**TUGAS FINAL PROJECT**

**PRAKTIKUM ALGORITMA DAN PEMROGRAMAN**



**DISUSUN OLEH**

**IDA AYU TARIA PUTRI MAHADEWI 2008561081**

**I PUTU GEDE MAYSA PUTRA 2008561086**

**I DEWA AGUNG ADWITYA PRAWANGSA 2008561091**

**DEWA NYOMAN AGUNG ADIPURWA M. 2008561096**

**ANAK AGUNG SAGUNG PRAMI APSARI K. 2008561101**

**I MADE KRISNA DWIPA JAYA 2008561106**

**IDA AYU BINTANG KARTIKA MAHARANI 2008561111**

**KELAS E**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**2021**

# KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Tuhan Yang Maha Esa, karena atas rahmat dan hidayah-Nya kami dapat menyelesaikan laporan ini tepat pada waktunya. Ucapan terimakasih juga kami haturkan kepada bapak dosen yang telah membimbing kami, sehingga saya dapat menyelesaikan Tugas Final Project Algoritma dan Pemrograman.

Kami menyadari dalam laporan ini masih banyak kesalahan dan kekurangan, hal ini disebabkan terbatasnya kemampuan pengetahuan dan pengalaman yang kami miliki, namun demikian banyak pula pihak yang telah membantu kami dengan menyediakan sumber informasi, memberikan masukan pemikiran, oleh karena itu kami mengharapkan kritik dan saran demi perbaikan dan kesempurnaan laporan ini diwaktu yang akan datang, semoga laporan ini dapat bermanfaat bagi kami dan orang banyak supaya mengetahui apa-apa yang ada dalam Tugas Final Project Algoritma dan Pemrograman.

Bali, 30 Mei 2021

Penulis

# DAFTAR ISI

[KATA PENGANTAR i](#_Toc73307715)

[DAFTAR ISI ii](#_Toc73307716)

[BAB I PENDAHULUAN 1](#_Toc73307717)

[1.1 Latar belakang 1](#_Toc73307718)

[1.2 Rumusan Masalah 2](#_Toc73307719)

[1.3 Tujuan Penelitian 2](#_Toc73307720)

[1.4 Manfaat Penelitian 2](#_Toc73307721)

[BAB II LANDASAN TEORI 3](#_Toc73307722)

[2.1 Struktur Dasar 3](#_Toc73307723)

[2.1.1 Tipe data 3](#_Toc73307724)

[2.1.2 Konstanta 3](#_Toc73307725)

[2.1.3 Variabel 4](#_Toc73307726)

[2.1.4 Deklarasi 4](#_Toc73307727)

[2.1.5 Operator 5](#_Toc73307728)

[2.1.6 Komentar 7](#_Toc73307729)

[2.1.7 Input 7](#_Toc73307730)

[2.1.8 Output 8](#_Toc73307731)

[2.2 Perulangan dan Penyeleksian Kondisi 8](#_Toc73307732)

[2.2.1 Perulangan 8](#_Toc73307733)

[2.2.2 Penyeleksian Kondisi 9](#_Toc73307734)

[2.3 Array 11](#_Toc73307735)

[2.4 Pointer 12](#_Toc73307736)

[2.5 Struct 13](#_Toc73307737)

[2.6 Operasi File 14](#_Toc73307738)

[2.7 CRUD 17](#_Toc73307739)

[2.8 Parameter 18](#_Toc73307740)

[BAB III DESAIN DAN METODE 21](#_Toc73307741)

[3.1 Desain Flowchart 21](#_Toc73307742)

[3.2 Metode 22](#_Toc73307743)

[BAB IV HASIL DAN IMPLEMENTASI 24](#_Toc73307744)

[4.1 Source Code 24](#_Toc73307745)

[4.2 Output Program 51](#_Toc73307746)

[BAB V PENUTUP 63](#_Toc73307747)

[5.1 Kesimpulan 63](#_Toc73307748)

[5.2 Saran 63](#_Toc73307749)

[DAFTAR PUSTAKA 64](#_Toc73307750)

# BAB I

# PENDAHULUAN

## Latar belakang

Menghadapi suatu perkembangan zaman bukanlah hal yang mudah, diperlukannya pengetahuan akan teknologi agar dapat bersaing pada era *society* 5.0. Bertambahnya usia para pekerja membuatnya kesusahan dalam menyelesaikan pekerjaannya. Membuat program merupakan salah satu cara memudahkan para pakerja menyelesaikan pekerjaannya. Menurut Anwar Harjono, program adalah urutan instruksi untuk menjalankan suatu komputasi (Kawengian, 2017). Beberapa contoh program yang sering dijumpai dalam kehidupan sehari-hari yaitu kalkulator, mesin kasir, dan pendataan dalam sekolah ataupun perpustakaan.

Dalam perpustakaan terdapat suatu program untuk membantu pekerjaan pustakawan yang disebut dengan *Library Simulator*. Program ini berisikan beberapa menu yaitu peminjaman buku, inventaris buku, sistem keanggotaan, dan pengembalian buku. Menu peminjaman buku digunakan untuk mencatat data peminjaman terkait buku yang dipinjam, identitas peminjam, tanggal pinjam dan pengembaliannya. Menu inventaris buku digunakan untuk memeriksa ketersediaan buku dan menyimpan data buku baru. Menu sistem keanggotaan untuk membuat, memperbaharui, dan menghapus data terkait keanggotaan, serta menu terakhir yaitu pengembalian buku untuk mencatat data pengembalian buku sesuai dengan tenggat waktu yang ditentukan atau tidak.

Adanya program *Library Simulator* dapat membantu pustakawan dalam hal mencatat buku maupun keanggotaan menjadi lebih efisien baik dari segi waktu maupun penghematan kertas. Program ini juga menyimpan secara digital, tidak seperti buku tulis yang menyimpan data secara konvensional yang dapat hilang secara fisik. Dibalik kelebihan yang dimiliki oleh sebuah program, terdapat juga kekurangannya yaitu memerlukan biaya yang lebih mahal dan perlunya waktu untuk belajar

cara menggunakan program tersebut. Jika dilihat dari kekurangan dan kelebihannya, program lebih optimal digunakan untuk pemakaian yang lama, sedangkan buku untuk pemakaian yang singkat. Jadi, sangat penting bagi pustakawan untuk memelajari cara menggunakan program ini agar dapat menyimpan data secara aman dalam jangka waktu yang lama.

## Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dibuat suatu rumusan masalah sebagai berikut :

1. Bagaimana desain pada program *Library Simulator*?
2. Bagaimana algoritma pada program *Library Simulator?*
3. Bagaimana *source code* pada program *Library Simulator*?
4. Bagaimana tampilan pada program *Library Simulator*?

## Tujuan Penelitian

Sesuai dengan rumusan masalah, maka dapat ditentukan tujuan penelitian sebagai berikut :

1. Untuk mengetahui desain pada program *Library Simulator*
2. Untuk mengetahui algoritma pada program *Library Simulator*
3. Untuk mengetahui *source code* pada program *Library Simulator*
4. Untuk mengetahui tampilan pada program *Library Simulator*

## Manfaat Penelitian

Adapun manfaat dari penulisan laporan ini adalah sebagai berikut :

1. Dapat memberikan gambaran terkait pembuatan program *Library Simulator*.
2. Sebagai bahan referensi untuk pembuatan program *Library Simulator*.

# BAB II

# LANDASAN TEORI

## Struktur Dasar

Bahasa Pemrograman C merupakan sebuah bahasa pemrograman komputer yang biasa digunakan untuk membuat berbagai aplikasi (general-purpose programming language), mulai dari sistem operasi (seperti Windows atau Linux, Mac), antivirus, software pengolah gambar (image processing), hingga compiler untuk bahasa pemrograman, dimana C banyak digunakan untuk membuat bahasa pemrograman lain yang salah satunya adalah PH. Meskipun termasuk general- purpose programming language, yaitu bahasa pemrograman yang bisa membuat berbagai aplikasi, bahasa pemrograman C paling cocok merancang aplikasi yang berhubungan langsung dengan Sistem Operasi dan hardware. Ini tidak terlepas dari tujuan awal bahasa C dikembangkan.

### **Tipe data**

Tipe data merupakan jenis data yang dapat diolah oleh komputer untuk memenuhi kebutuhan dalam pemrograman computer. Setiap variabel atau konstanta yang ada dalam kode program, sebaiknya di tentukan dengan pasti tipe datanya. Adapun tipe data dasar bahasa C yaitu :

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipe Data** | **Ukuran Memori** | **Format** | **Jangkauan Nilai** |
| Char | 1 byte | %c | -128 higga + 127 |
| Int | 4 byte | %i , %d | -32768 hingga 32767 |
| Float | 4 byte | %f | -3.4E-38 hingga 3.4E +38 |
| Double | 8 byte | %lf | -1.7E-308 hingga 1.7+308 |
| Void | 0 byte | - | - |

### **Konstanta**

Konstanta merupakan suatu tempat di dalam memori yang dibuat untuk menyimpan suatu nilai. Nilai yang disimpan dalam konstanta

bersifat tetap (konstan). Dari awal program hingga akhir program, nilai konstanta tetap dan tidak berubah-ubah. Bahasa C menyediakan beberapa karakter khusus yang disebut dengan karakter escape, diantaranya :

\a : untuk bunyi bell (alert)

\b : mundur satu spasi (backspace)

\f : ganti halaman (form feed)

\n : ganti baris baru (new line)

\r : ke kolom pertama, baris yang sama (carriage return)

\v : tabulasi vertical

\’ : karakter petik tunggal

\” : karakter petik ganda

\\ : karakter garis miring

\0 : nilai kosong (null)

### **Variabel**

Variabel merupakan ariabel merupakan penanda identitas yang digunakan untuk menampung suatu nilai. Nilai tersebut dapat diubah sepanjang kode program. Secara teknis, variabel merujuk kepada suatu alamat di memory komputer. Setiap variabel memilki nama sebagai identitas untuk variabel tersebut. Variabel bisa terdiri dari huruf, angka dan karakter underscore( \_ ).

### **Deklarasi**

Deklarasi dalam bahasa C dapat digambarkan sebagai atribut simbol atau pengenal. Deklarasi berupa nama jenis variabel, fungsi atau kelas.

Aturan deklarasi, yaitu :

* Variabel harus dimulai dengan alfabet atau garis bawah
* Variabel harus di deklarasi pada sisi kiri operator
* Nama variabel tidak boleh menggunakan kata kunci “ for”, “switch”, dan “case”
* Diperbolehkan menggunakan huruf, angka dan garis bawah

### **Operator**

Operator merupakan simbol atau karakter yang digunakan oleh program untuk melakukan sebuah operasi bilangan dan operasi string. untuk melakukan hal tersebut tentu kita harus mengetahui operator-operator yang digunakan dalam suatu bahasa pemrograman tertentu. Adapun operator- operator yang ada yaitu :

1. Operator Penugasan

|  |  |  |
| --- | --- | --- |
| **Operator** | **Sama Dengan** | **Contoh** |
| = | a = b | a = b |
| += | a = a + b | a += b |
| -= | a = a - b | a -= b |
| \*= | a = a \* b | a\*= b |
| /= | a = a / b | a /= b |
| %= | a = a % b | a %= b |

1. Operator Aritmatika

|  |  |
| --- | --- |
| **Operator** | **Artinya** |
| \* | Perkalian |
| + | Pertambahan |
| - | Pengurangan |
| / | Pembagian |
| % | Sisa Pembagian (Modulus) |

1. Operator Hubungan

|  |  |
| --- | --- |
| **Operator** | **Artinya** |
| < | Kurang dari |
| <= | Kurang dari sama dengan |
| > | Lebih dari |
| >= | Lebih dari sama dengan |
| != | Tidak sama dengan |
| == | Sama dengan |

1. Operator Unary

|  |  |  |
| --- | --- | --- |
| **Operator** | **Arti** | **Contoh** |
| - | Unary minus | a + -b\*c |
| -- | Penurunan dengan penambahan nilai 1 | a-- |
| ++ | Peningkatan dengan penambahan nilai 1 | a++ |
| ! | Unary not | !a |
| ~ | Bitwise not | ~a |
| \* | Menghasilkan nilai dari pointer | \*a |
| & | Menghasilkan alamat memori operand | &a |
| Sizeof | Ukuran dari operand dalam byte | sizeof(I) |

1. Operator Logika

|  |  |
| --- | --- |
| **Operator** | **Artinya** |
| && | Logika AND |
| || | Logika OR |
| ! | Logika NOT |

1. Operator Bitwise

|  |  |
| --- | --- |
| **Operator** | **Artinya** |
| >> | Operator bit ke kanan |
| & | Bitwise AND |
|  | Bitwise XOR |
| | | Bitwise OR |
| ~ | Bitwise NOT |

### **Komentar**

Komentar digunakan untuk memudahkan pembacaan dan pemahaman dari suatu program atau digunakan untuk dokumentasi program. Komentar merupakan keterangan program atau penjelasan program. Untuk memberikan komentar program bisa menggunakan pembatas /\* dan \*/ . untuk komentar yang hanya terdiri dari satu baris dapat menggunakan tanda // . komentar program tidak diproses dalam program.

### **Input**

Sebuah program mempunyai proses memasukkan data. Untuk program bahasa C, pemasukkan data dilakukan dengan menggunakan fungsi scanf(). Perlu diperhatikan dalam menginput data yaitu menentukan format tipe data yang ingin diinputkan. Adapun format tipe data yang digunakan yaitu :

|  |  |
| --- | --- |
| **Format Tipe Data** | **Kegunaan** |
| %c | Untuk inputan tipe data char |
| %i atau %d | Untuk inputan tipe data integer |
| %f | Untuk inputan tipe data float |
| %o | Untuk inputan data angka berbasis oktal |
| %u | Untuk inputan tipe data unsigned integer |
| %s | Untuk inputan tipe data string |
| %x | Untuk inputan data angka berbasis hexadesimal |

scanf() mempunyai kelemahan yaitu menginputkan data string. apabila data string yang diinputkan mengandung spasi maka data setelah spasi tidak tersimpan. Untuk mengatasi kelemahan ini bisa menggunakan fungsi fgets(). Adapun fungsi yang bisa digunakan untuk menginputkan data karakter selain fungsi scanf() dan fungsi gets(), yaitu :

* getch() yaitu untuk membaca tipe karakter tanpa menampilkan karakter tersebut ke layar.
* getche() yaitu untuk membaca karakter dengan menampilkan karakter yang diinputkan ke layar.

### **Output**

Output digunakan untuk mengeluarkan hasil proses komputer sehingga bisa dibaca oleh user. Salah satu fungsi untuk menampilkan output adalah fungsi printf(). Printf() digunakan untuk mencetak data berupa teks, numerik, variabel dan konstanta. Selain printf() Adapun perintah untuk menampilkan output yaitu :

* puts() yaitu untuk menampilkan data string secara otomatis yang akan diakhiri dengan perpindahan baris.
* putchar() yaitu untuk menampilkan suatu karakter.

## Perulangan dan Penyeleksian Kondisi

### **Perulangan**

Perulangan atau yang sering disebut dengan “looping”, merupakan proses yang dilakukan secara berulang-ulang dalam batas yang telah ditentukan. Dengan menggunakan proses perulangan, penulisan kode program dapat menjadi lebih efisien (Jeklin Harefa, 2017). Dalam bahasa C, terdapat beberapa macam struktur kendali perulangan, antara lain: for, do…while, dan while.

1. **Perulangan For**

Perulangan for adalah perulangan yang simple dan banyak digunakan karena strukturnya yang mudah dipahami, berikut adalah cara untuk mendeklarasikan perulangan for :

for(kondisi awal; syarat kondisi; iterasi)

{

pernyataan atau statement;

}

1. **Perulangan While**

Dalam perulangan while, pada prosesnya harus membuat variabel dengan nilai sebagai kondisi awal, lalu memerlukan syarat kondisi terlebih dahulu untuk melakukan looping dan diakhir kondisi diberikan iterasi.

Kondisi awal :

while(syarat kondisi)

{

statement;

iterasi;

}

1. **Perulangan Do-While**

Sedangkan pada perulangan Do-While, perulangan ini melakukan terlebih dahulu perulangan lalu membaca syarat kondisinya, perhatikan struktur Do-While berikut

Kondisi awal :

do

{

statement;

iterasi;

}

while(syarat kondisi);

### **Penyeleksian Kondisi**

Pada umumnya satu permasalahan yang kompleks memerlukan suatu penyeleksian kondisi. Dengan menyeleksi suatu kondisi, maka selanjutnya dapat ditentukan tindakan apa yang harus dilakukan, tergantung pada hasil kondisi penyeleksian. Jadi suatu aksi hanya dikerjakan apabila persyaratan atau kondisi tertentu terpenuhi. Penyeleksian kondisi digunakan untuk mengarahkan perjalanan suatu proses. Fungsi penyeleksian kondisi penting dalam bahasa C, terutama untuk program yang kompleks. Penyeleksian kondisi ini memiliki beberapa metode dalam pengaplikasianya di pemrograman, diantaranya yaitu :

1. If

Struktur if dibentuk dari pernyataan if dan sering digunakan untuk menyeleksi suatu kondisi tunggal. Bila proses yang diseleksi terpenuhi atau bernilai benar, maka pernyataan yang ada di dalam blok if akan diproses dan dikerjakan. Bentuk umum struktur kondisi if adalah :

if(kondisi)

pernyataan;

1. If Else

Dalam struktur kondisi if else minimal terdapat dua pernyataan. Jika kondisi yang diperiksa bernilai benar atau terpenuhi maka pernyataan pertama yang dilaksanakan dan jika kondisi yang diperiksa bernilai salah maka pernyataan yang kedua yang dilaksanakan. Jika memiliki lebih dari dua pernyataan, maka akan digunakannya if else if. Bentuk umumnya adalah sebagai berikut :

if(kondisi)

pernyataan;

else if(kondisi)

pernyataan;

else

pernyataan;

1. Nested If

Nested if atau if bersarang merupakan suatu kondisi yang didalamnya terdapat kondisi. Hal ini biasanya terjadi ketika terdapat beberapa kondisi yang didalamnya juga memungkinkan untuk terjadinya beberapa kondisi. Bentuk umum dari if bersarang adalah :

if(kondisi)

{

if(kondisi){

pernyataan;

}else if(kondisi){

pernyataan;

}else

pernyataan;

}

1. Ternary Operator

Operator ternary ini mirip dengan if namun penulisannya berbeda. Penulisan yang lebih singkat dari metode if yang membuat operator ini digunakan di beberapa kondisi. Namun, operator ini hanya dapat membuat dua kondisi yaitu ketika kondisi terpenuhi (*true)* dan kondisi tidak terpenuhi (*false*). Bentuk umum dari operator ini yaitu :

kondisi ? true : false;

1. Switch Case

Struktur kondisi switch....case....default digunakan untuk penyeleksian kondisi dengan kemungkinan yang terjadi cukup banyak. Struktur ini akan melaksanakan salah satu dari beberapa pernyataan ‘case’ tergantung nilai kondisi yang ada di dalam switch. Selanjutnya proses diteruskan hingga ditemukan pernyataan ‘break’. Jika tidak ada nilai pada case yang sesuai dengan nilai kondisi, maka proses akan diteruskan kepada pernyataan yang ada di bawah ‘default’. Bentuk umum dari struktur kondisi ini adalah :

switch(kondisi)

{

case 1 : pernyataan-1;

break;

case 2 : pernyataan-2;

break;

default : pernyataan-n

}

## Array

Array adalah semacam struktur data yang bisa menyimpan kumpulan elemen berurutan dengan ukuran tetap. Array biasanya digunakan untuk menyimpan kumpulan data, tetapi seringkali lebih berguna untuk menganggap array sebagai kumpulan variabel dari tipe yang sama. Alih-alih mendeklarasikan variabel individual, seperti number0, number1, …, dan number99, mendeklarasikan satu variabel array seperti angka dan menggunakan angka [0], angka [1], dan …, angka [99] untuk mewakili variabel individual. Elemen spesifik dalam array diakses oleh indeks. Semua array terdiri dari lokasi memori yang berdekatan. Sebuah array dapat dideklarasikan dalam bentuk satu dimensi, dua, bahkan n dimensi tergantung pada masalah dan cara penyelesaiaan anda. untuk mendeklarasikan sebuah array dapat dilihat seperti contoh berikut :

tipe\_array nama\_array[];

mula-mula definisikan tipe dari array, kemudian beri nama dan terakhir jangan lupa tambahkan tanda [] (siku).

Untuk inisialisasi isi array dapat dilakukan dengan dua cara, yang pertama adalah sebagai berikut :

int JumlahBarang[5] = {1, 2, 3, 4, 5};

dengan cara ini isi dari array secara langsung diinisialisasi. Berikutnya dengan cara memasukkannya satu persatu dengan menggunakan indeks sebagai berikut :

int jumlahBarang[5];

jumlahBarang[0] = 1;

jumlahBarang[1] = 2;

jumlahBarang[2] = 3;

jumlahBarang[3] = 4;

jumlahBarang[4] = 5;

untuk mengakses isi dari array kita dapat menggunakan perulangan (for loop), contohnya sebagai berikut :

for(int i = 0; i < size ; i++){

printf("%d ", jumlahBarang[i]);

}

## Pointer

Pointer di dalam bahasa C cukup mudah dan menyenangkan untuk dipelajari. Di dalam pemrograman C ada beberapa tugas yang apabila dilakukan dengan pointer akan lebih mudah, contohnya alokasi memori dinamis (Dynamic memory allocation) yang tidak dapat dilakukan tanpa pointer. Jadi sangat penting untuk mempelajari pointer agar menjadi seorang programmer C yang sesungguhnya. Setiap variabel itu memiliki lokasi memorinya masing-masing dan setiap memori memiliki alamat yang ditentukan dan dapat diakses menggunakan operator ampersand (&), gunanya untuk menunjuk ke alamat dalam memori. Langkah menggunakan pointer adalah sebagai berikut:

* Pertama-tama, mendefinisikan variabel pointer,
* kemudian, tetapkan alamat variabel ke pointer
* akhirnya akses nilai pada alamat yang tersedia dalam variabel pointer. caranya dilakukan dengan menggunakan operator unary \* yang mengembalikan nilai variabel yang terletak di alamat yang ditentukan oleh operand-nya.

Contoh penggunaan pointer :

#include <stdio.h>

int main () {

int num = 12; /\* variabel sebenarnya \*/

int \*pt; /\* variabel pointer \*/

pt = # /\* menyimpan alamat dari variabel lain\*/

printf("Address of num variable: %xn", &num );

/\* alamat yang tersimpan di dalam pointer \*/

printf("Address stored in pt variable: %xn", pt);

/\* mengakses nilai dengan pointer \*/

printf("Value of \*pt variable: %dn", \*pt );

return 0;

}

### **Struct**

Di dalam Array memungkinkan untuk menampung beberapa item data dari jenis yang sama. Demikian pula di dalam struct, struct sendiri adalah tipe data yang ditentukan pengguna lain yang tersedia dalam C yang memungkinkan untuk menggabungkan item data dari berbagai jenis. Untuk mendefinisikan struktur, Anda harus menggunakan pernyataan struct. Pernyataan struct mendefinisikan tipe data baru, dengan lebih dari satu anggota. Format pernyataan struct adalah sebagai berikut :

- struct [tag struktur]

{

definisi anggota;

definisi anggota;

...

definisi anggota;

} [satu variabel struktur atau lebih];

Tag struktur bersifat opsional dan setiap definisi anggota adalah definisi variabel normal, contahnya int i; atau float f; atau definisi variabel lain yang valid. Pada akhir definisi struktur, sebelum titik koma akhir, Anda dapat menentukan satu atau lebih variabel struktur tetapi bersifat opsional.

## Operasi File

Dalam pemrograman skala menengah dan besar, input dan output data dalam program sering berjumlah banyak dan perlu ditulis atau disimpan dalam sebuah file. File merupakan sebuah organisasi dari sejumlah record. Pada masing-masing record bisa terdiri dari satu atau beberapa field dan setiap field terdiri dari satu atau beberapa byte. Byte tersebut merupakan susunan dari 8 bit. Penggunaan operasi file dalam sebuah program sangat dibutuhkan dalam pembuatan program yang sesungguhnya. File ini dibutuhkan sebagai tempat penampung data-data selama operasi program. Jika hanya menggunakan variabel biasa, data yang biasanya diolah program hanya akan tersimpan sementara dalam memori dan akan hilang ketika program close. Berbeda dengan memori penyimpanan data berbasis file akan tersimpan walaupun program telah diclose maupun komputer telah di *shutdown*. Operasi file dalam bahasa C merupakan suatu proses input atau output file dan memanipulasi file yang berupa membuka file, melaksanakan proses file, dan menutup file.

* **Membuka File**

Membuka file ini digunakan untuk membuka atau mengaktifkan file menggunakan fungsi fopen() yang dimana file tersebut dapat berupa file teks ataupun file biner. File biner merupakan file yang memiliki pola penyimpanan di dalam disk dalam bentuk biner yaitu seperti bentuk pada memori atau RAM komputer. Sedangkan, file teks merupakan file yang memiliki pola penyimpanan datanya dalam bentuk karakter. Penambahan yang perlu dilakukan untuk menemukan mode teks ataupun biner adalah untuk file teks yaitu “t” dan unutk file biner yaitu “b”. Prototye yang digunakan untuk fungsi fopen() ini yaitu terdapat pada header fungsi stdio.h.

Bentuk umum dari fopen() ini yaitu :

FILE \*fopen(char \*namafile, char \*mode);

Keterangan :

* Nama file berupa nama file dari file yang akan dibuka ataupun diaktifkan.
* Mode berupa jenis operasi file yang akan dilakukan terhadap file.

Adapun jenis-jenis operasi file yaitu :

1. r yaitu menyatakan file hanya dapat dibaca (file sudah harus ada).
2. w yaitu menyatakan file baaru akan dibuat atau diciptakan (file yang sudah ada akan dihapus).
3. a yaitu untuk membuka file yang sudah ada dan akan dilakukan proses penambahan data (apabila file belum ada, maka akan secara otomatis akan dibuat).
4. r+ yaitu untuk membuka file yang sudah ada dan akan dilakukan proses pembacaan dan penulisan.
5. w+ yaitu untuk membuka file dengan tujuan untuk peembacaan dan penulisan (apabila file sudah ada, maka isinya akan dihapus).
6. a+ yaitu untuk membuka file dengan operasi yang akan dilakukan berupa perekaman maupun pembacaan (apabila file sudah ada, maka isinya akan dihapus).

Contoh :

pf = fopen(“filepath”,”w”);

Variabel pf dideklarasikan sebagai berikut :

FILE \*pf;

Keterangan :

* Menciptakan dan mengaktifkan file bernama “filepath”.
* Dengan mode yaitu “w” atau mode penulisan ke file dan menempatkan pointer ke FILE ke variabel pointer pf.
* **Menutup File**

Untuk menutup file, fungsi yang digunakan yaitu fclose(). Prototype yang digunakan untuk fungsi fclose() ini yaitu terdaapat pada header fungsi stdio.h. File perlu ditutup menggunakan fclose() karena keterbatasan jumlah file yang dapat dibuka secara serentak. Pada saat kita membuka file, maka kita mengalokasikan memori dan pada saat itu proses open tersebut akan terus berjalan. Jika kita ingin membuka file yang lainnya dan tidak meng-*close* file sebelumnya, maka akan terjadi crash memori.

Bentuk umum dari fclose() ini yaitu :

int fclose(FILE \*pf);

Terdapat juga bentuk umum lain dari fclose() yaitu :

int fcloseall(void);

* **Melaksanakan Proses**

1. Menulis Karakter

Dalam menulis karakter, bentuk umum yang digunakan yaitu :

puts(int ch, file \*fp);

Keterangan :

* fp merupakan pointer file yang dihasilkan oleh fopen().
* ch merupakan karakter yang akan ditulis.

1. Membaca Karakter

Dalam membaca karakter dari file, bentuk umum yang digunakan yaitu :

getc(FILE \*fp);

Keterangan :

* fp merupakan pointer yang dihasilkan oleh fopen().
* Fungsi feof() digunakan untuk mendeteksi akhir file.
* Pada saat membaca data feof(FILE \*fp).

1. Membaca dan Menulis String

Fungsi yang digunakan untuk membaca dan menulis string yaitu fgets() dan fputs().

Bentuk umum yang digunakan yaitu :

fgets(char \*str, int p, file \*fp);

fputs(char \*str, file \*fp);

1. Membaca dan Menulis Blok Data

Fungsi yang digunakan untuk membaca dan menulis blok data yaitu fread() dan fwrite().

Bentuk umum yang digunakan yaitu :

fread(void \*buffer, int b\_byte, int c, file \*fp);

fwrite(void \*buffer, int b\_byte, int c, file \*fp);

Keterangan :

* buffer merupakan pointer ke sebuah area di memori yang menampung data yang akan dibaca dari file.
* b\_byte merupakan banyak byte yang akan dibaca ataupun ditulis ke file.
* c merupakan banyaknya item dibaca atau ditulis.

1. Membaca dan Menulis File yang Terformat

Jika yang diinginkan data bilangan dapat disimpan ke dalam file dalam keadaan terformat, maka fungsi yang digunakan yaitu :

fprintf(ptr\_file, “string control”, daftar argument);

fscanf(pts\_file, “string control”, daftar argument);

## CRUD

CRUD merupakan singkatan dari *create*, *read*, *update*, dan *delete*. Operasi CRUD merupakan manipulasi data data dasar untuk database. CRUD ini sering kali digunakan pada aplikasi-aplikasi pengolahan data yang kebanyakan mengguanakan fungsi CRUD didalamnya . Fungsi ini digunakan untuk menambahkan data, menghapus data, serta mengupdate data.

* Create (C) yaitu membuat sebuah data baru.

Contoh :

Kita sedang melakukan registrasi disebuah web itu sudah merupakan create (C) dari CRUD karena kita membuat dan menyimpan data registrasi ke database.

* Read (R) yaitu membaca atau menampilkan suatu data.

Contoh :

Membaca atau menampilkan suatu data yang berada yang berada di suatu database, kemudian di tampilkan pada WEB dengan menggunakan bahasa pemrograman.

* Update (U) yaitu suatu proses mengedit sebuah data dari database yang kemudian diedit dengan menggunakan bahsa pemrograman.

Contoh :

Pada saat kita akan atau ingin mengedit profile di suatu aplikasi.

* Delete (D) yaitu fungsi ini hampir sama dengan fungsi update akan tetapi, pada proses ini melakukan proses untuk penghapusan data di suatu database.

Contoh :

Pada sebuah blog terkadang terdapat beberapa komentar, kemudian kita ingin menghapus komentar tersebut, maka proses tersebut sudah termasuk ke dalam proses delete dalam CRUD.

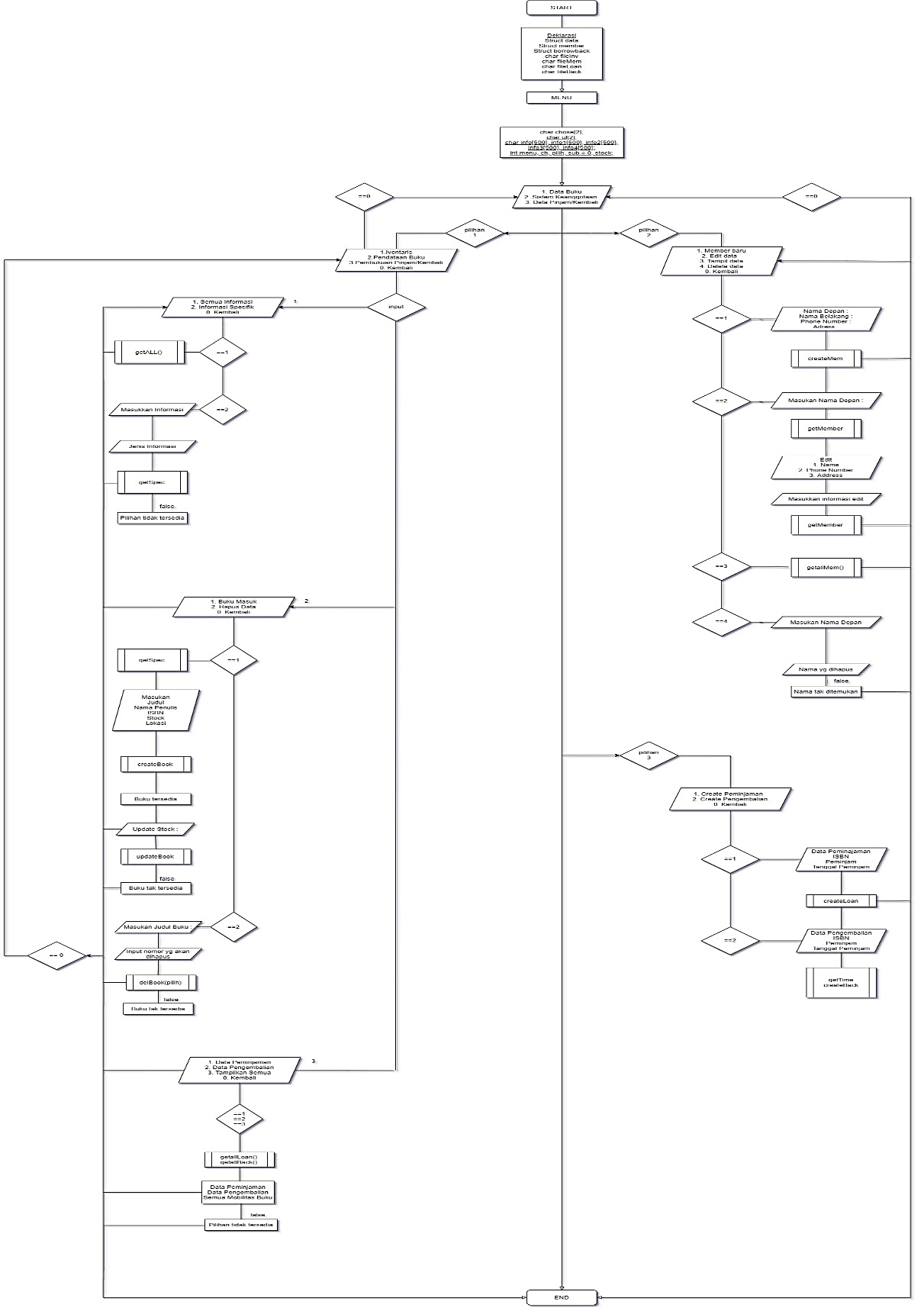
## Parameter

Parameter adalah sarana komunikasi antar fungsi dalam sebuah program. Parameter terdiri dari daftar variable pada header fungsi. Dalam pemograman, diusahakan agar fungsi bersifat independen artinya tidak tergantung pada fungsi lain. Setiap fungsi hanya mengerjakan satu tugas tertentu. Terdapat dua jenis parameter yaitu parameter formal dan parameter aktual. Parameter formal merupakan variabel yang ada pada daftar parameter dalam definisi fungsi. Sedangkan parameter aktual merupakan variabel atau parameter yang digunakan dalam pemanggilan fungsi. Terdapat beberapa cara yang dapat digunakan untuk melewatkan suatu parameter dalam bahasa pemrograman C yaitu pemanggilan secara nilai atau call by value dan pemanggilan secara referensi atau call by reference.

1. Pemanggilan Secara Nilai (Call by Value)

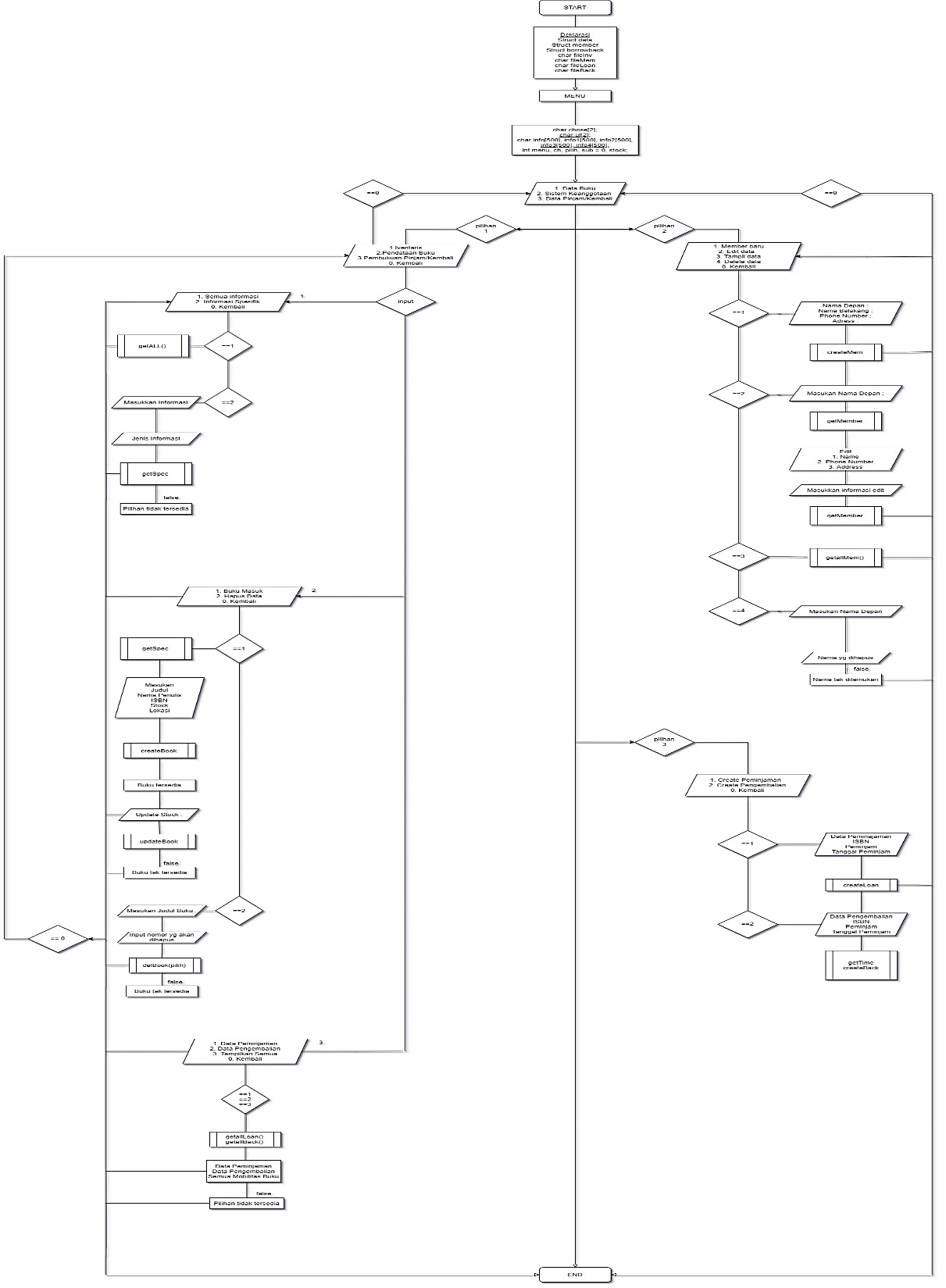
* Pemanggilan secara nilai atau call by value akan menyalin nilai dari parameter aktual ke parameter formal.
* Yang dikirimkan ke fungsi merupakan nilai dari datanya, bukan alamat memori letak dari datanya.
* Fungsi yang menerima kiriman nilai akan menyimpannya di alamat terpisah dari nilai aslinya yang digunakan oleh bagian program yang memanggil fungsi.
* Perubahan nilai di fungsi atau parameter formal tidak akan merubah nilai asli di bagian program yang memanggilnya.
* Pengiriman parameter secara nilai atau call by value merupakan pengiriman searah, yaitu dari bagian program yang memanggil fungsi ke fungsi yang dipanggil.
* Pengiriman suatu nilai dapat dilakukan untuk suatu ungkapan, tidak hanya untuk sebuah variabel, elemen array, ataupun konstanta saja.

1. Pemanggilan Secara Referensi (Call by Reference)

* Pemanggilan secara referensi atau call by reference merupakan upaya yang dapat dilakukan untuk melewatkan alamat dari suatu variabel ke dalam fungsi.
* Yang dikirimkan ke fungsi merupakan alamat letak dari nilai datanya, bukan nilai datanya.
* Fungsi yang menerima kiriman alamat ini, maka menggunakan alamat yang sama untuk mendapatkan nilai datanya.
* Perubahan nilai di fungsi akan merubah nilai asli pada bagian program yang memanggil fungsi.
* Pengiriman parameter secara referensi merupakan pengiriman yang memiliki dua arah, yaitu dari fungsi pemanggil ke fungsi yang dipanggil dan begitu juga sebaliknya.
* ****Pengiriman secara acuan tidak dapat dilakukan untuk suatu ungkapan.

# BAB III

# DESAIN DAN METODE

1. Desain Flowchart
2. Metode

Metode yang digunakan dalam program ini untuk menciptakan sebuah implementasi sistem CRUD (Create, Read, Update, dan Delete) yaitu sebagai berikut :

1. Struktur dasar

Berdasarkan landasan teori, struktur dasar meliput tipe data, konstantan,variabel, deklarasi, operator,komentar, input, dan output. Sesuai namanya yaitu struktur dasar tentu menjadi hal yang paling mendasar dalam membangun sebuah program sehingga pasti digunakan dalam program simulator perpustakaan ini sesuai dengan keggunaanya masing-masing sesuai penjelasan dalam landasan teori.

1. Perulangan

Metode perulangan digunakan dalam program ini untuk berbagai hal seperti menampilan menu berulang-ulang sampai user menginputkan pilihan untuk keluar dari program dengan memanfaatkan metode baik do-while ataupun while, membaca file secara terus menerus hingga sampai pada akhir file atau ¬end of file (EOF) dengan memanfaatkan baik do-while ataupun while, menulis atau meginputkan informasi dalam array ke dalam file secara terus menerus hingga sampai pada akhir index perulangan yang telah dibatasi dengan memanfaatkan metode perulangan for.

1. Penyeleksian kondisi

Metode penyeleksian kondisi digunakan dalam program ini untuk beberapa hal seperti membuat percabangan atas pilihan menu, dan pilihan lainnya yang tersedia dengan memanfaatkan baik switch case ataupun if else, membuat percabangan untuk menangani sejumlah error yang mungkin terjadi akibat kesalahan user atau sebuah kondisi dalam program itu sendiri.

1. Function dan Parameter

Metode function digunakan dalam program ini untuk mempersingkat program dan menghindari baris program yang sama ditulis berulang. Selain itu diguanakn untuk mengembalikan sejumlah nilai sesuai kebutuhan dengan call by value.

1. Array

Metode array digunakan untuk menangani memori penyimpanan dari program ini dalam prosesnya.

1. Abstrak data type (ADT)/Struct

Metode ADT digunakan dalam program ini untuk menciptakan sebuah tipe data baru yang menyimpan kebutuhan yang diinginkan seperti ADT untuk menyimpan informasi buku, informasi anggota, informasi peminjaman, atau informasi pengembalian.

1. Operasi File

Metode operasi file digunakan dalam program ini untuk menyimpan data dalam bentuk file agar dapat diakses kembali dan data tidak hilang begitu saja apabila program tersebut selesai dijalankan.

# BAB IV

# HASIL DAN IMPLEMENTASI

1. Source Code

*//Library Simulator*

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

struct data

{

    char title[500];

    char author[500];

    char isbn[500];

    int stock;

    char loc[500];

} loan[500];

struct member

{

    char nama[500];

    char noTelp[500];

    char Alamat[500];

} mem[500];

struct borrowback

{

    char title[500];

    char isbn[500];

    char name[500];

    char loanTime[500];

    char backTime[500];

} row[500];

char fileInv*[]* = "DB/dataBook.txt";

char fileMem*[]* = "DB/dbMember.txt";

char fileLoan*[]* = "DB/dbLoan.txt";

char fileBack*[]* = "DB/dbBack.txt";

*//Sistem Inventaris*

long getLong(char \*fileName)

{

    FILE \*fp = NULL;

    long off;

    fp = fopen(fileName, "r");

    if (fp == NULL)

    {

        printf("failed to fopen %s\n", fileName);

        exit(EXIT\_FAILURE);

    }

    if (fseek(fp, 0, SEEK\_END) == -1)

    {

        printf("failed to fseek %s\n", fileName);

        exit(EXIT\_FAILURE);

    }

    off = ftell(fp);

    if (off == -1)

    {

        printf("failed to ftell %s\n", fileName);

        exit(EXIT\_FAILURE);

    }

    return off;

    if (fclose(fp) != 0)

    {

        printf("failed to fclose %s\n", fileName);

        exit(EXIT\_FAILURE);

    }

}

int getAll()

{

    int i = 0, loop;

    FILE \*rd;

    rd = fopen(fileInv, "r");

    char ch;

    if (rd == NULL)

    {

        printf("File tidak ditemukan");

        exit(1);

    }

    long size = getLong(fileInv);

    if (size == 0)

    {

        printf("File kosong");

        exit(1);

    }

    do{

        fscanf(rd, " %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c %d %200[^;]%\*c", &loan[i].title, &loan[i].author, &loan[i].isbn, &loan[i].stock, &loan[i].loc);

        printf("[%d]\nTitle: %s\nAuthor: %s\nISBN: %s\nStock: %d\nLocation: %s\n", i + 1, loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, &loan[i].loc);

        ch = fgetc(rd);

        i++;

        loop = i;

    }while (ch != EOF);

    return loop;

    fclose(rd);

}

int getSpec(char \*find, int kind)

{

    int loop = getAll();

    int size = strlen(find);

    int err;

    if (kind == 1)

    {

        system("cls");

        for (int i = 0; i < loop; i++)

        {

            if (strncmp(loan[i].title, find, size) == 0)

            {

                printf("[%d]\nTitle:%s\nAuthor:%s\nISBN:%s\nStock:%d\nLocation:%s\n", i + 1, loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

                err = 0;

            }

        }

        if (err != 0)

        {

            puts("Buku tidak tersedia atau periksa kembali tulisan anda.");

            return 1;

        }

        system("pause");

    }

    else if (kind == 2)

    {

        system("cls");

        for (int i = 0; i < loop; i++)

        {

            if (strncmp(loan[i].author, find, size) == 0)

            {

                printf("[%d]\nTitle:%s\nAuthor:%s\nISBN:%s\nStock:%d\nLocation:%s\n", i + 1, loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

                err = 0;

            }

        }

        if (err != 0)

        {

            puts("Buku tidak tersedia atau periksa kembali tulisan anda.");

            return 1;

        }

        system("pause");

    }

    else if (kind == 3)

    {

        system("cls");

        for (int i = 0; i < loop; i++)

        {

            if (strncmp(loan[i].isbn, find, size) == 0)

            {

                printf("[%d]\nTitle:%s\nAuthor:%s\nISBN:%s\nStock:%d\nLocation:%s\n", i + 1, loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

                err = 0;

            }

        }

        if (err != 0)

        {

            puts("Buku tidak tersedia atau periksa kembali tulisan anda.");

            return 1;

        }

        system("pause");

    }

    else

    {

        puts("Pilihan anda tidak tersedia");

    }

}

int viewBook(int num)

{

    int i = num - 1;

    printf("[%d]\nTitle:%s\nAuthor:%s\nISBN:%s\nStock:%d\nLocation:%s\n", num, loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

}

int getBook(char \*title)

{

    int loop = getAll(), num;

    int err = getSpec(title, 1);

    system("cls");

    if (err == 1)

    {

        printf("Buku tidak tersedia");

        return 0;

    }

    if (err != 1)

    {

        printf("Pilih Buku: ");

        scanf("%d", &num);

        viewBook(num);

        return num;

    }

}

void createBook(char \*title, char \*author, char \*isbn, int stock, char \*loc)

{

    FILE \*cb;

    char ch;

    cb = fopen(fileInv, "a");

    if (cb == NULL)

    {

        printf("File tidak ditemukan");

    }

    fprintf(cb, "\n%s;\n%s;\n%s;\n%d\n%s;", title, author, isbn, stock, loc);

    fclose(cb);

}

void updateBook(int next, int stock)

{

    int loop = getAll();

    system("cls");

    FILE \*ub;

    char ch;

    ub = fopen(fileInv, "w");

    if (ub == NULL)

    {

        printf("File tidak ditemukan");

    }

    for (int i = 0; i < loop; i++)

    {

        if (i == next - 1)

        {

            fprintf(ub, "%s;\n%s;\n%s;\n%d\n%s;", loan[i].title, loan[i].author, loan[i].isbn, stock, loan[i].loc);

        }

        else

        {

            fprintf(ub, "%s;\n%s;\n%s;\n%d\n%s;", loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

        }

    }

    fclose(ub);

}

void delBook(int target)

{

    int loop = getAll(), i;

    system("cls");

    FILE \*db;

    for (i = target-1; i < loop; i++)

    {

        loan[i] = loan[i + 1];

    }

    i -= 1;

    db = fopen(fileInv, "w");

    for (int j = 0; j < i; j++)

    {

        if (j > 0)

        {

            fprintf(db, "\n");

        }

        fprintf(db, "%s;\n%s;\n%s;\n%d\n%s;", loan[j].title, loan[j].author, loan[j].isbn, loan[j].stock, loan[j].loc);

    }

    fclose(db);

}

*//Sistem Keanggotaan*

void createMem(char \*name, char \*phone, char \*address)

{

    FILE \*cm;

    cm = fopen(fileMem, "a");

    int buff = getLong(fileMem);

    if (buff > 0)

    {

        fprintf(cm, "\n");

    }

    fprintf(cm, "%s;\n%s;\n%s;", name, phone, address);

    fclose(cm);

}

int getallMem()

{

    FILE \*gm;

    char ch;

    int size, loop, i = 0;

    gm = fopen(fileMem, "r");

    if (gm == NULL)

    {

        printf("File tidak ditemukan");

        exit(1);

    }

    do{

        fscanf(gm, " %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c", mem[i].nama, mem[i].noTelp, mem[i].Alamat);

        printf("[%d]\nName: %s\nPhone Number: %s\nAddress: %s\n", i + 1, mem[i].nama, mem[i].noTelp, mem[i].Alamat);

        ch = fgetc(gm);

        i++;

        loop = i;

    }while (ch != EOF);

    return loop;

    fclose(gm);

}

void viewMem(int num)

{

    int i = num - 1;

    printf("[%d]\nName: %s\nPhone Number: %s\nAddress: %s\n", num, mem[i].nama, mem[i].noTelp, mem[i].Alamat);

}

int getspecMem(char \*info, int kind)

{

    int loop = getallMem(), err,find;

    system("cls");

    int size = strlen(info);

    if (kind == 1)

    {

        for (int i = 0; i < loop; i++)

        {

            system("cls");

            if (strncmp(mem[i].nama, info, size) == 0)

            {

                printf("[%d]\nName: %s\nPhone Number: %s\nAddress: %s\n", i + 1, mem[i].nama, mem[i].noTelp, mem[i].Alamat);

                err = 0;

                find = 0;

            }

        }

        if (err != 0)

        {

            printf("Anggota tidak ditemukan!");

            find = 1;

        }

    }

    else

    {

        printf("Pilihan tidak tersedia");

        find = -1;

    }

    return find;

}

int getMember(char \*name)

{

    int loop = getallMem(), num;

    system("cls");

    int err = getspecMem(name, 1);

    if (err == 1)

    {

        printf("Nama tidak tergabung dalam anggota");

        return 0;

    }

    if (err != 1)

    {

        printf("\nPilih Member; ");

        scanf("%d", &num);

        system("pause");

        system("cls");

        viewMem(num);

        return num;

    }

}

void updateMem(int next, char \*info, int kind)

{

    int loop = getallMem();

    system("cls");

    FILE \*um;

    um = fopen(fileMem, "w");

    if (um == NULL)

    {

        printf("File tidak ditemukan");

    }

    if (kind == 1)

    {

        for (int i = 0; i < loop; i++)

        {

            if (i == next - 1)

            {

                if (i > 0)

                {

                    fprintf(um, "\n");

                }

                fprintf(um, "%s;\n%s;\n%s;", info, mem[i].noTelp, mem[i].Alamat);

            }

            else

            {

                fprintf(um, "%s;\n%s;\n%s;", mem[i].nama, mem[i].noTelp, mem[i].Alamat);

            }

        }

    }

    else if (kind == 2)

    {

        for (int i = 0; i < loop; i++)

        {

            if (i == next - 1)

            {

                if (i > 0)

                {

                    fprintf(um, "\n");

                }

                fprintf(um, "%s;\n%s;\n%s;", mem[i].nama, info, mem[i].Alamat);

            }

            else if (i != next - 1)

            {

                fprintf(um, "%s;\n%s;\n%s;", mem[i].nama, mem[i].noTelp, mem[i].Alamat);

            }

        }

    }

    else if (kind == 3)

    {

        for (int i = 0; i < loop; i++)

        {

            if (i == next - 1)

            {

                if (i > 0)

                {

                    fprintf(um, "\n");

                }

                fprintf(um, "%s;\n%s;\n%s;", mem[i].nama, mem[i].noTelp, info);

            }

            if (i != next - 1)

            {

                fprintf(um, "%s;\n%s;\n%s;", mem[i].nama, mem[i].noTelp, mem[i].Alamat);

            }

        }

    }

    else

    {

        printf("Pilihan tidak tersedia");

    }

    fclose(um);

}

void delMem(int target)

{

    int loop = getallMem(), i = 0;

    system("cls");

    FILE \*dm;

    for (i = target - 1; i < loop; i++)

    {

        mem[i] = mem[i + 1];

    }

    i -= 1;

    dm = fopen(fileMem, "w");

    for (int j = 0; j < i; j++)

    {

        if (j > 0){

            fprintf(dm, "\n");

        }

        fprintf(dm, "%s;\n%s;\n%s;", mem[j].nama, mem[j].noTelp, mem[j].Alamat);

    }

    fclose(dm);

}

*//Sistem Peminjaman/Pengembalian*

char \*getTime()

{

    struct tm \*info;

    time\_t rawtime;

    time(&rawtime);

    info = localtime(&rawtime);

    char \*time = asctime(info);

    int len = strlen(time);

    if (time[len - 1] == '\n')

    {

        time[len - 1] = 0;

    }

    return time;

}

int getallLoan()

{

    int i = 0, loop;

    FILE \*gal;

    gal = fopen(fileLoan, "r");

    char ch;

    if (gal == NULL)

    {

        printf("File tidak ditemukan!");

    }

    long size = getLong(fileLoan);

    if (size == 0)

    {

        printf("File Kosong");

        exit(1);

    }

    while (ch != EOF)

    {

        fscanf(gal, "%200[^;]%\*c %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c", row[i].title, row[i].isbn, row[i].name, row[i].loanTime);

        printf("[%d]\nTitle: %s\nISBN: %s\nLoan by %s\nLoan time %s\n", i + 1, row[i].title, row[i].isbn, row[i].name, row[i].loanTime);

        ch = fgetc(gal);

        i++;

        loop = i;

    }

    return loop;

    fclose(gal);

}

void delLoan(char \*isbn, char \*namaDepan)

{

    int loop = getallLoan(), i, target;

    system("cls");

    FILE \*dl;

    int size = strlen(namaDepan);

    for (i = 0; i < loop; i++)

    {

        if (strcmp(row[i].isbn, isbn) == 0 && strncmp(row[i].name, namaDepan, size) == 0)

        {

            target = i;

        }

    }

    for (i = target; i < loop; i++)

    {

        row[i] = row[i + 1];

    }

    i -= 1;

    dl = fopen(fileLoan, "w");

    for (int j = 0; j < i; j++)

    {

        if (j > 0)

        {

            fprintf(dl, "\n");

        }

        fprintf(dl, "%s;\n%s;\n%s;\n%s;", row[j].title, row[j].isbn, row[j].name, row[j].loanTime);

    }

    fclose(dl);

}

char \*getspecLoan(char \*isbn)

{

    int loop = getAll();

    system("cls");

    for (int i = 0; i < loop; i++)

    {

        if (strcmp(loan[i].isbn, isbn) == 0)

        {

            return loan[i].title;

        }

    }

}

char \*getDate(char \*isbn, char \*namaDepan)

{

    int loop = getallLoan(), i = 0;

    system("cls");

    int size = strlen(namaDepan);

    for (int i = 0; i < loop; i++)

    {

        if (strcmp(row[i].isbn, isbn) == 0 && strncmp(row[i].name, namaDepan, size) == 0)

        {

            return row[i].loanTime;

        }

    }

}

void minusStock(char \*isbn)

{

    int loop = getAll();

    system("cls");

    FILE \*ms;

    ms = fopen(fileInv, "w");

    char ch;

    if (ms == NULL)

    {

        printf("File tidak ditemukan");

        exit(1);

    }

    for (int i = 0; i < loop; i++)

    {

        if (strcmp(isbn, loan[i].isbn) == 0)

        {

            loan[i].stock -= 1;

        }

    }

    for (int i = 0; i < loop; i++)

    {

        if (i > 0)

        {

            fprintf(ms, "\n");

        }

        fprintf(ms, "%s;\n%s;\n%s;\n%d\n%s;", loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

    }

    fclose(ms);

}

int getallBack()

{

    int i = 0, j = 0, loop;

    FILE \*gab;

    gab = fopen(fileBack, "r");

    char ch;

    if (gab == NULL)

    {

        printf("File tidak ditemukan!");

    }

    long size = getLong(fileBack);

    if (size == 0)

    {

        printf("File Kosong");

        exit(1);

    }

    do

    {

        fscanf(gab, " %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c %200[^;]%\*c", row[i].title, row[i].isbn, row[i].name, row[i].loanTime, row[i].backTime);

        printf("[%d]\nTitle: %s\nISBN: %s\nLoan by %s\nLoan time at %s\nBack time at %s\n", i + 1, row[i].title, row[i].isbn, row[i].name, row[i].loanTime, row[i].backTime);

        ch = fgetc(gab);

        i++;

        loop = i;

    } while (ch != EOF);

    return loop;

    fclose(gab);

}

void plusStock(char \*isbn)

{

    int loop = getAll();

    system("cls");

    FILE \*ms;

    ms = fopen(fileInv, "w");

    char ch;

    if (ms == NULL)

    {

        printf("File tidak ditemukan");

        exit(1);

    }

    for (int i = 0; i < loop; i++)

    {

        if (strcmp(isbn, loan[i].isbn) == 0)

        {

            loan[i].stock += 1;

        }

    }

    for (int i = 0; i < loop; i++)

    {

        if (i > 0)

        {

            fprintf(ms, "\n");

        }

        fprintf(ms, "%s;\n%s;\n%s;\n%d\n%s;", loan[i].title, loan[i].author, loan[i].isbn, loan[i].stock, loan[i].loc);

    }

    fclose(ms);

}

int checkStock(char \*isbn)

{

    int loop = getAll(), i = 0;

    system("cls");

    for (int i = 0; i < loop; i++)

    {

        if (strcmp(loan[i].isbn, isbn) == 0)

        {

            return loan[i].stock;

        }

    }

}

void createLoan(char \*isbn, char \*name, char \*loanTime)

{

    FILE \*cl;

    cl = fopen(fileLoan, "a");

    int size = getLong(fileLoan);

    char \*title = strdup(getspecLoan(isbn));

    int stock = checkStock(isbn);

    if (stock == 0)

    {

        printf("Stock Kosong");

        exit(1);

    }

    if (size > 0)

    {

        fprintf(cl, "\n");

    }

    fprintf(cl, "%s;\n%s;\n%s;\n%s;", title, isbn, name, loanTime);

    minusStock(isbn);

    fclose(cl);

}

void createBack(char \*isbn, char \*name, char \*backTime)

{

    FILE \*cb;

    cb = fopen(fileBack, "a");

    int size = getLong(fileBack);

    char \*title = strdup(getspecLoan(isbn));

    char \*loanTime = strdup(getDate(isbn, name));

    int len = strlen(loanTime);

    if (loanTime[len - 1] == '\n')

    {

        loanTime[len - 1] = 0;

    }

    if (size > 0)

    {

        fprintf(cb, "\n");

    }

    fprintf(cb, "%s;\n%s;\n%s;\n%s;\n%s;", title, isbn, name, loanTime, backTime);

    plusStock(isbn);

    delLoan(isbn, name);

    fclose(cb);

}

int main()

{

    char chose[2];

    char ul[2];

    char info[500], info1[500], info2[500], info3[500], info4[500];

    int menu, ch, pilih, sub = 0, stock;

    menu :

    system("cls");

    printf("Menu:\n");

    puts("1. Data Buku");

    puts("2. Sistem Keanggotaan");

    puts("3. Data Pinjam/Kembali");

    puts("0. Exit");

    printf("Pilih menu: ");

    scanf("%d", &menu);

    switch (menu)

    {

    case 1:

        do

        {

            dataBuku :

            system("cls");

            puts("Data Buku");

            puts("1. Iventaris");

            puts("2. Pendataan Buku");

            puts("3. Pembukuan Peminjaman/Pengembalian");

            puts("0. Kembali");

            printf("PILIH: ");

            fflush(stdin);

            gets(chose);

            if (strcmp("1", chose) == 0)

            {

                inventaris:

                system("cls");

                puts("Inventaris Buku");

                puts("1. Semua informasi");

                puts("2. Informasi Spesifik");

                puts("0. Kembali");

                printf("PILIH: ");

                scanf("%d", &sub);

                if (sub == 1)

                {

                    system("cls");

                    getAll();

                }

                else if (sub == 2)

                {

                    system("cls");

                    printf("Masukan Informasi:");

                    fflush(stdin);

                    gets(info);

                    printf("Jenis Informasi\n1.Judul\n2.Author\n3.ISBN\n");

                    scanf("%d", &pilih);

                    getSpec(info, pilih);

                }

                else if (sub == 0){

                    goto dataBuku;

                }

                else

                {

                    system("cls");

                    puts("Pilihan tidak tersedia");

                    system("pause");

                    goto inventaris;

                }

            }

            else if (strcmp("2", chose) == 0)

            {

                pendataan :

                system("cls");

                puts("Pendataan Buku");

                printf("1. Buku Masuk\n2. Hapus Data\n0. Kembali\nPILIH: ");

                scanf("%d", &sub);

                if (sub == 1)

                {

                    system("cls");

                    printf("Masukan Judul Buku: ");

                    fflush(stdin);

                    gets(info);

                    int next = getBook(info);

                    if (next == 0)

                    {

                        system("cls");

                        printf("Buku tidak tersedia\n");

                        printf("Masukan Judul: ");

                        fflush(stdin);

                        gets(info1);

                        printf("Masukan Nama Penulis: ");

                        fflush(stdin);

                        gets(info2);

                        printf("Masukan ISBN: ");

                        fflush(stdin);

                        gets(info3);

                        printf("Masukan Stock: ");

                        scanf("%d", &stock);

                        printf("Masukan Lokasi: ");

                        fflush(stdin);

                        gets(info4);

                        createBook(info1, info2, info3, stock, info4);

*//buat baru*

                    }

                    if (next != 0)

                    {

                        printf("Buku tersedia");

                        printf("Update Stock: ");

                        scanf("%d", &stock);

                        updateBook(next, stock);

*//update*

                    }

                }

                else if (sub == 2)

                {

                    system("cls");

                    printf("Masukan Judul Buku: ");

                    fflush(stdin);

                    gets(info);

                    int next = getBook(info);

                    if (next == 0)

                    {

                        puts("Buku tidak tersedia");

                    }

                    if (next != 0)

                    {

                        printf("Yakin ingin menghapus data ini?\nInput kembali nomor buku yang ingin kamu hapus!\n");

                        printf("Input");

                        scanf("%d", &pilih);

                        delBook(pilih);

                    }

                }

                else if (sub == 0){

                    goto dataBuku;

                }

                else

                {

                    system("cls");

                    printf("Pilihan tidak tersedia\n");

                    system("pause");

                    goto pendataan;

                }

            }

            else if (strcmp("3", chose) == 0)

            {

                system("cls");

                pinjamKembali :

                puts("Data Pinjam/Pengembalian");

                printf("1. Data Peminjaman\n2. Data Pengembalian\n3. Tampilkan Semua\n0. Kembali\n");

                printf("Anda ingin mengakses yang mana?(input nomor akses)\nPilih: ");

                scanf("%d", &pilih);

                if (pilih == 1)

                {

                    system("cls");

                    printf("Data Peminjaman\n");

                    getallLoan();

                }

                else if (pilih == 2)

                {

                    system("cls");

                    printf("Data Pengembalian\n");

                    getallBack();

                }

                else if (pilih == 3)

                {

                    system("cls");

                    printf("Semua Mobilitas Buku");

                    getallLoan();

                    getallBack();

                }

                else if (pilih == 0){

                    goto dataBuku;

                }

                else

                {

                    system("cls");

                    puts("Pilihan tidak tersedia!");

                    system("pause");

                    goto pinjamKembali;

                }

            }

            else if (strcmp("0", chose) == 0){

                goto menu;

            }

            else

            {

                system("cls");

                printf("Pilihan tidak tersedia..");

            }

            printf("\nUlang? Ketik 1: ");

            fflush(stdin);

            gets(ul);

        } while (strcmp("1", ul) == 0);

        break;

    case 2:

        do

        {

            memberSystem :

            system("cls");

            puts("Sistem Member");

            puts("1. Member baru");

            puts("2. Edit data");

            puts("3. Tampil data");

            puts("4. Delete data");

            puts("0. Kembali");

            printf("PILIH: ");

            scanf("%d", &ch);

            if (ch == 1)

            {

                system("cls");

                printf("Name Depan: ");

                fflush(stdin);

                gets(info);

                printf("Nama Belakang: ");

                fflush(stdin);

                gets(info1);

                strcat(info, info1);

                printf("Phone Number: ");

                fflush(stdin);

                gets(info2);

                printf("Address: ");

                fflush(stdin);

                gets(info3);

                createMem(info, info2, info3);

                system("pause");

            }

            else if (ch == 2)

            {

                system("cls");

                printf("Masukan Nama Depan: ");

                fflush(stdin);

                gets(info);

                int next = getMember(info);

                if (next == 0)

                {

                    puts("Anggota tidak ditemukan");

                }

                if (next != 0)

                {

                    printf("Apa yang ingin kamu edit?\n1.Name\n2.Phone Number\n3.Address\n");

                    printf("Pilih:");

                    scanf("%d", &pilih);

                    printf("\nMasukan Informasi (yang ingin diedit)");

                    fflush(stdin);

                    gets(info);

                    updateMem(next, info, pilih);

                }

                system("pause");

            }

            else if (ch == 3)

            {

                getallMem();

                system("pause");

            }

            else if (ch == 4)

            {

                system("cls");

                printf("Masukan Nama Depan: ");

                fflush(stdin);

                gets(info);

                int next = getMember(info);

                if (next == 0)

                {

                    puts("Anggota tidak ditemukan");

                }

                if (next != 0)

                {

                    printf("\nYakin ingin mengahapus?\nInput nomor anggota yang ingin kamu hapus!\n");

                    printf("Input: ");

                    scanf("%d", &pilih);

                    delMem(pilih);

                }

                system("pause");

            }

            else if (ch == 0){

                goto menu;

            }

            else

            {

                printf("\nPilihan tidak tersedia");

                goto memberSystem;

            }

        } while (ch != 0);

    case 3:

        do

        {

            system("cls");

            printf("Data Pinjam/Kembali\n");

            printf("1. Create Peminjaman\n2. Create Pengembalian\n0. Kembali\n");

            printf("PILIH: ");

            scanf("%d", &pilih);

            if (pilih == 1)

            {

                printf("Data Peminjaman\nMasukan Data\n");

                printf("ISBN: ");

                fflush(stdin);

                gets(info);

                printf("Peminjam: ");

                fflush(stdin);

                gets(info1);

                printf("Tanggal Peminjaman");

                strcpy(info2, getTime());

                createLoan(info, info1, info2);

            }

            else if (pilih == 2)

            {

                printf("Data Pengembalian\nMasukan Data\n");

                printf("ISBN: ");

                fflush(stdin);

                gets(info);

                printf("Nama Depan: ");

                fflush(stdin);

                gets(info1);

                printf("Tanggal Pengembalian");

                strcpy(info2, getTime());

                createBack(info, info1, info2);

            }

            else if (pilih == 0){

                goto menu;

            }

            else

            {

                system("cls");

                puts("Pilihan tidak tersedia");

                system("pause");

            }

        } while (pilih != 0);

        break;

    case 0:

        system("cls");

        exit(0);

    default:

        printf("Menu tidak tersedia tolong lah ya...FUCEK");

    }

    return 0;

}

Program diatas bernama app.c, program ini mengimplementasikan sistem CRUD (create,read,update,delete). Adapun fungsi main sebagai fungsi utama, fungsi ini didukung oleh 26 subprogram atau subfungsi agar berjalan sesuai dengan fungsinya. Berikut penjelasan ide dan kegunaan dari setiap subprogram yang dibagi 3 sesuai kebutuhan yaitu sistem inventaris, sistem keanggotaan,dan sistem peminjaman/pengembalian:

1. **Sistem Inventaris**
   1. getLong(char\*fileName)

Fungsi getLong berguna untuk membaca size dari sebuah file. Fungsi ini penting untuk menangani *error* yang terjadi ketika ternyata file ditemukan tetapi dalam keadaan kosong.Selain itu, fungsi ini membantu fungsi lainnya menangani kesalahan *enter* saat penulisan isi file.

* 1. getAll()

Fungsi getAll berguna untuk membaca isi dari fileInv dan menyalin ke dalam *array* serta me-*return* banyak array yang terisi. Banyak array diketahui melali seberapa banyak fungsi ini mekalukan perulangannya.

* 1. getSpec(char\*find,int kind)

Fungsi getSpec berguna untuk membaca sekaligus mencari informasi spesifik di dalam fileInv. Ketika mencari sebuah informasi, user akan diminta untuk memasukan informasinya kemudia memilih jenis informasi apa ini, kedua informasi itu akan digunakan sebagai parameter dari fungsi ini.

* 1. viewBook (int num)

Fungsi viewBook berguna untuk melihat buku dengan informasi spesifik dan memanfaatkan index array sebagai parameternya.

* 1. getBook(char\*title)

Fungsi getBook berguna untuk mendapatkan informasi apakah buku tersedia atapun tidak tersedia dengan memasukan judul buku kedalam parameternya.

* 1. createBook(char\*title,char\*author,char\*isbn,int stock,char\*loc)

Fungsi ini berguna untuk membuat sebuah informasi buku baru dan melanjutkan tugas fungsi getBook, ketika buku yang dicari tidak tersedia sehingga user perlu memasukan informasi buku tersebut ke dalam fileInv.

* 1. updateBook (int next,int stock)

Fungsi ini melajutkan tugas fungsi getBook, ketika buku yang dicari tersedia dan user perlu mengupdate stock buku tersebut ke dalam fileInv.

* 1. delBook (int target)

Fungsi ini berguna untuk menghapus seluruh informasi sebuah buku dalam fileInv.

1. **Sistem Keanggotaan**
   1. createMem (char\*name,char\*phone,char\*addres)

Fungsi createMem berguna untuk menambahkan informasi anggota perpustakaan baru kemudian menyimpannya dalma fileMem.

* 1. getallMem ()

Fungsi ini berguna untuk membaca dan menyimpan informasi di dalam fileMem ke array kemudian menampilkan seluruh informasinya kepada user. Selain itu mengembalikan sebuah nilai sebanyak array yang digunakan.

* 1. viewMem (int num)

Fungsi viewMem tidak jauh berbeda dari fungsi viewBook hanya saja array of struct yang digunakan berbeda, tetapi hal yang sama yaitu berguna untuk melihat buku dengan informasi spesifik dan memanfaatkan index array sebagai parameternya.

* 1. getspecMem (char\*info, int kind)

Fungsi getspecMem berguna untuk membaca sekaligus mencari informasi spesifik di dalam fileMem. Ketika mencari sebuah informasi, user akan diminta untuk memasukan informasinya kemudia memilih jenis informasi apa ini, kedua informasi itu akan digunakan sebagai parameter dari fungsi ini.

* 1. getMember (char\*name)

Fungsi getMember berguna untuk mendapatkan informasi apakah sesorang telah tergabung atau belum dalam sistem keanggotaan, jika tergabung fungsi ini dapat dilanjutkan ke fungsi createMem atau updateMem. Selain itu fungsi ini juga mengembalikan sebuah nilai yang mengindikasi letak index arrat dari informasi yang telah diperoleh atau dipilih.

* 1. updateMem (int next, char \*info, int kind)

Fungsi ini updateMem berguna untuk memperbarui informasi dari anggota sesuai indexnya.

* 1. delMem (int target)

Fungsi delMem berguna untuk menghapus secara permanen seluruh informasi dari seorang anggota.

1. **Sistem Peminjaman Atau Pengembalian**
   1. getTime()

Fungsi getTime berguna untuk mengambil keteranagn waktu dari komputer dengan memanfaatkan *library* bahasa C yaitu time.h, kemudian mengembalikannya agar dapat digunakan di dalam fungsi lainnya.

* 1. getallLoan()

Fungsi getallLoan berguna untuk membaca informasi di dalam fileLoan kemudian menyimpannya ke dalam array untuk selanjutnya ditampilkan.

* 1. delLoan(char\*isbn,char\*namaDepan)

Fungsi delLoan berguna untuk menghapuse secara permanen data peminjamkan agar nanti bisa dialokasikan ke dalam fileBack yaitu file penyimpan data pengembalian.

* 1. getspecLoan(char\*isbn)

Fungsi ini berguna membaca sekaligus mencari informasi spesifik di dalam fileLoan yang kemudian akan ditampilkan.

* 1. getDate(char\*isbn,char\*namaDepan)

Fungsi ini berguna untuk menyalin tanggal dan waktu peminjaman di dalam fileLoan.

* 1. getallBack()

Fungsi ini berguna untuk membaca fileBack kemuida menyalin informasi ke dalam array untuk ditampilkan. Selain itu mengembalikan sebuah nilai sebanyak array yang digunakan.

* 1. plusStock(char\*isbn)

Fungsi ini berguna untuk menambahakn 1 stock kembali ke dalam fileInv setalah sebuah buku dikembalikan.

* 1. minusStock(char\*isbn)

Fungsi ini kembalikan dari fungsi plusStock yaitu mengurangi 1 stock di fileInv setalah sebuah buku dipinjam.

* 1. checkStock(char\*isbn)

Fungsi ini berguna untuk menangani *error* ketika stock sebuah judul buku sudah habis atau kosong.

* 1. createLoan(char\*isbn, char\*name,char\*loanTime)

Fungsi ini berguna untuk mencatat informasi peminjaman kemudian menyimpannya ke dalam fileLoan.

* 1. createBack(char\*isbn, char\*name,char\*backTime)

Fungsi ini dapat digunakan untuk mencatan informasi pengembalian kemudian menyimpannya ke dalam file Back.

1. Output Program

Sebuah gambar berisi teks

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

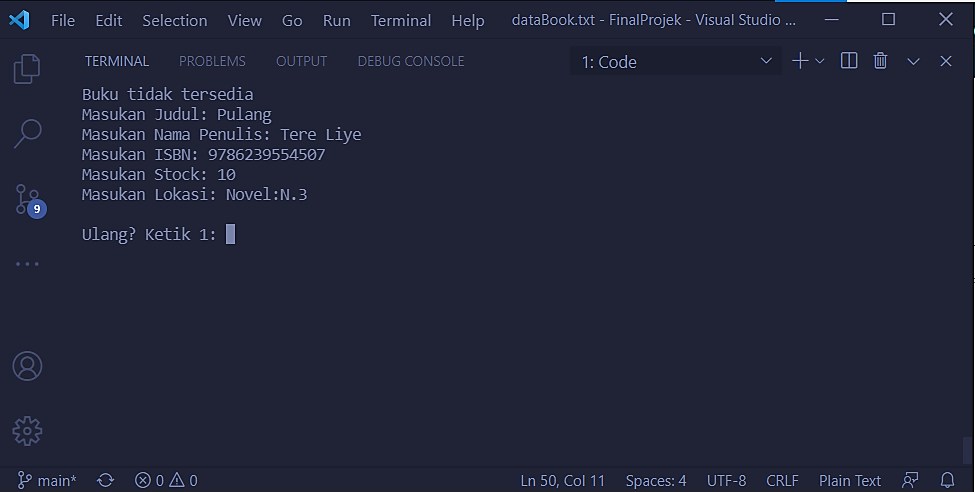
Deskripsi dibuat secara otomatis Sebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks, cuplikan layar, monitor, layar

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, monitor, layar

Deskripsi dibuat secara otomatis

Sebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatis Sebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, monitor, cuplikan layar, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, layar, elektronik

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, monitor, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, layar

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatisSebuah gambar berisi teks, cuplikan layar, elektronik, komputer

Deskripsi dibuat secara otomatisSebuah gambar berisi teks

Deskripsi dibuat secara otomatis

# BAB V

# PENUTUP

1. Kesimpulan

Berdasarkan hasil laporan final project di atas, maka dapat disimpulkan bahwa :

1. Dengan mempelajari dan memahami materi struktur dasar, perulangan dan penyeleksian kondisi, array, struct, pointer, operasi file, CRUD, dan parameter dapat membuat program *Library Simulator* ini.
2. Program *Library Simulator* dapat membantu para pustakawan untuk menyelesaikan tugasnya menjadi lebih mudah. Selain itu, adanya program ini dapat menyimpan data lebih aman daripada menulis data dibuku tulis.
3. Saran

Demikianlah laporan praktikum ini yang dapat kami paparkan, besar harapan kami laporan ini dapat bermanfaat bagi pembaca. Karena keterbatasan pengetahuan dan referensi, penulis menyadari makalah ini masih jauh dari sempurna, oleh karena itu saran dan kritik yang membangun diharapkan agar laporan ini dapat disusun menjadi lebih baik lagi dimasa yang akan datang.

# DAFTAR PUSTAKA

Kawengian, K. 2017. Peranan Komunikasi Pemerintah. *Acta Diurna.* Vol (8).

Aji, B. A. S., Kom, S., Kom, M., & Rahmanti, F. Z. 2021. *Dasar Pemrograman Dalam Bahasa C*. Deepublish.

A. M. Bachtiar. 2018. *Pemrograman C dan C++*. Bandung : INFORMATIKA

Jeklin Harefa, S. 2017. *Perulangan Dalam C*. Diambil kembali dari BINUS UNIVERSITY School of Computer Science. Tersedia pada link: <https://socs.binus.ac.id/2017/03/20/perulangan-dalam-bahasa-c/#:~:text=Perulangan%20atau%20yang%20sering%20disebut,dalam%20batas%20yang%20telah%20ditentukan>. Diakses pada tanggal 23 Mei 2021.

Suyanto. 2009. *Perintah Input – Output*. Diambil Kembali dari Universitas Bina Darma. Tersedia pada link: <http://eprints.binadarma.ac.id/629/1/PRAK.%20ALGORITMA%20%26%20PEMROGRAMAN%20materi%2014.pdf>. Diakses pada tanggal 23 Mei 2021.

Farkhan. 2020. Array, Pointer, dan Struct Dalam Bahasa C. Tersedia pada link: <https://mochammadfarkhan.com/2020/02/array-pointer-dan-struct-dalam-bahasa-c/>. Diakses pada tanggal 23 Mei 2021.

Budiaman, I. 2017. Pertemuan 11: Operasi File Dalam C. Tersedia pada link: <https://docplayer.info/41014643-Pertemuan-11-operasi-file-dalam-c.html>. Diakses pada tanggal 9 Mei 2021.

Gunardi, S. 2017. Operasi File Dasar Pemrograman. Tersedia pada link: <https://docplayer.info/39568882-Operasi-file-dasar-pemrograman.html>. Diakses pada tanggal 9 Mei 2021.

Unknown. 2017. Pengenalan CRUD (Create, Read, Update, Delete). Tersedia pada link: <http://muhfajarshodiq.blogspot.com/2017/01/pengenalan-crud-create-read-update.html?m=1>. Diakses pada tanggal 9 Mei 2021.

Wirdasari, D. 2010. Mengenal Subroutine Pada Pemrograman “C”. Jurnal SAINTIKOM. Vol (9):7-9.