Code  
int* main()  
{  
 *// Substitute the full file path  
 // for the string file\_path* FILE\* fp = fopen("file\_path", "r");  
  
 *if* (!fp)  
 printf("Can't open file\n");  
  
 *else* {  
 *// Here we have taken size of  
 // array 1024 you can modify it  
 char* buffer[1024];  
  
 *int* row = 0;  
 *int* column = 0;  
  
 *while* (fgets(buffer,  
 1024, fp)) {  
 column = 0;  
 row++;  
  
 *// To avoid printing of column  
 // names in file can be changed  
 // according to need  
 if* (row == 1)  
 *continue*;  
  
 *// Splitting the data  
 char*\* value = strtok(buffer, ", ");  
  
 *while* (value) {  
 *// Column 1  
 if* (column == 0) {  
 printf("Name :");  
 }  
  
 *// Column 2  
 if* (column == 1) {  
 printf("\tAccount No. :");  
 }  
  
 *// Column 3  
 if* (column == 2) {  
 printf("\tAmount :");  
 }  
  
 printf("%s", value);  
 value = strtok(**NULL**, ", ");  
 column++;  
 }  
  
 printf("\n");  
 }  
  
 *// Close the file* fclose(fp);  
 }  
 *return* 0;  
}

Output Hasil Program



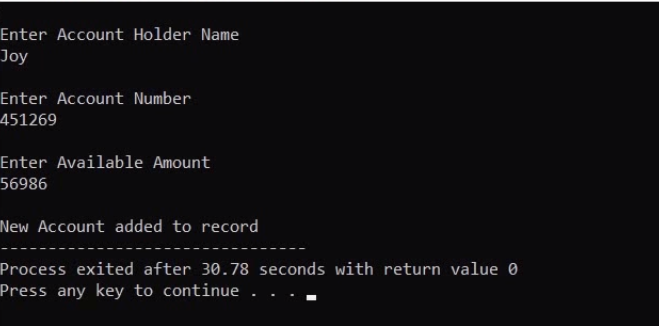
Untuk menambahkan datanya ke dalam file csv. Di sini kita gunakan fprintf(). Penambahan Data berkaitan dengan membuka file CSV yang ada, mengambil input pengguna untuk data yang akan ditambahkan ke file, dan kemudian menambahkan data ini ke file CSV.

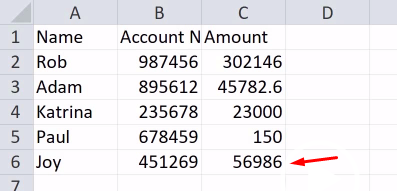
1. Buka File CSV menggunakan File Pointer dalam mode append yang akan menempatkan penunjuk ke bagian akhir file.
2. Ambil Masukan dari pengguna dalam variabel sementara.
3. Gunakan fprintf () dan pisahkan variabel sesuai dengan urutan dan komanya.
4. Tutup file.

Berikut contoh implementasi dalam program

*// C program for the above approach*#include <conio.h>  
#include <stdio.h>  
#include <string.h>  
  
*// Driver Code  
int* main()  
{  
 *// Substitute the file\_path string  
 // with full path of CSV file* FILE\* fp = fopen("file\_path", "a+");  
  
 *char* name[50];  
 *int* accountno, amount;  
  
 *if* (!fp) {  
 *// Error in file opening* printf("Can't open file\n");  
 *return* 0;  
 }  
  
 *// Asking user input for the  
 // new record to be added* printf("\nEnter Account Holder Name\n");  
 scanf("%s", &name);  
 printf("\nEnter Account Number\n");  
 scanf("%d", &accountno);  
 printf("\nEnter Available Amount\n");  
 scanf("%d", &amount);  
  
 *// Saving data in file* fprintf(fp, "%s, %d, %d\n", name,  
 accountno, amount);  
  
 printf("\nNew Account added to record");  
  
 fclose(fp);  
 *return* 0;  
}

Output Program





**Operasi File Text**

Operasi file text digunakan untuk menyimpan detail dari sebuah akun meliputi nama, umur, tanggal lahir dan jenis kelamin. Selain itu file text juga digunakan sebagai penyimpan riwayat transaksi yang dilakukan. Tidak ada metode khusus yang digunakan dalam operasi file text ini, data hanya ditambahkan baris per baris begitu pula untuk sistem pembacaan file.

Pembaca file text

Pembacaan file dilakukan dengan menggunakan fungsi fgets() karena file tidak akan dibaca secara terformat melainkan hanya sebagai string. Langkah-langkah pembacaan sebuah file text yaitu:

1. Buka File txt menggunakan File Pointer dalam mode read karena file text dibuka hanya untuk tujuan pembacaan.
2. Jika file tidak ada maka program tidak akan dilanjutkan.
3. Deklarasikan sebuah array sebagai penampung karakter yang akan dibaca.
4. Fungsi fgets() digunakan untuk membaca setiap karakter yang ada pada file kemudian disimpan pada array penampung.
5. Karakter yang telah tersimpan dalam array selanjutnya dicetak menggunakan fungsi printf().
6. Selama fungsi fgets() tidak mengembalikan nilai NULL, ulangi langkah 3 dan 4.
7. Jika fgets() mengembalikan nilai NULL, menandakan bahwa penunjuk telah mencapai akhir dari file.
8. Tutup file.

Implementasi pembacaan file text dalam program untuk menampilkan detail dari sebuah akun :

|  |  |
| --- | --- |
| Kode program | Output program |
|  |  |

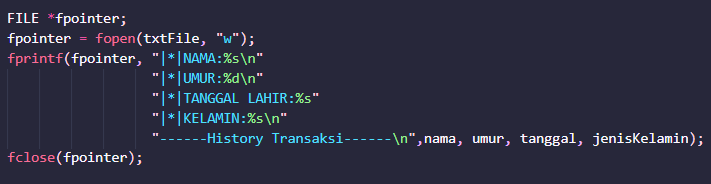
**Penulisan file text**

Isi file text ditulis dengan menggunakan fungsi fprintf(). Identitas dan transaksi yang dilakukan akan ditulis kedalam file text tersebut. Langkah-langkah menulis file text yaitu:

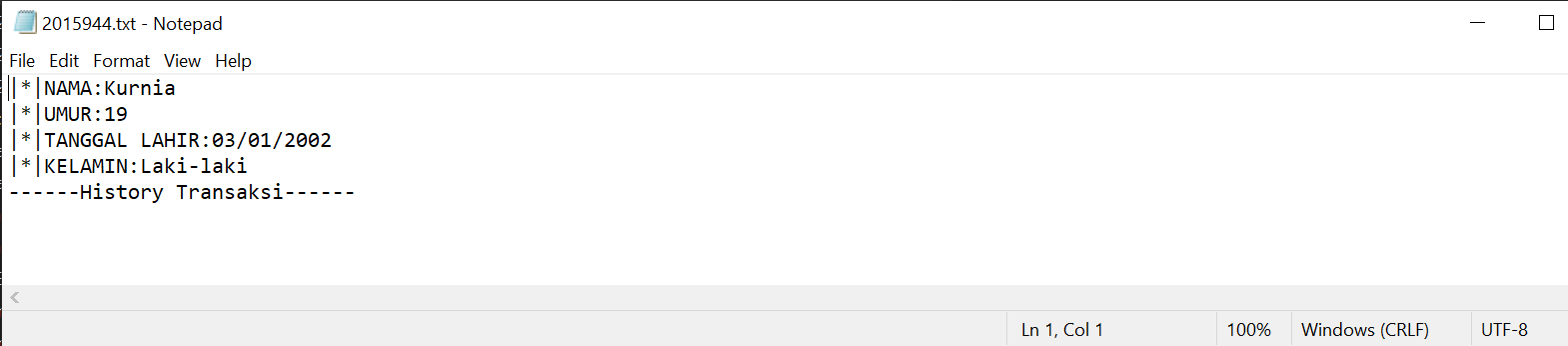
1. Buka File txt menggunakan File Pointer dalam mode append karena akan dilakukan penulisan pada akhir file.
2. Tulis data ke dalam file menggunakan fungsi fprintf.
3. Setelah penulisan selesai, tutup kembali file.

Implementasi penulisan file text dalam program :

Kode program :



Output pada file :



**BAB IV**

**HASIL DAN IMPLEMENTASI**

* 1. **Program Utama Tampilan Awal**

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <time.h>

Pada library yang dipakai di program ini ada 4.

#include <stdio.h>

Sebagai standar input output program yang merupakan standar library wajib dalam bahasa C

#include <stdlib.h>

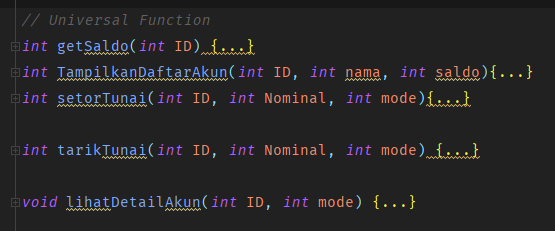
Sebagai bagian untuk kebutuhan library tambahan

#include <string.h>

Sebagai manipulasi string untuk memodifikasi beberapa input string

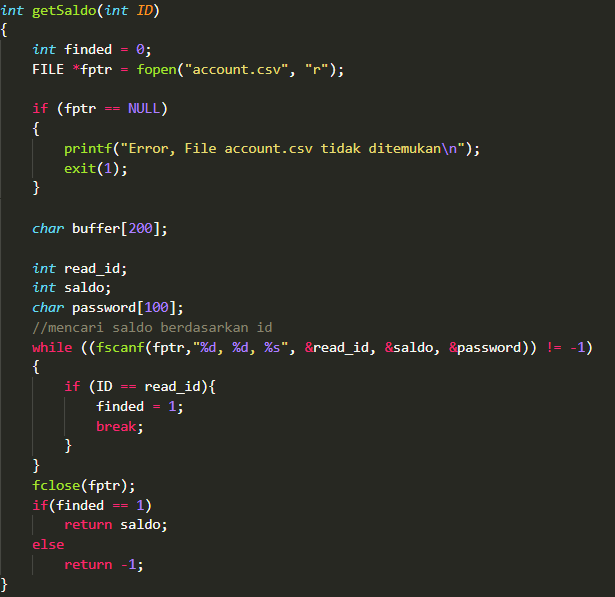
#include <time.h>

Sebagai pembaca waktu ketika transaksi terjadi



Berikut ini adalah universal function, diberikan nama seperti itu karena fungsi ini bisa dipakai untuk kedua mode, yaitu mode user dan mode admin. Yang akan dijelaskan lebih detail pada penjelasan berikutnya

**Fungsi getSaldo()**



Fungsi getSaldo merupakan fungsi universal yang sesuai dengan namanya untuk membaca saldo dari user bersangkutan pada database file csv. Fungsi getSaldo ini mendapatkan sebuah parameter berupa ID dari user. File account.csv akan dibuka kemudian setiap akun yang ada akan dibaca. Jika ID dari akun yang dicari ditemukan maka fungsi ini akan mengembalikan saldo dari ID tersebut, sedangkan jika tidak ada ID yang cocok dengann ID yang dicari maka akan dikembalikan nilai -1 yang berarti bahwa akun tidak ditemukan.

**Fungsi TampilkanDaftarAkun()**

*int* TampilkanDaftarAkun(*int* ID, *int* nama, *int* saldo){  
 FILE \* fpointer, \*fp;  
 fpointer = fopen("account.csv", "r");  
  
 *char* dumpCsv[100];  
 *char* dumpTxt[100];  
 *char* fileNameTXT[15];  
 *char* IdLog[15];  
 *char* saldoLog[100];  
 *char* \*token;  
 *int* cek = 0;  
 *while* (fgets(dumpCsv, *sizeof*(dumpCsv), fpointer)){  
 cek++;  
 }  
 *if*(cek == 1){  
 printf("Tidak ada akun yang terdaftar\n");  
 *return* 1;  
 }  
 fclose(fpointer);  
 fpointer = fopen("account.csv", "r");  
 fgets(dumpCsv, *sizeof*(dumpCsv), fpointer);  
 *while* (fgets(dumpCsv, *sizeof*(dumpCsv), fpointer)){  
 token = strtok(dumpCsv, ",");  
 strcpy(IdLog, token);  
 token = strtok(**NULL**, ",");  
 strcpy(saldoLog, token);  
  
 *if* (ID == 1){  
 printf(" ID: %s\n", IdLog);  
 }  
  
 strcpy(fileNameTXT,IdLog);  
 strcat(fileNameTXT, ".txt");  
 fp = fopen(fileNameTXT, "r");  
 fscanf(fp, "%[^\n]", dumpTxt);  
 *if* (nama == 1){  
 printf(" %s\n", strtok(dumpTxt, " "));  
 }  
  
 *if* (saldo == 1){  
 printf(" Saldo: %s\n", saldoLog);  
 }  
 printf("\n");  
 }  
 fclose(fp);  
 fclose(fpointer);  
 printf("-----------------------------\n");  
 *return* 0;  
}

Pada fungsi tampilkan daftar akun ini bertujuan agar kita bisa melihat daftar-daftar akun yang ada dalam database, dalam menampilkan datanya ada tiga, yaitu ID, nama, dan saldo. Tidak semua informasi ditampilkan ke layar, kita bisa menampilkan namanya saja, dan lain-lain. Konsepnnya yaitu memakai switch menggunakan parameter. Jika parameter ID bernilai satu, maka ID akan ditampilkan, jika nama bernilai 1, maka nama akan ditampilkan, dan jika saldo bernilai 1, maka saldo akan ditampilkan. Apabila kita ingin menampilkan semua datanya kita hanya perlu membuat parameter tersebut bernilai 1 semua.

**Fungsi setorTunai()**

*int* setorTunai(*int* ID, *int* Nominal, *int* mode){ *// PAKAI INI UNTUK SETOR TUNAI  
 //mode : 0 = setor tunai, 1 = transfer masuk  
 int* nominalDump;  
 *char* fileTXT[15];  
  
 itoa(ID, fileTXT, 10);  
 strcat(fileTXT, ".txt");  
  
 FILE \*fpointer, \*fp;  
 fpointer = fopen("account.csv", "r");  
 fp = fopen("temp.csv", "w");  
  
 *char* dump[100], dump2[100], charNominal[100];  
 *char* \*token;  
 fgets(dump, *sizeof*(dump), fpointer);  
 fprintf(fp, dump);  
 *while* (fgets(dump, *sizeof*(dump), fpointer)){  
 strcpy(dump2, dump);  
 token = strtok(dump, ",");  
 *if* (atoi(token) == ID){  
 fprintf(fp, token);  
 fprintf(fp, ",");  
 token = strtok(**NULL**, ",");  
 nominalDump = atoi(token);  
 nominalDump = nominalDump + Nominal;  
 itoa(nominalDump, charNominal, 10);  
 fprintf(fp, charNominal);  
 fprintf(fp, ",");  
 token = strtok(**NULL**, ",");  
 fprintf(fp, token);  
 }*else*{  
 fprintf(fp, dump2);  
 }  
 }  
 fclose(fp);  
 fclose(fpointer);  
  
 fpointer = fopen("temp.csv", "r");  
 fp = fopen("account.csv", "w");  
  
 *while* (fgets(dump, *sizeof*(dump), fpointer)){  
 fprintf(fp, dump);  
 }  
  
 fclose(fp);  
 fclose(fpointer);  
 remove("temp.csv");  
  
 *char* tgl[20], waktu[20];  
  
 time\_t t = time(**NULL**);  
 *struct* tm tm = \*localtime(&t);  
 *char* Bulan[10],Hari[10],Jam[10],Menit[10],Detik[10];  
 itoa(tm.tm\_year + 1900, tgl,10); strcat(tgl, "/");  
 itoa(tm.tm\_mon + 1, Bulan, 10);  
 itoa(tm.tm\_mday, Hari, 10);  
 strcat(tgl,Bulan); strcat(tgl, "/");  
 strcat(tgl,Hari);  
  
 itoa(tm.tm\_hour, waktu, 10); strcat(waktu, ":");  
 itoa(tm.tm\_min, Menit, 10);  
 itoa(tm.tm\_sec, Detik, 10);  
 strcat(waktu,Menit); strcat(waktu, ":");  
 strcat(waktu,Detik);  
  
  
 fpointer = fopen(fileTXT, "a");  
 *if*(mode == 0)  
 fprintf(fpointer, "[+] SETOR TUNAI\n");  
 *else if*(mode == 1)  
 fprintf(fpointer, "[+] TRANSFER MASUK\n");  
  
 fprintf(fpointer, " TGL: %s\n", tgl);  
  
 fprintf(fpointer, " PUKUL: %s\n", waktu);  
 fprintf(fpointer, " NOMINAL: %d\n\n", Nominal);  
  
 fclose(fpointer);  
}

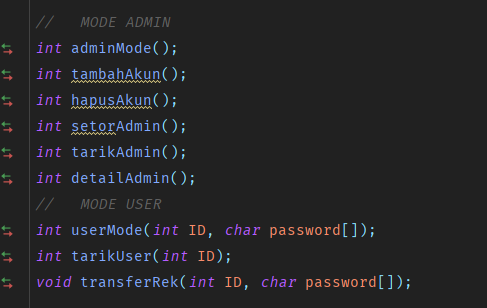
Fungsi setor tunai bertujuan untuk mengirim uang ke dalam akun. Konsepnya dalam fungsi ini membutuhkan tiga parameter untuk dijalankan yaitu ID, nominal dan mode. Untuk mode ini digunakan untuk identitas fungsi digunakan oleh mode apa, mode 0 adalah untuk setor tunai sedangkan mode 1 adalah setor tunai untuk transfer.

Fungsi ini sama seperti fungsi Tarik tunai. Hanya saja yang membedakan, fungsi Tarik tunai untuk mengurangi nominal yang ada dalam akun tersebut.

**Fungsi lihatDetailAkun()**

*void* lihatDetailAkun(*int* ID, *int* mode)  
{  
*// system("cls");  
 char* buff[100], txtFile[15];  
 itoa(ID, txtFile, 10);  
 strcat(txtFile, ".txt");  
 *//membuka akun file txt* FILE \*fptr = fopen(txtFile, "r");  
  
 *if* (fptr == **NULL**)  
 {  
 printf("Error, file %d.txt tidak ditemukan\n", ID);  
 exit(1);  
 }  
  
 printf(" PROGRAM KAS KELAS\n"  
 " LIHAT DETAIL AKUN\n");  
 *if* (mode == 0)  
 printf(" --ADMIN--\n");  
 *else* printf(" --USER--\n");  
 printf("INFORMASI AKUN :\n");  
  
 printf(" ID : %d\n", ID);  
 *for* (*int* i = 0; i < 4; i++)  
 {  
 fgets(buff, *sizeof*(buff), fptr);  
 printf("%s", buff);  
 }  
 *//mencari saldo id bersangkutan di account.csv* printf("Saldo : %d\n", getSaldo(ID));  
 *//mencetak keseluruhan isi file akun txt  
 while* (fgets(buff, *sizeof*(buff), fptr))  
 printf("%s", buff);  
  
 printf("\n----------------------------\n");  
 system("pause");  
 fclose(fptr);  
}

Fungsi lihat detail akun ini bertujuan untuk membaca file txt akun dan menampilkannya ke layar. Karna fungsi ini hanya untuk menampilkan saja, maka tidak ada mode write pada operasi filenya. Fungsi ini akan mencetak semua isi data dalam file txt akun tersebut.



Disini adalah beberapa fungsi yang dipakai untuk kedua mode, jadi program akan berpindah ke dalam fungsi-fungsi tersebut ketika memilih menu yang sesuai dengan fungsinya.

* 1. **Fungsi main()**

*int* main(){  
 *int* pilih, ID;  
 *char* password[100];  
 *char* \*token;  
  
 printf(" PROGRAM KAS KELAS \n");  
 printf("1. Admin\n");  
 printf("2. User\n");  
 printf("3. Keluar Program\n");  
 *while* (1){  
 printf("Pilih Menu:");  
 scanf("%d", &pilih);  
  
 *if* (pilih == 1){ *// 1. Admin  
 while* (1){  
 printf("Masukan ID:");  
 scanf("%d", &ID);  
 fflush(**stdin**);  
 printf("Masukan password:");  
 fgets(password, *sizeof*(password), **stdin**);  
  
 FILE \*fpointer;  
 fpointer = fopen("account.csv", "r");  
  
 *char* akun[100];  
 *while* (fgets(akun, *sizeof*(akun), fpointer)){  
 token = strtok(akun, ",");  
 *while* (token != **NULL**){  
 *if* ((ID == atoi(token)) && (ID / 100000 == 10)){  
 token = strtok(**NULL**, ",");  
 token = strtok(**NULL**, ",");  
 *if*(strcmp(token, password) == 0){  
 fclose(fpointer);  
 adminMode();  
 *return* 0;  
 }  
 }  
 token = strtok(**NULL**, ",");  
 }  
 }  
 *if* (token == **NULL**){  
 printf("ID atau Password salah\n");  
 }  
 }  
 }  
 *else if* (pilih == 2){ *// 2. User  
 while* (1){  
 printf("Masukan ID:");  
 scanf("%d", &ID);  
 fflush(**stdin**);  
 printf("Masukan password:");  
 fgets(password, *sizeof*(password), **stdin**);  
  
 FILE \*fpointer;  
 fpointer = fopen("account.csv", "r");  
  
 *char* akun[100];  
 *while* (fgets(akun, *sizeof*(akun), fpointer)){  
 token = strtok(akun, ",");  
 *while* (token != **NULL**){  
 *if* ((ID == atoi(token)) && (ID / 100000 == 20)){  
 token = strtok(**NULL**, ",");  
 token = strtok(**NULL**, ",");  
 *if*(strcmp(token, password) == 0){  
 fclose(fpointer);  
 userMode(ID, password);  
 *return* 0;  
 }  
 }  
 token = strtok(**NULL**, ",");  
 }  
 }  
 *if* (token == **NULL**){  
 printf("ID atau Password salah\n");  
 }  
 }  
 }  
 *else if* (pilih == 3){  
 exit(0);  
 } *else*{  
 printf("Input tidak ada dalam menu\n");  
 }  
 }  
 *return* 0;  
}

Pada fungsi main itu sendiri ketika program dijalankan pertama kali, ia akan meminta user untuk login, tetapi sebelum itu program akan menyuruh pengguna untuk memilih mode apa yang akan dijalankan, apakah itu user mode atau admin mode, perbedaannya adalah bahwa user mode memiliki keterbatasan fitur, sedangkan untuk admin mode memiliki banyak fitur. Jika pengguna memilih untuk masuk mode admin, maka program akan menyuruh pengguna untuk memasukan ID admin dan password admin, secara default ID dan password admin adalah, ID:1012345 dan pass: admin. Apabila salah aka nada peringatan bahwa ID atau password salah, begitupun dengan login user mode. Untuk melakukan pengecekan berdasarkan ID dahulu, untuk admin dan user dibedakan melalui dua angka didepanya, admin yaitu 10, user yaitu 20. Program akan membaca file csv secara berurutan dan mengambilan kolom pertama untuk ID dan kolom ketiga sebagai password akun tersebut. Apabila benar, program akan masuk ke fungsi admin/user sesuai dengan pilihan mode.

* 1. **Admin Mode**

*int* adminMode(){  
 *int* pilih;  
 *while* (1){  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 "1. TAMBAH AKUN\n"  
 "2. HAPUS AKUN\n"  
 "3. SETOR TUNAI\n"  
 "4. TARIK TUNAI\n"  
 "5. LIHAT DETAIL AKUN\n"  
 "6. exit\n");  
 printf("pilih:");  
 scanf("%d", &pilih);  
  
 *switch* (pilih) {  
 *case* 1:  
 tambahAkun();  
 *break*;  
 *case* 2:  
 hapusAkun();  
 *break*;  
 *case* 3:  
 setorAdmin();  
 *break*;  
 *case* 4:  
 tarikAdmin();  
 *break*;  
 *case* 5:  
 detailAdmin();  
 *break*;  
 *case* 6:  
 exit(0);  
 *default*:  
 printf("maaf tidak ada dalam menu\n");  
 }  
 }  
 *return* 0;  
}

Pada code di sini, program admin mode akan menampilkannya sebagai menu admin dan pengguna bisa memilih beberapa menu yang bisa dioperasikan sebagai admin. Seperti yang dikatakan di awal bahwa setiap menu akan masuk ke dalam fungsi-fungsi yang sudah dibuat sesuai kategorinya, di sini fungsi-fungsi yang dipakai sebagai admin mode adalah tambahAkun(), setorAdmin(), tarikAdmin(), detailAdmin(), dan exit(). Apabila user salah menginput menu, maka program akan meminta ulang inputan yang benar.

**Fungsi tambahAkun()**

*int* tambahAkun(){  
 *int* umur, pilKelamin, ID;  
 *char* nama[50], tanggal[12], jenisKelamin[12], password[100];  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 " TAMBAH AKUN\n");  
 printf("NAMA:");  
 fflush(**stdin**);  
 scanf("%s", nama);  
 fflush(**stdin**);  
 printf("UMUR:");  
 scanf("%d", &umur);  
 fflush(**stdin**);  
 printf("TANGGAL LAHIR (dd/mm/yy):");  
 fgets(tanggal, *sizeof*(tanggal), **stdin**);  
 fflush(**stdin**);  
 printf("JENIS KELAMIN:\n");  
 printf("1. Laki-laki | 2. Perempuan\n");  
 printf("pilih:");  
 scanf("%d", &pilKelamin);  
 *if*(pilKelamin == 1){  
 strcpy(jenisKelamin, "Laki-laki");  
 }  
 *else if* (pilKelamin = 2){  
 strcpy(jenisKelamin, "Perempuan");  
 }  
 printf("------------------------------------\n");  
 srand(time(0));  
 ID = (10\*200000)+(rand()%100000);  
 printf("ID: %d (ingat ID ini baik-baik)\n", ID);  
 printf("Masukan password:");  
 fflush(**stdin**);  
 fgets(password, *sizeof*(password), **stdin**);  
 *int* passRepeat[100];  
 *while* (1){  
 printf("Masukan kembali password:");  
 fgets(passRepeat, *sizeof*(passRepeat), **stdin**);  
 *if* (strcmp(passRepeat, password)==0){  
 printf("Pembuatan akun berhasil\n");  
 *break*;  
 }*else*{  
 printf("Password tidak sesuai dengan sebelumnya, masukan kembali\n");  
 }  
 }  
  
 *//Write to database BEGIN  
 char* txtFile[15];  
 itoa(ID, txtFile, 10);  
 strcat(txtFile, ".txt");  
  
 FILE \*fpointer;  
 fpointer = fopen(txtFile, "w");  
 fprintf(fpointer, " NAMA:%s\n"  
 " UMUR:%d\n"  
 " TANGGAL LAHIR:%s"  
 " KELAMIN:%s\n"  
 "------History Transaksi------\n",nama, umur, tanggal, jenisKelamin);  
 fclose(fpointer);  
  
 fpointer = fopen("account.csv", "a");  
 fprintf(fpointer, "%d,0,%s", ID, password);  
 fclose(fpointer);  
 *//Write to database END* system("pause");  
 *return* 0;  
}

Fungsi tambahAkun() adalah fungsi yang hanya dimiliki oleh mode admin. Di sini program akan menyimpan data-data ke dalam variabel yang independent, yaitu ada variabel

*int* umur, pilKelamin, ID;  
*char* nama[50], tanggal[12], jenisKelamin[12], password[100];

untuk nama, tanggal, jenisKelamin, dan umur akan ditulis ke dalam file .txt dan ID dan password akan dimasukan ke dalam file csv

Program akan meminta user pertama kali untuk menginputkan Nama, kemudian dilanjutkan dengan umur, tanggal lahir, jenis kelamin (untuk jenis kelamin hanya diberikan pilihan 1.Laki-laki. 2.Perempuan), dan terakhir adalah password yang diinputkan dua kali sebagai konfirmasi. ID tidak ditentukan oleh admin, tetapi akan digenerate oleh sistem di mana algoritma generatenya adalah di sini menggunakan random yang di modulus dengan 100000 untuk angka belakang dan ditambahkan dengan 2000000 sebagai angka depan 20 untuk ID user.

Disini program ketika user selesai menginputkan password, maka semua data yang sudah disimpan pada variabel akan dituliskan menggunakan operasi file yang pertama adalah menuliskan ke dalam file .txt dan nama filenya adalah ID user. Dan terakhir program akan menuliskan ID dan password ke dalam file csv, untuk saldo akan dituliskan 0 pada kolom kedua sebagai nominal default.

**Fungsi hapusAkun()**

*int* hapusAkun(){  
 *int* pilID;  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 " HAPUS AKUN\n"  
 "AKUN YANG TERDAFTAR\n");  
 *if* (TampilkanDaftarAkun(1,1,0))  
 {  
 *return* 0;  
 }*else*{  
 printf("PILIH ID:");  
 scanf("%d", &pilID);  
 }  
  
 FILE \*fpointer, \*fp;  
 fpointer = fopen("account.csv", "r");  
 fp = fopen("temp.csv", "w");  
  
 *char* dump[50];  
 *char* dump2[50];  
 *char* \*token;  
 fgets(dump, *sizeof*(dump), fpointer);  
 fprintf(fp,dump);  
 token = strtok(dump, ",");  
 *while* (fgets(dump, *sizeof*(dump), fpointer)){  
 strcpy(dump2, dump);  
 token = strtok(dump, ",");  
  
 *if* (atoi(token) != pilID){  
 fprintf(fp, dump2);  
 }  
  
 }  
 fclose(fp);  
 fclose(fpointer);  
  
 fpointer = fopen("temp.csv", "r");  
 fp = fopen("account.csv", "w");  
  
 *while* (fgets(dump, *sizeof*(dump), fpointer)){  
 fprintf(fp, dump);  
 }  
  
 fclose(fp);  
 fclose(fpointer);  
  
 remove("temp.csv");  
 *char* fileTarget[15];  
 itoa(pilID, fileTarget, 10);  
 strcat(fileTarget, ".txt");  
 remove(fileTarget);  
  
 printf("HAPUS AKUN BERHASIL\n");  
 system("pause");  
}

Fungsi hapusAkun() di sini juga adalah fungsi yang khusus dimiliki oleh mode admin.

Pada code ini program akan menampilkan semua daftar akun yang ada atau terdaftar, yang ditampilkan oleh fungsi univerisal yaitu fungsi

TampilkanDaftarAkun(1,1,0)

Yang akan ditampilkan di sini adalah semua akun user, admin tidak termasuk

Namun jika tidak ada akun user yang terdaftar atau kosong, maka program tidak akan bisa melakukan hapus akun, akan langsung Kembali (return) ke menu admin mode.

Jika ada program akan peminta pengguna untuk memasukan ID akun. Pencarian akun di sini menggunaka algoritma secara linear, artinya akan dicari satu persatu dari yang pertama sampai akhir akun yang terdaftar. Jika ditemukan maka program akan langsung menghapusnya. Untuk operasi filenya logikanya kita membuat sebuah file csv Salinan yang diberi nama dump.csv untuk menyalin semua data csv yang original. Kemudian file csv yang original akan ditulis ulang berdasarkan data di dump.csv kecuali akun yang dihapus.

**Fungsi setorAdmin()**

*int* setorAdmin(){  
 *int* plhID, nominal;  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 " SETOR TUNAI\n"  
 "AKUN YANG TERDAFTAR\n");  
 *if* (TampilkanDaftarAkun(1,1,1))  
 {  
 *return* 0;  
 }*else*{  
 printf("PILIH ID:");  
 scanf("%d", &plhID);  
 }  
  
 FILE \*fpointer;  
 fpointer = fopen("account.csv", "r");  
  
 *char* dump[100];  
 *char* \*token;  
 *while* (fgets(dump, *sizeof*(dump), fpointer)){  
 token = strtok(dump, ",");  
 *if* (atoi(token) == plhID){  
 printf("Masukan nominal:");  
 scanf("%d", &nominal);  
 setorTunai(plhID, nominal, 0);  
 printf("Setor tunai berhasil dilakukan\n");  
 system("pause");  
 fclose(fpointer);  
 *return* 0;  
 }  
 }  
  
 fclose(fpointer);  
 printf("Akun tidak ditemukan!!\n");  
 system("pause");  
}

Pada code ini, pertama sama dengan fungsi hapus akun, yaitu program akan menampilkan daftar akun yang ada, jika tidak ada akun user yang ada maka setor tunai tidak akan bisa dilakukan dan akan langsung return.

Jika ada, program akan meminta pengguana untuk input ID akun yang akan disetorkan. Pada pencarian akan dilakukan seperti hapus akun, yaitu pencarian secara linear. Jika ditemukan, maka program akan mengambil kolom kedua pada akun tersebut yang merupakan nominal, di awal program melakukan read untuk mengecek ID yang dicek di fungsi if sampai pada bari akhir file csv. Ketika sudah diapatkan, program akan menyuruh pengguan untuk input nominal berapa yang akan ditambahkan. Sesudah itu fungsi setor tunai akan dipanggil di mana fungsi ini adalah fungsi universal, data yang diminta sebagai parameter fungsi itu adalah ID dan nominal.

**Fungsi tarikAdmin()**

*int* tarikAdmin()  
{  
 *int* plhID, nominal;  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 " TARIK TUNAI\n"  
 "AKUN YANG TERDAFTAR\n");  
 *if* (TampilkanDaftarAkun(1,1,1))  
 {  
 *return* 0;  
 }*else*{  
 printf("PILIH ID:");  
 scanf("%d", &plhID);  
 }  
  
 FILE \*fpointer;  
 fpointer = fopen("account.csv", "r");  
  
 *char* dump[100];  
 *char* \*token;  
 *while* (fgets(dump, *sizeof*(dump), fpointer))  
 {  
 token = strtok(dump, ",");  
 *if* (atoi(token) == plhID)  
 {  
 printf("Masukan nominal:");  
 scanf("%d", &nominal);  
 *if* (tarikTunai(plhID, nominal, 0))  
 {  
 printf("Tarik tunai berhasil dilakukan\n");  
 }  
 *else* {  
 printf("Tarik tunai gagal karena saldo tidak mencukupi\n");  
 }  
 system("pause");  
 fclose(fpointer);  
 *return* 0;  
 }  
 }  
  
 fclose(fpointer);  
 printf("Akun tidak ditemukan!!\n");  
 system("pause");  
}

Pada code, program akan menampilkan daftar akun user yang akan dilakukan Tarik tunai, apabila tidak ada maka Tarik tunai tidak bisa dilakukan.

Algoritmanya hampir mirip dengan fungsi setor tunai hanya yang membedakannya adalah pada fungsi Tarik tunai pada I kondisional if, di sana program akan memanggil fungsi universal tarikTunai(); jika fungsi itu berhasil dilakukan makan akan mereturn hasil 1 (true) dan akan printf bahwa Tarik tunai berhasil dilakukan, tetapi apabila nominalnya tidak mencukupi, fungsi Tarik tunai akan meretun 0 (false) dan diberitahukan bahwa transaksi gagal dilakukan.

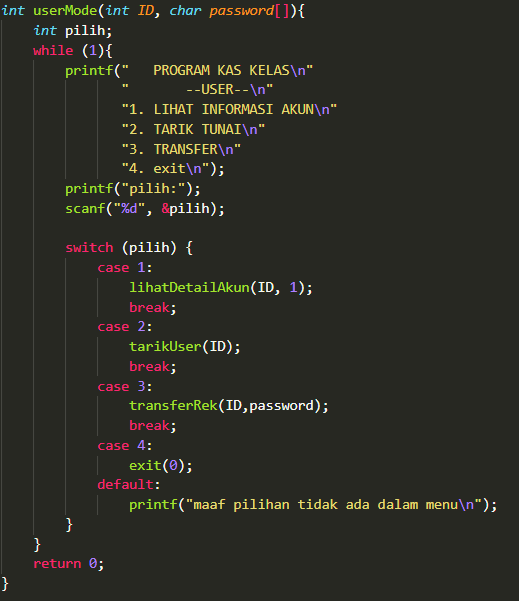
**Fungsi detailAdmin()**

*int* detailAdmin(){  
 *int* plhID, nominal;  
 printf(" PROGRAM KAS KELAS\n"  
 " --ADMIN--\n"  
 " TARIK TUNAI\n"  
 "AKUN YANG TERDAFTAR\n");  
 *if* (TampilkanDaftarAkun(1,1,0))  
 {  
 *return* 0;  
 }*else*{  
 printf("PILIH ID:");  
 scanf("%d", &plhID);  
 }  
 lihatDetailAkun(plhID, 0);  
}

Di sini program akan pertama kali akan menampilkan daftar akun yang akan dilihat detail data-data akun user tersebut, apabila tidak ada akun user sama sekali, maka program akan return Kembali ke menu admin mode.

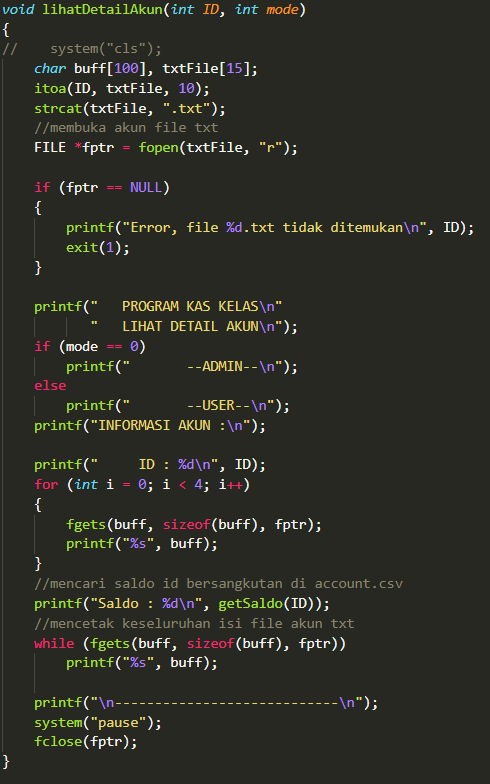
Namun jika ada, program meminta pengguna untuk menginputkan ID akun apa yang ingin dilihat. Lihat detail akun akan dilakukan oleh fungsi universal yaitu fungsi lihatDetailAkun(); dan ada dua parameter untuk dikirim sebagai data yang akan dipakai dalam fungsi tersebut, yaitu ID dan mode yang merupakan Boolean/integer sebagai switch dimana 0 untuk admin mode, 1 untuk user mode.

* 1. **User Mode**



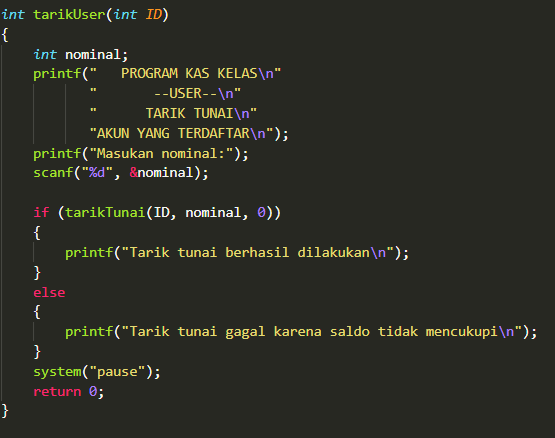
Pada fungsi user mode, user akan disajikan daftar menu yang dapat diakses sebagai seorang user. User dapat memilih dari menu 1 hingga menu 4, jika menu yang dimasukkan tidak tersedia maka user akan mendapat pesan bahwa menu yang dimasukkan tidak tersedia. Masing-masing menu didukung oleh sebuah fungsi yang akan menjalankan tugas tertentu sesuai dengan menu yang dipilih. Fungsi tersebut yaitu fungsi lihatDetailAkun(), tarikUser(), transferRek(), dan exit() untuk keluar.

**Fungsi lihatDetailAkun()**



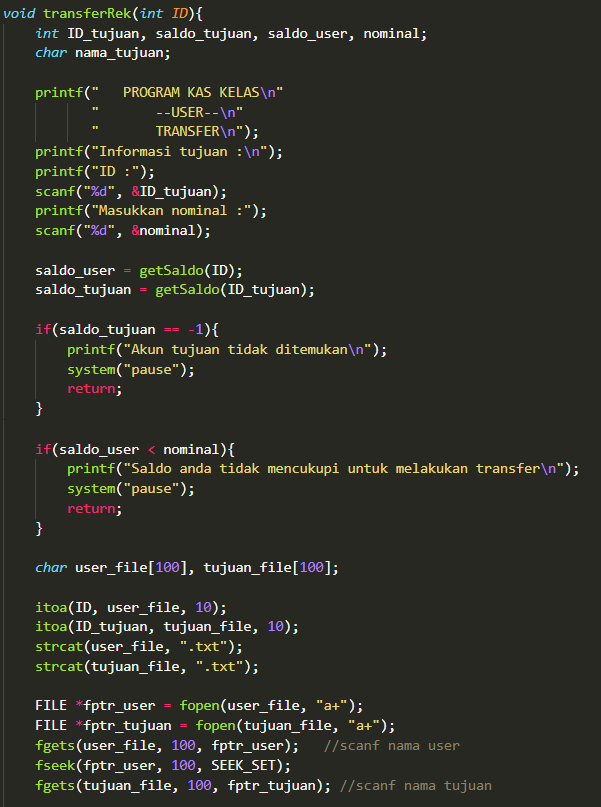
Fungsi lihatDetailAkun memiliki parameter yang diterima melalui call by value yaitu ID dari user dan sebuah mode. Mode disini hanya digunakan untuk mengatur tampilan karena fungsi ini merupakan fungsi yang akan digunakan pada user mode dan admin mode. File dari user bersangkutan akan dibuka. Jika file tidak ada maka akan ditampilkan pesan bahwa file tidak ditemukan. Jika file berhasil dibuka maka seluruh isi dari file tersebut yang berisi detail identitas user dan histori transaksi yang terjadi akan ditampilkan secara keseluruhan.

**Fungsi tarikUser()**



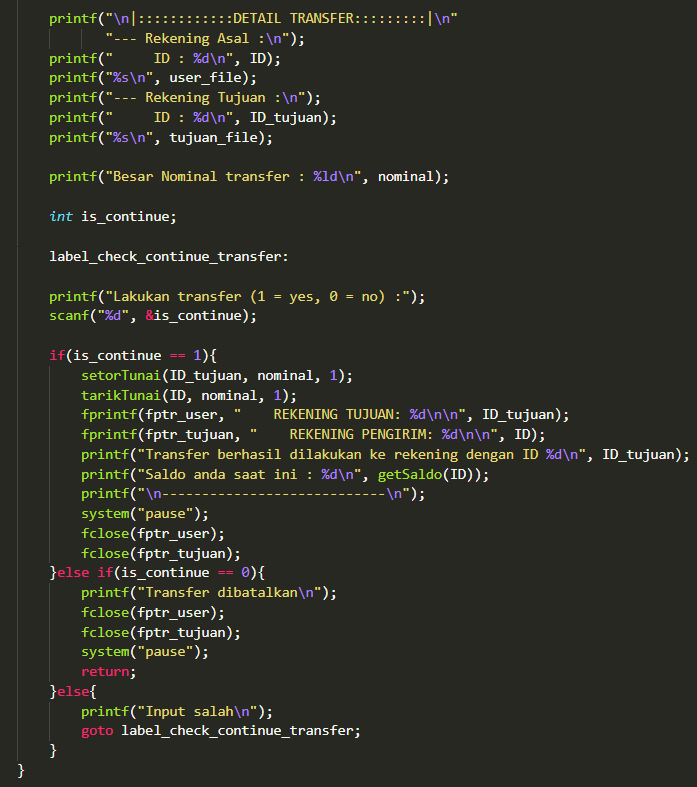
Fungsi tarikUser dideklarasikan sebagai fungsi yang menerima parameter dengan call by value. Parameter yang diterima berupa ID dari user. Pada fungsi ini terlebih dahulu user akan diminta untuk memasukkan besarnya nominal penarikan yang akan dilakukan. Fungsi universal tarikTunai akan dipanggil dengan call by value antara lain ID, nominal, dan sebuah mode yaitu 0 sebagai tanda bahwa akan dilakukan tarik tunai. Mode ini diperlukan karena fungsi universal tarikTunai juga digunakan untuk melakukan transfer.

**Fungsi transferRek()**

­­ 

Menu transfer merupakan fasilitas khusus yang diberikan untuk user yang memungkinkan user melakukan transfer ke akun lain yang terdaftar. Parameter yang diterima oleh fungsi ini yaitu ID dari user. User akan diminta untuk memasukkan informasi tujuan transfer dimana berupa ID dari akun tujuan. Selanjutnya user juga akan diminta untuk memasukkan nominal besarnya transfer yang ingin dilakukan. Saldo user dan saldo id tujuan akan dibaca terlebih dahulu menggunakan fungsi universal getSaldo().

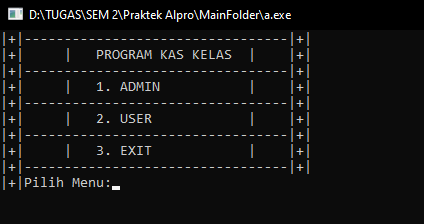
Jika fungsi getSaldo mengembalikan nilai -1 berarti akun tujuan tidak ditemukan dan user akan nmendapat pesan bahwa akun tidak ditemukan. Jika saldo user kurang dari nominal transfer maka pesan saldo tidak mencukupi akan ditampilkan.



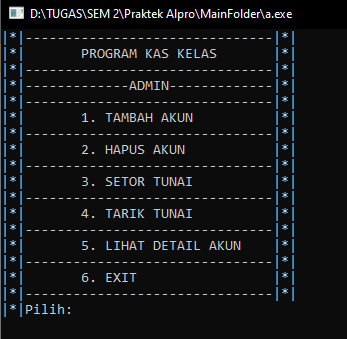
Untuk keperluan tampilan, identitas nama dari user dan ID tujuan akan ditampilkan. Nama ini diambil dari file txt detail akun bersangkutan. Setelah detail transfer ditampilkan ke user, user kemudian diberikan pilihan apakah akan melanjutkan untuk melakukan transfer. Jika user memilih untuk melanjutkan transfer maka fungsi universal tarikTunai akan dilakukan untuk akun user dan fungsi setorTunai akan dilakukan untuk akun tujuan transfer. Jika user memilih untuk tidak melanjutkan transfer maka transfer yang sebelumnya dipersiapkan tidak dilanjutkan.

* 1. **Output Program**

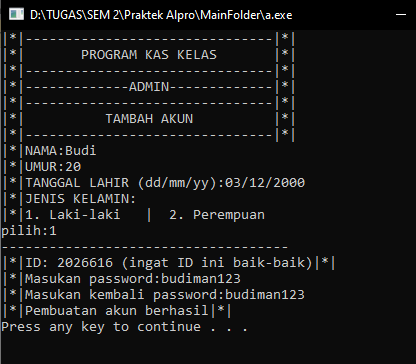
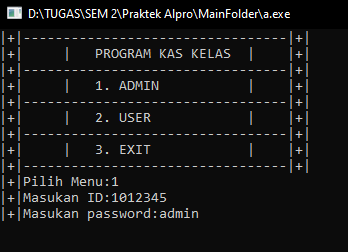
**Tampilan awal program**



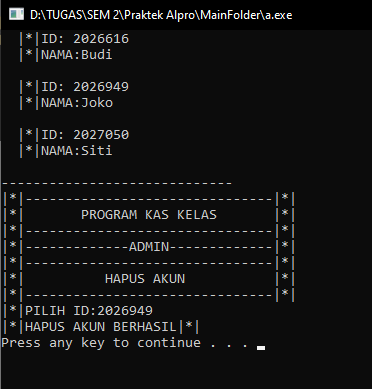
**Mode Admin**



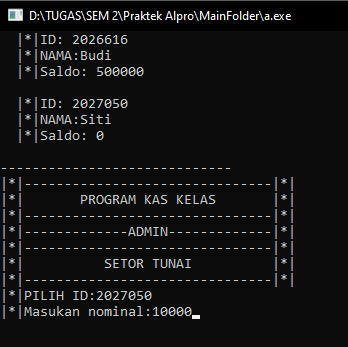
1. **Tambah Akun (Mode Admin)**

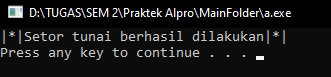


1. **Hapus Akun (Mode Admin)**

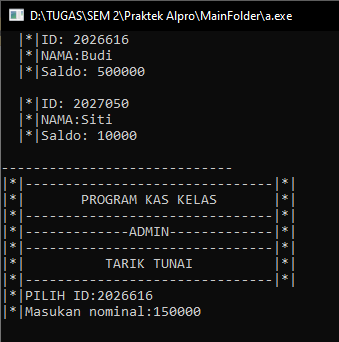


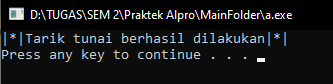
1. **Setor tunai (Mode Admin)**



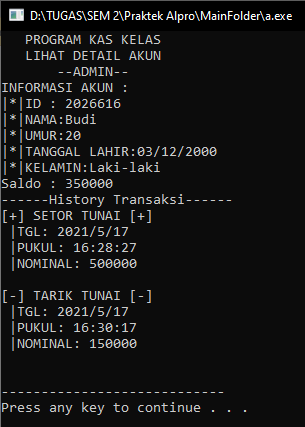


1. **Tarik tunai (Mode Admin)**

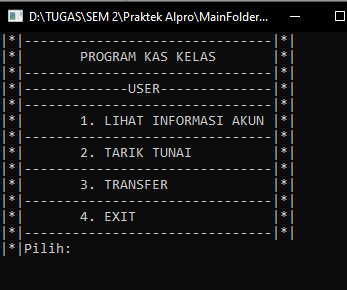
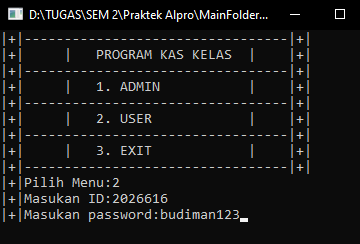




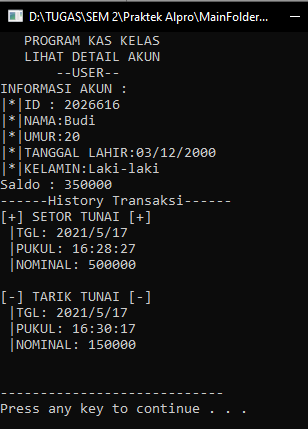
1. **Lihat detail akun (Mode Admin)**



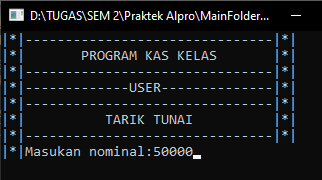
**Mode User**

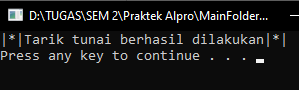


1. **Lihat informasi akun (Mode User)**

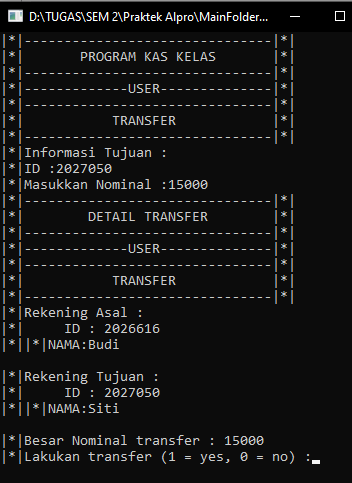


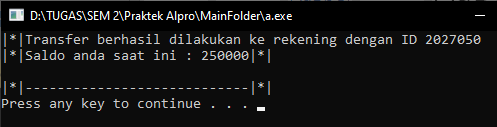
1. **Tarik tunai (mode user)**





1. **Transfer tunai (Mode User)**





**BAB V**

**PENUTUP**

* 1. **Kesimpulan**

Berdasarkan atas apa yang telah diuruaikan pada bab sebelumnya dapat ditarik kesimpulan bahwa program kas kelas ini dapat membantu menyelesaikan masalah yang terjadi pada kegiatan kas manual seperti yang disebut kan pada latar belakang dimana program ini dapat memenuhi aktivitas pencatatan kas yang baik seperti pencatatan kas yang rinci mengenai kondisi keuangan dan kestabilan dalam kondisi yang lama berbeda dari kas manual yang kondisi nya bisa berubah sewaktu waktu.Program ini dibuat melalui segala materi yang telah kami terima antara lain:Operasi File,Perulangan,Function,dll sehingga kami dapat membuat program ini agar dapat memenuhi masalah yang terjadi pada kas manual agar memudahkan aktivitas kebedaharaan kelas.

* 1. **Saran**

Program ini diharapkan dapat membantu dalam menyelesaikan masalah yang terjadi dan dapat terus digunakan sehingga program ini dapat berguna untuk kegiatan keuangan kelas. Semoga dengan ini kedepannya akan menjadi lebih baik dan tidak terjadi masalah.

**DAFTAR PUSTAKA**

Anonim. 2020. *Relational Database from CSV Files in C*. <https://www.geeksforgeeks.org/relational-database-from-csv-files-in-c/>. Diakses pada 17 Mei 2021 pukul 20.30.

Nursoyo, Yulia Hadi. 2018. *Aplikasi Pengelolaan Kas Berbasis Web di Himpunan Mahasiswa Teknik Informatika dengan Metode Perancangan Berorientasi Objek*. Fakultas Sains dan Teknologi, Universitas Muhammadiyah Sukabumi.

Solichin, Achmad. 2009. *Buku Gratis : Pemrograman Bahasa C dengan Turbo C*. <https://achmatim.net/2009/12/10/buku-gratis-pemrograman-bahasa-c-dengan-turbo-c/>. Diakses pada 18 Mei 2021 pukul 11.30.