

TUGAS PRAKTIKUM ALGORITMA DAN PEMROGRAMAN

FINAL PROJECT

PROGRAM SIMULASI RUMAH SAKIT SPESIALIS



Disusun Oleh:

Kelompok IV

KELAS B

Ni Kadek Evi Dianasari	2008561021
I Gusti Ngurah Febri Ananda Krisna	2008561025
I Made Dirga Adi Guna	2008561036
Gede Gery Sastrawan	2008561039

Dosen Pengampu :

Dr. Ngurah Agus Sanjaya ER, S.Kom., M.Kom.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

2021

KATA PENGANTAR

Puja-puji syukur saya panjatkan kehadapan Ida Sang Hyang Widhi Wasa/ Tuhan Yang Maha Esa karena atas segala rahmat-Nya sehingga kami dapat menyelesaikan laporan Final Project Praktikum Algoritma Pemrograman ini dengan baik. Tidak lupa kami juga mengucapkan banyak terimakasih kepada orang tua kami, dosen pembimbing, asisten dosen pembimbing, dan teman-teman kami yang telah mendukung kami untuk menyelesaikan laporan Final Project ini dengan baik.

Harapan kami, dengan adanya laporan ini mampu berguna bagi para pembaca sebagaimana yang telah tertulis di dalamnya. Kami mohon kritik dan saran, agar laporan ini menjadi lebih baik lagi.

Kami mohon maaf apabila masih terdapat kekurangan di dalam laporan yang saya tulis. Karena keterbatasan pengetahuan maupun pengalaman kami. Kami yakin masih banyak kekurangan dalam laporan ini, oleh karena itu kami sangat mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini. Atas perhatiannya, kami ucapkan terima kasih.

Denpasar, Mei 2021

Penyusun

DAFTAR ISI

KATA PENGANTAR.....	ii
DAFTAR ISI	iii
BAB I LATAR BELAKANG	1
1.1 Latar Belakang.....	1
BAB II LANDASAN TEORI.....	2
2.1 Tipe Data Primitive Bahasa C	2
2.1.1 Integer	2
2.1.2 Float	2
2.1.3 Char.....	2
2.2 Operator Aritmatika, Pembanding, dan Logika.....	3
2.3 Percabangan dalam Bahasa C.....	3
2.3.1 Percabangan If	3
2.3.2 Percabangan Switch	5
2.4 Perulangan dalam Bahasa C	5
2.4.1 For	6
2.4.2 Nested Loop (Perulangan Bersarang).....	6
2.4.3 While.....	7
2.4.4 Do-While	7
2.5 Array	7
2.6 Pointer.....	8
2.7 Pelengkap dari Fungsi Operasi String dan Karakter.....	9
2.7.1 Pelengkap dari Fungsi Operasi String dan contohnya.....	9
2.7.2 Pelengkap dari Fungsi Operasi karakter dan contohnya.....	12
2.8 Call by Value and Call by Reference.....	13
2.8.1 Cara Melewatkan Parameter.....	13
2.8.2 Cara Pemanggilan Secara Referensi	14
2.9 Operasi File dalam Bahasa Pemrograman.....	15
2.9.1 Membuka file.....	15
2.9.2 Menutup file	17

2.9.3	Meletakkan Data ke Penyangga	17
2.9.4	Manipulasi File	18
2.10	Fungsi-Fungsi dari File Operasi.	18
2.10.1	fputc()	19
2.10.2	fgetc()	19
2.10.3	putw()	19
2.10.4	getw()	20
2.10.5	fputs()	20
2.10.6	fgets()	20
2.10.7	fprintf()	20
2.10.8	fscanf()	21
2.10.9	fwrite().....	21
2.10.10	fread().....	21
2.10.11	fseek() dan feof().....	22
BAB III DESAIN DAN METODE.....		23
3.1	Flowchart Program	23
3.2	Metode Program	23
BAB III HASIL DAN IMPLEMENTASI.....		25
3.1	Screenshot Kode Program.....	25
3.1.1	File main.c	25
3.1.2	File cekCovid.c	26
3.1.3	cekCovid.h.....	28
3.1.4	cekPasien.c	28
3.1.5	cekPasien.h	29
3.1.6	libraryDatabasePasien.c.....	29
3.1.7	libraryDatabasePasien.h	29
3.1.8	Login.c.....	30
3.1.9	Login.h	33
3.1.10	readPasien.c	33
3.1.11	readPasien.h.....	36
3.2	Penjelasan Singkat Kode Program.....	36

3.2.1	main()	36
3.2.2	cekCovid()	37
3.2.3	cekPasien()	38
3.2.4	inputDatabasePasien()	39
3.2.5	Char* lower(char str[])	40
3.2.6	Char* upper(char str[])	40
3.2.7	eliglibe()	41
3.2.8	signup()	42
3.2.9	read()	42
3.2.10	signin()	42
3.2.11	addAdmin()	43
3.2.12	lookupPasien()	43
3.2.13	printPasien()	43
3.2.14	updatePasien()	44
3.2.15	clear()	45
3.2.16	pause()	45
3.2.17	title()	45
3.3	Screenshot Run Program	45
BAB IV PENUTUP		50
4.1	Kesimpulan	50
4.2	Saran	50
DAFTAR PUSTAKA		51

BAB I

LATAR BELAKANG

1.1 Latar Belakang

Rumah sakit adalah salah satu tempat yang tidak akan pernah sepi dan luput dari berbagai aktivitas. Segala urusan seperti registrasi pasien, kunjungan pasien, semua dilakukan setiap harinya. Hal tersebut mengharuskan segala kegiatan dalam rumah sakit harus dilakukan secara terorganisir dan terstruktur.

Di era Covid-19 ini rumah sakit juga menjadi salah satu tempat yang rentan akan covid itu sendiri. Covid-19 adalah penyakit yang disebabkan oleh virus severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). Di Indonesia saat ini sedang gencarnya pandemic covid tersebut tersebar. Sehingga diperlukan suatu protokol yang membatasi beberapa orang dalam mengunjungi rumah sakit. Dengan suatu protokol tersebut memungkinkan persentase tersebarnya covid berkurang.

Kami sebagai programmer menciptakan suatu program yang dapat memenuhi segala aktivitas yang terstruktur dan terorganisir serta terhindar dari penyebaran covid-19 di area rumah sakit. Program tersebut merupakan program sederhana yang terdapat beberapa menu utama seperti menu untuk pengunjung dan admin. Menu pengunjung ini akan dibuat sebuah protocol covid dimana akan dilakukan pengecekan gejala terindikasi covid untuk menghindari penyebaran covid pada area rumah sakit. Sedangkan Menu Admin digunakan untuk mengontrol segala aktivitas seperti Read data pasien dan Update data pasien secara terstruktur dan terorganisir. Berikutnya, penulis ingin menyampaikan bagaimana program tersebut dibuat karena program tersebut akan sangat berguna apabila akan diimplementasikan kedepannya.

BAB II

LANDASAN TEORI

2.1 Tipe Data Primitive Bahasa C

Tipe data primitive pada Bahasa C adalah tipe data yang sudah terdefinisi dan disediakan dalam bahasa pemograman C. Dalam Bahasa C terdapat empat jenis tipe data yaitu int (integer), float, double, dan char (Character). Keempat tipe data tersebut adalah tipe data dasar yang memiliki arti dan fungsi masing-masing sebagai berikut,

2.1.1 Integer

Int atau yang lebih dikenal integer adalah tipe data dasar yang menyatakan bilangan bulat. Jangkauan yang dapat digunakan untuk tipe data int adalah dari -32768 s/d 32767. Format yang biasanya digunakan untuk tipe ini adalah “%d” tetapi bisa juga menggunakan “%i”. perbedaan antara %d dan %i adalah, %i akan menghasilkan output berupa nilai heksadesimal (bila input diawali 0x) atau oktal (diawali 0) sedangkan %d hanya menghapus 0 bila terdapat 0 di awal inputnya, sebagai contoh bila kita menggunakan %i di scanf dan printf lalu memasukan input 033 maka output yang dihasilkan adalah 27, sedangkan pada %d menghasilkan nilai 33.

2.1.2 Float

Float adalah tipe data dasar yang menyatakan bilangan pecahan. Jangkauan yang dapat digunakan untuk tipe data float adalah dari -3.4E38 s/d 3.4E+38. Format yang digunakan untuk tipe ini adalah “%f”. Contoh penggunaan dari tipe data float, saat kita akan memasukan berat badan.

2.1.3 Char

Char adalah tipe data dasar yang menyatakan Karakter/string. Jangkauan yang dapat digunakan untuk tipe data char adalah dari -128 s/d 127. Format yang digunakan untuk tipe ini adalah “%c” untuk satu karakter dan “%s” untuk beberapa karakter.

2.2 Operator Aritmatika, Pembandingan, dan Logika

Dalam bahasa C dapat juga dilakukan berbagai macam perhitungan dan pembandingan. Hal tersebut pada Bahasa C dikenal dengan operator aritmatika untuk menghitung dan operator pembandingan untuk membandingkan. Adapun bagian dari operator aritmatika yaitu Penjumlahan (+), Pengurangan (-), Perkalian (*), Pembagian (/), dan Sisa Bagi (%). Sedangkan bagian dari operator pembandingan yaitu Lebih Besar (>), Lebih Kecil (<), Sama Dengan (==), Tidak Sama dengan (!=), Lebih Besar Sama dengan (>=), dan Lebih Kecil Sama dengan (<=). Terakhir Adapun bagian dari operator logika yaitu AND (&&), OR(||), dan NOT (!).

2.3 Percabangan dalam Bahasa C

Percabangan adalah sebuah penyeleksian kondisi pada suatu program, dimana intruksi yang ada dalam program tersebut dijalankan/dieksekusi tidak secara sekuensial (berurut) melainkan dijalankan berdasarkan pada kondisi yang telah ditentukan. Kondisi menggunakan operator – operator Boolean yang hanya memberikan hasil true atau false seperti ==, >, <, >=, <=, !=, &, &&, |, ||. Adapun jenis jenis percabangan dalam bahasa C yang kami gunakan dalam program adalah sebagai berikut.

2.3.1 Percabangan If

Percabangan if adalah percabangan yang digunakan untuk mengontrol jalannya program dengan memperhatikan kondisi tertentu suatu pernyataan atau keadaan. Bila kondisi suatu keadaan bernilai true/benar/1 maka statement dalam program akan dieksekusi. Sedangkan bila kondisi yang diuji false/salah/0 maka program akan melewati pernyataan ini dan menjalankan pernyataan lain sampai mendapat kondisi bernilai true.

Percabangan If memiliki bentuk umum yaitu IF kondisi then {pernyataan yang dijalankan jika kondisi terpenuhi}, else {pernyataan

yang dijalankan jika kondisi tidak terpenuhi}. Dalam percabangan if memiliki empat kasus diantaranya adalah sebagai berikut,

- 1) Percabangan Tunggal, adalah percabangan yang hanya ada satu alternatif intruksi yang diperhatikan. Jadi ada kemungkinan *compiler* sama sekali tidak menjalankan sebuah intruksi apabila kondisi yang dihasilkan memberikan nilai *false*. Artinya dalam percabangan if tunggal hanya ada dua keadaan, bila kondisi intruksi tersebut *true* maka yang dijalankan adalah intruksi tersebut dan menghasilkan output dari *statement* yang telah dibuat, sebaliknya bila *false* maka program tidak akan menjalankan intruksi apapun dan tidak akan memberikan output. Contoh dari percabangan tunggal dalam aplikasi pemograman adalah sebagai berikut,
- 2) Percabangan Ganda, adalah percabangan dengan dua alternatif intruksi yang diperhatikan. Jadi *compiler* pasti menjalankan sebuah intruksi. Artinya dalam percabangan if ganda ada dua keadaan yang pasti salah satunya dijalankan. Bila kondisi intruksi tersebut *true* maka yang dijalankan adalah intruksi yang pertama dan melewati intruksi selanjutnya. Namun bila bernilai *false* maka akan melewati intruksi pertama dan menjalankan intruksi kedua. Contoh dari percabangan ganda dalam aplikasi pemograman adalah sebagai berikut,
- 3) Percabangan Lebih dari 2, adalah Percabangan dengan lebih dari dua alternatif intruksi. Artinya *compiler* dapat memperhatikan struktur seleksi secara tidak terbatas atau sampai keinginan yang diberikan batasan. Artinya nanti akan ada banyak *statement* berbeda dalam kondisi yang beragam. Contoh dari percabangan

ganda dalam aplikasi pemograman adalah sebagai berikut,

- 4) Nested If (Struktur If Bersyarat), adalah terdapat suatu percabangan dalam percabangan (if dalam if). Biasanya Nested If digunakan untuk menyelesaikan masalah-masalah yang memiliki lebih dari dua cabang. Dalam Nested If dapat menyediakan alternatif dari satu kondisi, dari beberapa kondisi, atau kombinasi keduanya. Contoh dari Nested If dalam aplikasi pemograman adalah sebagai berikut,

2.3.2 Percabangan Switch

Percabangan Switch adalah percabangan yang mirip dengan percabangan if, tetapi percabangan ini lebih baik digunakan saat terdapat banyak kondisi yang diperhatikan. Percabangan switch biasanya digunakan untuk menyederhanakan percabangan if else if dan Nested If yang sangat kompleks. Setiap case dalam percabangan switch harus diberikan kondisi break karena bila tidak program akan menjalankan dan mengeluarkan statement dalam intruksi atau kondisi selanjutnya dan akan berhenti sampai ada fungsi break. Kelemahan pada percabangan switch adalah kondisi yang diperiksa harus berupa data ordinal (bertipe integer atau char), dan tidak boleh bertipe real. Dalam percabangan switch kita akan mengenal yang namanya fungsi default, fungsi tersebut akan berjalan apabila input yang diberikan oleh user tidak terdapat pada salah satu case yang telah tersedia.

2.4 Perulangan dalam Bahasa C

Perulangan adalah sebuah kondisi yang dilakukan program untuk menjalankan satu atau beberapa pernyataan/aksi/instruksi secara berulang kali sebanyak yang diminta/diinput user. Struktur intruksi perulangan biasanya terdiri atas kondisi perulangan (suatu kondisi yang harus dipenuhi agar perulangan dapat terjadi), Badan perulangan/body (deretan

instruksi yang akan diulang-ulang pelaksanaannya), dan Pencacah perulangan/counter (suatu variabel yang nilainya harus berubah agar perulangan dapat terjadi dan pada akhirnya membatasi jumlah perulangan yang dapat dilaksanakan). Adapun jenis-jenis perulangan ada empat secara umum dibagi menjadi dua yaitu Counted Loop (for dan Nested loop) dan Uncounted Loop (while dan do-while). Counted Loop merupakan perulangan yang sudah ditentukan jumlah dalam melakukan suatu perulangan. Sedangkan Uncounted Loop, merupakan perulangan yang tidak jelas berapa kali ia harus mengulang.

2.4.1 For

For adalah suatu perintah perulangan proses yang telah diketahui seberapa banyak jumlah pengulangan yang akan dieksekusi (*Counted Loop*). For jika diibaratkan sebuah aktivitas, misalnya sit-up, maka sudah ditentukan berapa kali mau melakukan sit-up, misalnya 25 kali, berarti kita melakukan gerakan sit-up berulang-ulang sebanyak 25 kali.

Dari penulisan struktur pada kode program perintah perulangan yang ditentukan jumlahnya, penggunaan for lebih efisien karena susunannya tidak membingungkan dan tampak lebih sederhana. Bentuk umum dari perulangan for adalah for(inisialisasi; syarat; penambahan/counter). Inisialisasi adalah pernyataan keadaan awal dari variabel control. Syarat adalah suatu kondisi yang dinyatakan agar dapat keluar dari sebuah perulangan. Penambahan/counter adalah perubah variabel control sehingga variabel kontrol tidak lagi berada di keadaan awal.

2.4.2 Nested Loop (Perulangan Bersarang)

Perulangan bersarang adalah ada perulangan dalam perulangan, artinya bila perulangan pertama kita definisikan i dan perulangan didalamnya j maka setiap data i akan melakukan perulangan sebanyak yang diminta kemudian j selain melakukan perulangan dari yang diminta ia juga akan mengikuti banyak perulangan pada data i (total data perulangan = $i*j$). Perulangan ini juga termasuk kedalam *Counted Loop*.

2.4.3 While

Perulangan while adalah perulangan yang selalu akan mengecek kondisi di depan atau awal (sebelum mengulang). Perulangan while termasuk dalam *uncounted loops*, artinya proses perulangan akan terus berlanjut ketika kondisi masih bernilai *true* dan berhenti saat kondisi bernilai *false*. Perulangan while dapat diibaratkan kita melakukan sit up sampai kita bosan melakukannya. Perulangan while sebenarnya dapat dikategorikan ke *counted loops* karena perulangan dapat dihitung jumlah berapa kali melakukannya. Hal tersebut karena kita dapat menambahkan sebuah counter.

2.4.4 Do-While

Perulangan do-while mirip seperti perulangan while hanya saja perbedaannya terletak pada pengecekan kondisinya. Perulangan do-while akan selalu mengecek kondisi di belakang (sesudah mengulang), jadi setidaknya pada do-while akan selalu menghasilkan perulangan sekurang-kurangnya satu kali.

2.5 Array

Array/larik adalah struktur data dimana terdiri dari sekumpulan variabel dengan tipe yang sama. Array tersebut akan sangat memudahkan kita dalam membuat program yang membutuhkan sejumlah variabel dengan tipe yang sama sehingga kita tidak perlu mengakses ataupun mendeklarasikannya satu persatu.

Array berfungsi untuk menyimpan sekumpulan data pada program yang akan dijalankan. Membuat array dalam Bahasa C memiliki kemiripan dengan membuat sebuah variabel hanya saja dalam array kita harus menentukan panjangnya. Panjang array tersebut akan bergantung dari banyaknya data yang ditampung di dalamnya.

Pada sebuah array kita dapat mendeklarasikan dan juga mengakses elemen yang terdapat di dalam array tersebut. Elemen tersebut adalah sebuah indeks. Indeks inilah yang membuat kita mudah dalam memproses suatu array.

Mendeklarasikan array yang terdapat data yang sama satu persatu tentu tidak akan efisien, sehingga akan mudah apabila kita mendeklarasikannya sekaligus. Cara mendeklarasikannya adalah dengan menggunakan array. Cara mendeklarasikan sebuah array adalah seperti berikut : <type variable> <nama variable>[<jumlah>]. Jadi ketika kita ingin mencetak seluruh variabel sebanyak seratus data, kita hanya perlu memanggil indeks array tersebut dengan menggunakan perulangan for dengan batasan seratus, `for(i=0; i<100; i++){ printf("%d ", angka[i]); }`.

Selanjutnya dalam mengakses elemen array yang terdapat variabel-variabel didalamnya, kita gunakan indeks yang sebelumnya telah dijelaskan pada pendeklarasian array. Pada indeks tersebut kita berikan penomoran array dimana indeks pada kebanyakan kompiler dimulai dari 0 dan berakhir pada n-1. Jadi ketika kita ingin mengakses suatu data tertentu, kita cukup menuliskan indeks data tersebut dikurang 1. Tetapi dalam array kita juga dapat menginisialisasikan indeks dimulai dengan 1.

Array sendiri sebenarnya dapat kita lihat dari dimensinya. Ada array yang hanya 1 dimensi, 2 dimensi, bahkan lebih dari itu kita sebut multidimensi. Berikut penjelasan singkat mengenai dimensi setiap array.

2.6 Pointer

Dalam bahasa C diperlukan suatu pointer untuk mengakses data dengan alamat memori tertentu. Tetapi pointer tidak sesimpel itu, sehingga berikut pengertian tentang pointer dan bagiannya serta pentingnya penggunaan pointer dalam suatu kode program. Pointer adalah penunjuk memori yang disajikan dalam bentuk variabel dimana setiap variabel yang dibuat ini akan berisi alamat memori dari variabel lain. Pointer ini memanggil data atau mengakses data yang disimpan pada alamat tertentu dalam memori. Intinya ketika membuat sebuah variabel baru, pasti variabel tersebut memiliki alamat memori. Alamat memori itulah yang akan diakses dan disimpan oleh sebuah pointer, kemudian dapat digunakan untuk mengaksesnya kapan saja saat pointer digunakan.

2.7 Pelengkap dari Fungsi Operasi String dan Karakter

Dalam Bahasa C kita mengenal secara umum apa itu fungsi. Fungsi adalah sub-program yang bisa digunakan kembali baik di dalam program itu sendiri, maupun di program yang lain. Biasanya kita selalu menggunakan fungsi main, tetapi sebenarnya dalam Bahasa C terdapat berbagai macam jenis fungsi diantaranya Fungsi Operasi String, Fungsi Operasi Karakter, Fungsi Matematik, dan Fungsi buatan sendiri. Kali ini akan dibahas mengenai fungsi operasi string dan fungsi operasi karakter.

2.7.1 Pelengkap dari Fungsi Operasi String dan contohnya

Fungsi operasi string adalah suatu fungsi yang dapat digunakan untuk memanipulasi string sesuai kebutuhan. Fungsi operasi string dapat diakses dengan menggunakan library dalam c yaitu `<string.h>`. Dalam library tersebut terdapat beberapa pelengkap dari fungsi ini dimana dapat digunakan sesuai dengan kebutuhan seorang programmer. Supaya lebih paham berikut pelengkap serta contoh dari fungsi operasi string dalam Bahasa c.

- `Strcat`, adalah fungsi dengan dua string sebagai parameternya. String pertama disebut string tujuan sedangkan string kedua disebut string sumber. Fungsi ini menambahkan string sumber ke bagian akhir dari string tujuan. Bentuk umumnya `strcat(tujuan, sumber);`.
- `Strchr`, adalah operasi scanning string. Fungsi akan mereturn lokasi dari found character, atau pointer null jika karakter tidak ditemukan. Contohnya biasa digunakan untuk mencari sebuah teks (string) di dalam string. Biasanya fungsi ini dipadukan dengan fungsi `strncpy()` untuk mencari dan mengubah isi teks.
- `Strcmp`, adalah fungsi yang membandingkan dua buah string dengan bentuk umum `strcmp(string1, string2);`. Kedua buah string dalam fungsi ini =diteruskan sebagai parameter kemudian dikembalikan dengan nilai +ve apabila

`string1 > string2`, dikembalikan 0 apabila `string1 = string2`, dan `-ve` apabila `string1 < string2`. Contoh biasanya digunakan untuk pencocokan password.

- `Strcpy`, seperti namanya yaitu cpy string ini berfungsi untuk mengcopy atau menyalin sebuah string asal ke string lainnya. Bentuk umumnya `strcpy(var_tujuan, string_asal);`. Contohnya biasanya digunakan untuk perpaduan dengan fungsi operasi string lainnya.
- `Strlen`, seperti namanya `len` atau `length` artinya fungsi ini menerima string sebagai parameter dan mengembalikannya dalam bentuk integer dari panjang sebuah string. Intinya yang dihitung adalah apa saja karakter yang terdapat dalam suatu string, termasuk dengan konstanta `\n`. Bentuk umum `strlen(string);`. Contoh dalam aplikasi biasanya digunakan untuk membatasi jumlah karakter pada suatu inputan.
- `Strncat`, memiliki fungsi yang sama dengan `strcat`, perbedaannya adalah jumlah karakter yang dipindahkan. Fungsi ini dapat memindahkan `n` karakter sesuai yang dibuat programmer. Bentuk umum `strncat(tujuan, sumber, n);`. Dengan `n` adalah jumlah karakter yang ingin dipindahkan. Contohnya dipakai dalam bahasa pemrograman untuk keperluan menampung dan memanipulasi data teks, misalnya untuk menampung (menyimpan) suatu kalimat.
- `Strncmp`, memiliki fungsi yang sama dengan `strcmp`, perbedaannya adalah dapat menentukan jumlah `n` karakter yang ingin dibandingkan. Bentuk umum `Strncmp(string1, string2, n)`. Contohnya sama dengan `strncmp`, yaitu melakukan perbandingan antar kata secara case sensitive. Misal, Jika isi variable `$newpassword` berbeda dengan `$confirmpassword` maka fungsi dari `strcmp` adalah melihat apakah kedua variable tersebut sama atau tidak.

- Strncpy, memiliki fungsi yang sama dengan strcpy, perbedaannya adalah dapat mengcopy string asal ke string lainnya sebanyak n karakter yang diinginkan. Bentuk umumnya: strncpy(var_tujuan, string_asal, n);.
- Strchr, adalah fungsi yang Mengembalikan pointer ke kejadian terakhir dari karakter dalam string. Bentuk umum char *strchr(const char *str, int c);. str adalah string dan c adalah karakter yang akan berlokasi.
- Strcmpi (string 1, string2), mirip dengan strcmp, perbedaannya fungsi ini mengabaikan karakter yang hurufnya kapital atau kecil. Contohnya ketika melakukan pemograman di website untuk jawaban dari kuis-kuis Oase.
- Strstr (), fungsi ini digunakan ketika ingin mencari sebuah string dalam string. Contoh biasanya digunakan dengan perpaduan strncpy untuk mencari lalu mengganti kata, istilahnya typo.
- Strlwr (string), fungsi ini menerima satu kata yang bisa berupa huruf kecil, huruf kapital, atau dua-duanya. Fungsi ini akan mengkonvert huruf-huruf tersebut menjadi lower case atau huruf kecil.
- Strupr (string), kebalikan dari fungsi strlwr. Fungsi ini akan mengkonvert string menjadi upper case atau huruf kapital. Biasanya digunakan untuk suatu formulir yang mengharuskan untuk mengeluarkan output yang huruf capital, misalnya ketika saat membuat sertifikat dengan mengcopy nama-nama yang sudah diberi fungsi ini sehingga tidak perlu lelah untuk mengubahnya lagi menjadi huruf-huruf capital.

2.7.2 Pelengkap dari Fungsi Operasi karakter dan contohnya

Fungsi operasi karakter adalah suatu fungsi yang dapat digunakan untuk memanipulasi karakter sesuai kebutuhan. Fungsi operasi karakter dapat diakses dengan menggunakan library dalam c yaitu `<ctype.h>`. Dalam library ini terdapat beberapa pelengkap dari fungsi ini dimana dapat digunakan sesuai dengan kebutuhan seorang programmer. Contohnya fungsi ini digunakan ketika membuat program khusus untuk satu karakter, bisa berupa program mengulang, atau digunakan sebagai nilai true false. Supaya lebih paham berikut pelengkap serta contoh dari fungsi operasi karakter dalam Bahasa c.

- `Isalpha`, adalah fungsi yang akan mereturn nilai bukan nol (0) apabila bernilai benar. Yaitu argument dalam syntax merupakan karakter yang berupa huruf dalam alphabet adalah bernilai benar. Bentuk umum `isalpha(char);`.
- `Isdigit`, adalah fungsi yang akan mereturn nilai bukan nol (0) apabila bernilai benar. Yaitu argument dalam syntax merupakan karakter yang berupa angka dari 0 sampai 9 adalah bernilai benar. Bentuk umum `isdigit(char);`.
- `Islower`, fungsi yang bernilai benar atau bukan 0 apabila karakter berupa huruf kecil. Bentuk umum `islower(char);`.
- `Isupper`, fungsi yang bernilai benar atau bukan 0 apabila karakter berupa huruf kapital. Bentuk umum `isupper(char);`.
- `Isspace`, fungsi yang bernilai benar atau bukan 0 apabila karakter berupa spasi(space), newline tab, dan tab. Bentuk umum `isspace(char);`.
- `Ispunct`, fungsi yang bernilai benar atau bukan 0 apabila karakter berupa tanda baca (punctuation), tidak termasuk karakter spasi, huruf alphabet, dan angka (digit).. Bentuk umum `ispunct(char);`.

- Tolower, mirip dengan fungsi `strlwr` dimana akan mengkonvert huruf capital menjadi huruf kecil. Perbedaannya jika dalam fungsi operasi karakter yang diubah hanya satu karakter saja. Bentuk umum `Tolower(char);`. Biasanya digunakan dalam pemograman Bahasa C untuk mengulang program dengan pilihan y/t.
- Toupper, mirip dengan fungsi `strupr` dimana akan mengkonvert huruf kecil menjadi huruf kapital. Perbedaannya jika dalam fungsi operasi karakter yang diubah hanya satu karakter saja. Bentuk umum `Toupper(char);`. Biasanya digunakan dalam pemograman Bahasa C untuk mengulang program dengan pilihan Y/T.

2.8 Call by Value and Call by Reference

2.8.1 Cara Melewatkan Parameter

Dalam Bahasa C untuk melewati parameter dapat dilakukan dengan dua cara. Cara pertama yaitu dengan pemanggilan dengan nilai (`call by value`) dan kedua yaitu pemanggilan dengan referensi (`call by reference`). Sebelum menjelaskan cara melewati parameter kita perlu mengetahui apa itu parameter. Parameter adalah variabel yang menyimpan nilai untuk diproses di dalam fungsi dimana dapat menyertai fungsi baik dalam keadaan saat deklarasi maupun saat pemanggilan fungsi. Jenis parameter ada dua yaitu pertama, Parameter Formal merupakan variabel yang ada pada daftar parameter dalam definisi fungsi sedangkan kedua, Parameter Aktual merupakan variabel yang dipakai dalam pemanggilan fungsi. Lalu apa itu melewati parameter? Melewatkan parameter berarti kita menggunakan atau mengolah parameter tersebut melalui isi program yang ada di dalam fungsi.

Cara untuk melewati parameter yang pertama adalah dengan Pemanggilan dengan Nilai (`Call by Value`). Pemanggilan dengan Nilai adalah pemanggilan yang akan memengaruhi dari kedua parameter formal

dan aktual. Cara ini menyalin nilai dari parameter aktual ke parameter formal dan sama sekali tidak mengubah nilai pada parameter aktual tersebut. Parameter aktual tidak akan berubah meskipun nilai pada parameter formal berubah-ubah. Ketika kita melewatkan parameter menggunakan cara Pemanggilan dengan Nilai ini, yang dikirimkan ke dalam fungsi adalah nilai dari data bukan alamat memori dari data tersebut. Fungsi yang menerima kiriman nilai menyimpannya di alamat yang berbeda dengan nilai aslinya yang digunakan oleh bagian program yang memanggil fungsi. Pengiriman nilai dapat dilakukan dengan ungkapan, variabel, elemen array, dan konstanta. Berikut contoh dari melewatkan parameter menggunakan cara Pemanggilan dengan nilai.

2.8.2 Cara Pemanggilan Secara Referensi

Cara untuk melewatkan parameter yang kedua adalah dengan Pemanggilan Secara Referensi (Call by Reference). Pada pemanggilan ini berupaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi, sehingga dengan cara ini dapat mengubah isi sebuah variabel diluar fungsi dengan membuat programnya didalam fungsi. Karena digunakan untuk melewatkan alamat maka diperlukan variabel pointer untuk menunjuk ke variabel lain.

Ketika kita melewatkan parameter menggunakan cara Pemanggilan secara referensi ini, yang dikirimkan ke fungsi adalah alamat yang menunjuk nilai datanya, bukan nilai data itu sendiri. Alamat yang digunakan fungsi yang menerima kiriman adalah sama untuk mendapat nilai datanya. Nilai asli dalam program yang memanggil perubahan nilai di fungsi ini akan berubah nilainya, parameter formal berubah maka parameter actual mengikuti perubahan parameter formal tersebut. Cara ini adalah cara dengan pengiriman dua arah, yaitu ketika pemanggil ke fungsi yang dipanggil dan fungsi yang dipanggil ke pemanggil. Berikut contoh dari melewatkan parameter menggunakan cara Pemanggilan Secara Referensi.

2.9 Operasi File dalam Bahasa Pemograman

Dalam bahasa pemograman kita tidak hanya mengenal cara mengoding untuk membuat suatu program, melainkan kita juga dapat mengoding untuk mengoperasikan file dengan bahasa pemograman yang telah kita pelajari, baik itu java, python, C, dll. Mengoperasikan file maksudnya bagaimana kita memanipulasi, menginput, atau mengoutputkan suatu data pada file misalnya seperti membuka file, merekam data ke file, membaca data dari file, dan menutup file. Operasi file ini dapat dilakukan di berbagai macam ekstension, bisa menggunakan Bahasa Python dengan ekstensi .py, Java .java, dan Bahasa yang sering dan akan kita bahas yaitu Bahasa C dengan ekstensi .c. Berikut hal yang perlu diperhatikan dalam Mengoperasikan file dalam program.

2.9.1 Membuka file

Sebelum memasukan data ke file atau membacanya tentunya kita perlu untuk membuka file. Dalam bahasa pemograman untuk membuka file kita dapat menggunakan pustaka `fopen()`. `fopen()` adalah pustaka yang digunakan ketika kita ingin membaca atau menulis data, dimana pustaka ini akan mereturn suatu nilai pointer dalam bentuk `FILE`. Berikut syntax atau operasi file menggunakan pustaka `fopen` :

```
FILE *fopen( const char*nama_file, const char*mode);
```

Operasi diatas adalah salah satu operasi file sebelum kita akan memanipulasi filenya. `Nama_file` adalah file yang dibuat dan ditentukan oleh programmer sendiri. Kemudian sebelum membuka tentunya kita perlu untuk membuat suatu file, dimana dapat kita buat dengan menambahkan mode diakhir. Berikut jenis-jenis mode yang ada pada operasi file. Pada tabel b merupakan data dimanipulasi dalam bentuk biner. Khusus untuk mode `r`, `rb`, dan `rb+` ketika file yang ditunjuk tidak tersedia maka program akan mereturn nilai `NULL`. Selain mode tersebut, ketika tidak ada file yang ditunjuk, maka program akan membuat suatu file baru.

Mode			Deskripsi	Starts
r	rb		Dibuka untuk dibaca	Awal
w	wb		Dibuka untuk ditulis (akan membuat file baru jika tidak ada file yang dituju). Menghapus isi dan overwrites file.	Awal
a	ab		Dibuka untuk menambah data baru diakhir file (akan membuat file baru jika tidak ada file yang dituju).	Akhir
r+	rb+	r+b	Dibuka untuk dibaca dan ditulis	Awal
w+	wb+	w+b	Dibuka untuk dibaca dan ditulis. Menghapus isi dan overwrites file.	Awal
a+	ab+	a+b	Dibuka untuk dibaca dan ditulis (akan menambahkan data sebelumnya pada file jika file yang dituju sudah ada).	Akhir

Sebagai contoh untuk syntax dan penggunaan mode read diatas adalah sebagai berikut:

```
// membuat pointer
```

```
File *fptr;
```

```
// membuka file
```

```
fptr = fopen("namafile.txt", "r");
```

2.9.2 Menutup file

Setelah Membuka tentu kita juga perlu menutup file yang telah dibuka. Biasanya file ditutup ketika file tidak lagi digunakan, program telah berakhir, atau ketika ingin membuka file lain lagi. Fungsi pustaka yang digunakan adalah `fclose()`. Bentuk umum dari fungsi satu ini adalah `int fclose(FILE *ponter-file);`

`fclose ()` akan membatasi koneksi dengan file kemudian akan membebaskan pointer dari file tersebut. `fclose` ini dapat digunakan ketika kita ingin membuka file lalu menulis kemudian ingin membaca data yang ditulis tersebut. `fclose` digunakan sebelum kita akan membaca data yang baru ditambahkan, supaya data baru dapat tersimpan dan terbaca. Hal penting yang harus diketahui adalah fungsi pustaka satu ini akan terpanggil secara otomatis untuk setiap file yang terbuka ketika program akan di akhiri. Terakhir untuk menutup suatu file maka file yang dibuat harus mencapai akhir dari file, sehingga untuk mendeteksi akhir dari file tersebut kita dapat mendefinisikan suatu fungsi yang bernama fungsi EOF (End of File). Ketika program sama dengan EOF maka operasi penutupan file tidak dapat dilakukan, begitupula sebaliknya.

2.9.3 Meletakkan Data ke Penyangga

Data dalam Bahasa C khususnya dapat bertipe karakter, integer, string, terformat, dan blok data untuk diletakkan kedalam penyangga (*buffer*). Dalam mengoperasikan suatu file terdapat beberapa pustaka untuk meletakkan data ke buffer diantaranya adalah `fputc()`, `fgetc()`, `putw()`, `getw()`, `fputs()`, `fgets()`, `fprintf()`, `fscanf()`, `fwrite()`, `fread()`. Yang termasuk dalam input data file adalah `fgetc()`, `getw()`, `fgets()`, `fscanf()`, dan `fwrite()`, lain daripada itu adalah termasuk dalam output data file. Mengoperasikan file dalam program kita cukup untuk menggunakan pustaka `fopen()`, `fget()`, dan `fputs()` karena penggunaannya mudah dipahami. Penjelasan untuk pustaka-pustaka diatas akan dijelaskan pada sub bab materi berikutnya.

2.9.4 Manipulasi File

Manipulasi file dalam operasi file adalah bagaimana kita melakukan sesuatu dengan file pada program tanpa melibatkan proses input output file. Manipulasi file dapat kita lakukan dengan mengecek keberadaan file, mengganti nama file, dan menghapus file. Berikut penjelasan lebih lanjut dalam memanipulasi file.

- Mengecek Keberadaan File, dapat kita lakukan dengan cara menggunakan pustaka `int access(const char* path, int amode;` `const char* path` menunjukkan dimana file itu berada. Sedangkan `amode` ini bukan mode operasi file seperti sebelumnya melainkan sebuah pemeriksaan lebih lanjut kepada file yang dicari. Ada beberapa jenis `amode` diantaranya, `amode 0` menunjukkan hanya akan memeriksa keberadaan file di disk, `amode 2` menunjukkan pemeriksaan apakah file dapat ditulis, `amode 4` menunjukkan pemeriksaan apakah file dapat dibaca, `amode 6` menunjukkan pemeriksaan apakah file dapat dibaca dan ditulis.
- Mengganti nama file, dapat kita lakukan dengan pustaka Fungsi `rename()`, artinya kita dapat mengganti nama file yang sudah ada pada disk sebelumnya. Bentuk umum untuk fungsi ini adalah, `int rename(const char * nama_lama, const char * nama_baru;`
- Menghapus file, dapat kita lakukan dengan pustaka `unlink()`, artinya kita akan menghapus file yang telah ada pada memory. Bentuk umum untuk fungsi ini adalah `int unlink(const char * path;`

2.10 Fungsi-Fungsi dari File Operasi.

Sebelumnya telah kita singgung fungsi-fungsi dari file operasi pada sub sub bab materi Meletakkan data ke penyangga. Dalam meletakkan data

yang dapat berbentuk lima macam tersebut ke dalam buffer kita dapat menggunakan fungsi sebagai berikut,

2.10.1 fputc()

fputc adalah fungsi pustaka yang digunakan ketika kita akan meletakkan data kedalam penyangga dan merecord nya ke dalam file. Bentuk umum dari fungsi pustaka ini adalah `int fputc(int char, FILE *stream);` parameter char artinya data yang direcord harus bertipe character, sedangkan stream artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi fputc() ini dalam program tidak akan menghasilkan output karena output sudah diarahkan kepada file yang dituju dan karena tidak diberikan fungsi read file.

2.10.2 fgetc()

fgetc adalah fungsi pustaka yang digunakan ketika kita akan membaca satu buah karakter untuk ditampilkan ke buffer. Bentuk umum dari fungsi pustaka ini adalah `int fgetc(FILE *stream);` parameter stream artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi fgetc() ini dalam program akan menghasilkan output sesuai dengan modifikasi kita dalam menggunakannya output yang dimodifikasi dapat bermacam-macam intinya dengan fungsi ini data yang ada pada file akan dibaca karena pada program diberikan fungsi read file.

2.10.3 putw()

putw adalah fungsi pustaka yang mirip fungsinya dengan fputc() hanya saja data yang diletakkan berbentuk integer. Bentuk umum dari fungsi pustaka ini adalah `putw(int number, FILE *fp);` parameter int artinya data yang direcord harus bertipe integer, sedangkan stream artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi putw() ini dalam program tidak akan menghasilkan output karena output sudah diarahkan kepada file yang dituju dan karena pada program tidak diberikan fungsi read file.

2.10.4 getw()

getw adalah fungsi pustaka yang mirip dengan fungsi fgetc hanya saja pada fungsi getw yang dibaca adalah sebuah nilai integer untuk ditampilkan ke buffer. Bentuk umum dari fungsi pustaka ini adalah `int getw(FILE *fp);` Fungsi getw() ini dalam program akan menghasilkan output sesuai dengan data yang ada pada file karena pada program diberikan fungsi read file.

2.10.5 fputs()

fputs adalah fungsi pustaka yang digunakan mirip dengan fputc dan putw tetapi perbedaannya adalah data yang diletakkan kedalam penyangga bernilai string. Bentuk umum dari fungsi pustaka ini adalah `int fputs (const char * str, FILE * stream);` parameter str artinya data yang direcord bertipe string yaitu array yang berisi urutan karakter, sedangkan stream artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi fputs() ini dalam program tidak akan menghasilkan output karena output sudah diarahkan kepada file yang dituju dan karena tidak diberikan fungsi read file.

2.10.6 fgets()

fgets adalah fungsi pustaka mirip dengan fgetc dan getw bedanya fungsi ini digunakan ketika kita akan membaca suatu nilai string untuk ditampilkan ke buffer. Bentuk umum dari fungsi pustaka ini adalah `char *fgets(char *str, int n, FILE *stream)` parameter str artinya menyimpan string untuk dibaca, n berarti jumlah karakter yang akan dibaca, sedangkan stream artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi fgets() ini dalam program akan menghasilkan output sesuai dengan data yang ada pada file, data akan dibaca karena pada program diberikan fungsi read file.

2.10.7 fprintf()

fprintf adalah fungsi pustaka yang digunakan ketika kita akan meletakkan data terformat kedalam penyangga dan merecord nya ke dalam file. Bentuk umum dari fungsi pustaka ini adalah `int fprintf(FILE *stream, const char *format, ...);` parameter stream artinya sebuah

pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi, sedangkan format ini berarti kita membuat suatu kalimat dengan beberapa kata dengan setiap kata adalah satu format. Fungsi `fprintf()` ini dalam program tidak akan menghasilkan output karena output sudah diarahkan kepada file yang dituju dan karena tidak diberikan fungsi `read file`.

2.10.8 `fscanf()`

`fscanf` adalah fungsi pustaka yang digunakan ketika kita akan membaca data terformat untuk ditampilkan ke buffer. Bentuk umum dari fungsi pustaka ini adalah `int fscanf(FILE *stream, const char *format, ...)`; parameter `stream` artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi, sedangkan Format Ini adalah C string yang berisi item berikut dalam satu atau lebih dari: karakter ruang, karakter non-ruang dan Format specifier. Fungsi `fscanf()` ini dalam program akan menghasilkan output sesuai dengan data yang ada pada file, data akan dibaca karena pada program diberikan fungsi `read file`.

2.10.9 `fwrite()`

`fwrite` adalah fungsi pustaka yang digunakan ketika kita akan meletakkan sebuah blok data kedalam penyangga dan merecord nya ke dalam file. Bentuk umum dari fungsi pustaka ini adalah `size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream)`; parameter `ptr` artinya pointer yang akan ditulis elemen dari array, `size_t` artinya ukuran untuk setiap elemen, `nmemb` artinya jumlah elemen, sedangkan `stream` artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi `fwrite()` ini dalam program tidak akan menghasilkan output karena output sudah diarahkan kepada file yang dituju dan karena tidak diberikan fungsi `read file`.

2.10.10 `fread()`

`fread` adalah fungsi pustaka yang digunakan ketika kita akan membaca sebuah struktur data untuk ditampilkan ke buffer. Bentuk umum

dari fungsi pustaka ini adalah `size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);` parameter `ptr` artinya pointer yang akan ditulis elemen dari array, `size_t` artinya ukuran untuk setiap elemen, `nmemb` artinya jumlah elemen, sedangkan `stream` artinya sebuah pointer yang menunjuk kepada file yang telah kita buat atau ingin kita manipulasi. Fungsi `fread()` ini dalam program akan menghasilkan output sesuai dengan data yang ada pada file, data akan dibaca karena pada program diberikan fungsi `read` file.

2.10.11 `fseek()` dan `feof()`

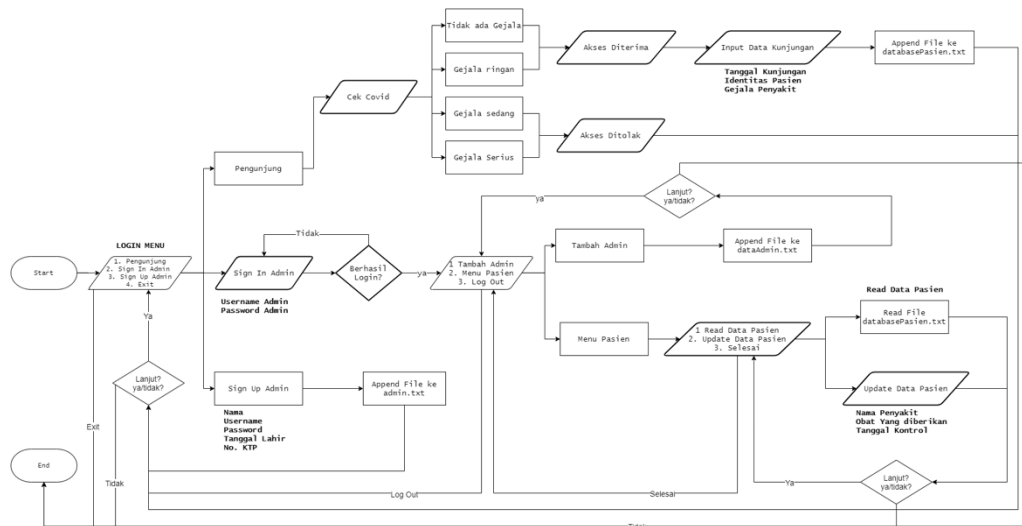
Selain sepuluh fungsi yang terdapat diatas, terdapat juga dua fungsi lain yang sebenarnya jarang digunakan dan dimanfaatkan. Yang pertama yaitu `fseek()`, digunakan untuk mengakses data secara random. Bentuk umum untuk `fseek()` adalah `int fseek(FILE *stream, long int offset, int whence);` Sedangkan `feof()` adalah fungsi yang digunakan untuk menampilkan isi file. Tetapi lebih berfungsi untuk mendeteksi akhir file. Bentuk umum untuk `feof()` adalah `int feof(FILE *stream);`

BAB III

DESAIN DAN METODE

3.1 Flowchart Program

Gambar 3.1 Flowchart Program



3.2 Metode Program

Pada program ini, digunakan seluruh materi yang terdapat pada modul 1 hingga modul 5. Pertama, pada modul 1 mengenai struktur dasar dalam bahasa c serta input-output. Pada program ini, digunakan beragam perintah, seperti printf(), scanf(), serta fgets(). Ketiga perintah ini digunakan hampir di setiap menu dan fungsi yang ada pada program. Selanjutnya, pada modul 2 mengenai penyeleksian kondisi dan perulangan. Pada penyeleksian kondisi, kami menggunakan if else serta switch case. Switch case kami gunakan hampir di setiap bagian yang memerlukan pilihan menu, sedangkan if else di setiap fungsi yang memerlukan syarat tertentu. Lalu mengenai perulangan, kami menggunakan perulangan for, while do, serta do while. Perulangan for kami gunakan salah satunya pada fungsi cekCovid untuk menanyakan kondisi pasien saat ini yang dilanjutkan dengan operasi aritmatika untuk menghitung persentase indikasi seorang pasien terkena

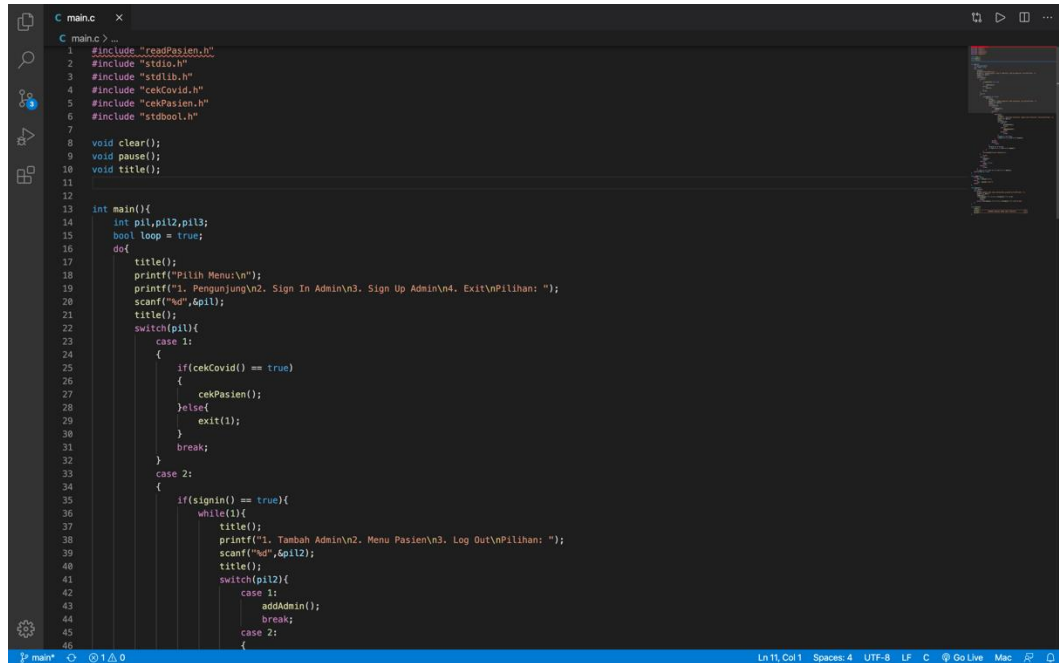
COVID-19. Selanjutnya, while do kami gunakan salah satunya pada fungsi pause() yang akan terus mengulang perintah yang terkandung di dalamnya selama nilai yang diinputkan user sama dengan 1. Lalu do while kami gunakan di menu utama dimana menu utama akan dieksekusi sekali terlebih dahulu dan akan terus dieksekusi selama loop masih bernilai true. Untuk modul 3 sendiri mengenai array, kami menggunakan array hampir di setiap file. Salah satunya di file cekPasien.c. Kami menggunakan array untuk memberikan batas maksimum karakter yang dapat ditampung dalam sebuah variabel char. Untuk modul 4 mengenai pointer dan fungsi. Pada program kami, kami menggunakan beragam fungsi dan mengeksekusinya di dalam fungsi main. Fungsi-fungsi yang kami buat ada yang berada di dalam file main.c maupun modular di file-file terpisah. Terkait modul 4, kami juga menggunakan operasi string pada berbagai fungsi kami, seperti strcpy serta strcmp untuk mengcopy data string serta membandingkan data string yang diinputkan oleh user. Sedangkan pointer, kami menggunakan pointer sebagian besar untuk operasi file pada modul 5. Di modul 5 terkait operasi file, kami menggunakan operasi file untuk mencetak data-data ke dalam database dalam file txt seperti database pasien, admin, dan lain sebagainya.

BAB III

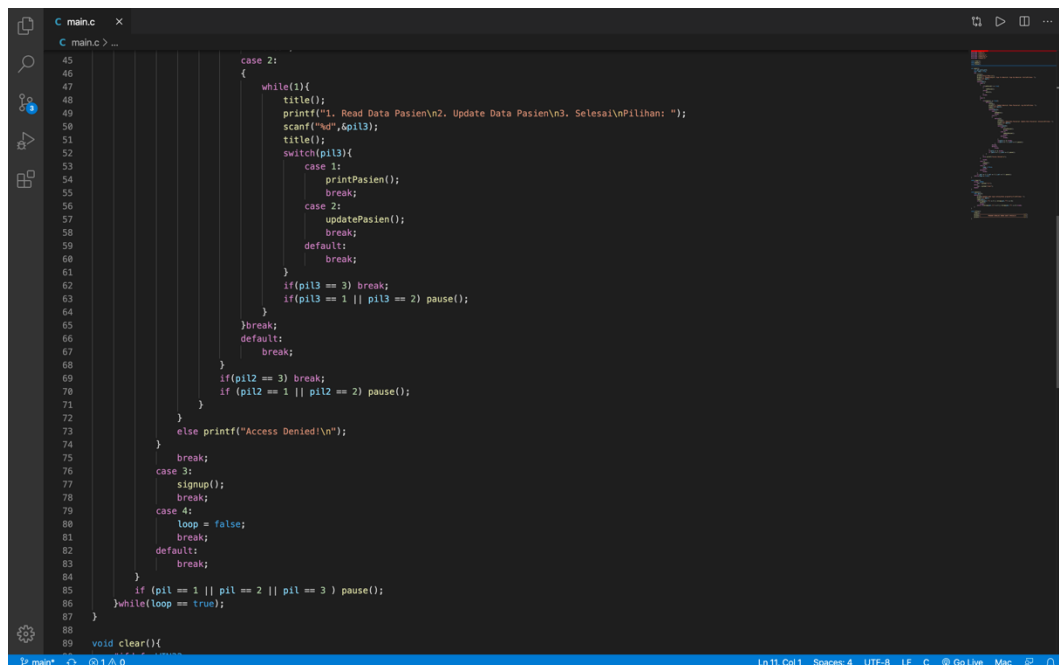
HASIL DAN IMPLEMENTASI

3.1 Screenshot Kode Program

3.1.1 File main.c



```
C main.c x
C main.c > ...
1 #include "readPasiem.h"
2 #include "stdio.h"
3 #include "stdlib.h"
4 #include "cekCovid.h"
5 #include "cekPasiem.h"
6 #include "stdbool.h"
7
8 void clear();
9 void pause();
10 void title();
11
12
13 int main(){
14     int pil,pil2,pil3;
15     bool loop = true;
16     do{
17         title();
18         printf("Pilih Menu:\n");
19         printf("1. Pengunjung\n2. Sign In Admin\n3. Sign Up Admin\n4. Exit\nPilihan: ");
20         scanf("%d",&pil);
21         title();
22         switch(pil){
23             case 1:
24             {
25                 if(cekCovid() == true)
26                 {
27                     cekPasiem();
28                 }else{
29                     exit(1);
30                 }
31                 break;
32             }
33             case 2:
34             {
35                 if(signin() == true){
36                     while(1){
37                         title();
38                         printf("1. Tambah Admin\n2. Menu Pasiem\n3. Log Out\nPilihan: ");
39                         scanf("%d",&pil2);
40                         title();
41                         switch(pil2){
42                             case 1:
43                             {
44                                 addAdmin();
45                                 break;
46                                 case 2:
```



```

47                             {
48                                 while(1){
49                                     title();
50                                     printf("1. Read Data Pasiem\n2. Update Data Pasiem\n3. Selesai\nPilihan: ");
51                                     scanf("%d",&pil3);
52                                     title();
53                                     switch(pil3){
54                                         case 1:
55                                         {
56                                             printPasiem();
57                                             break;
58                                         case 2:
59                                         {
60                                             updatePasiem();
61                                             break;
62                                         default:
63                                         {
64                                             break;
65                                         }
66                                     }
67                                     if(pil3 == 3) break;
68                                     if(pil3 == 1 || pil3 == 2) pause();
69                                 }break;
70                             }
71                             default:
72                             {
73                                 break;
74                             }
75                         }
76                         if(pil2 == 3) break;
77                         if (pil2 == 1 || pil2 == 2) pause();
78                     }
79                     else printf("Access Denied\n");
80                 }
81                 break;
82             case 3:
83             {
84                 signup();
85                 break;
86             case 4:
87             {
88                 loop = false;
89                 break;
90             default:
91             {
92                 break;
93             }
94         }
95         if (pil == 1 || pil == 2 || pil == 3 ) pause();
96     }while(loop == true);
97 }
98
99 void clear(){
```

```

C main.c X
C main.c > ...
78         break;
79     case 4:
80         loop = false;
81         break;
82     default:
83         break;
84     }
85     if (pil == 1 || pil == 2 || pil == 3) pause();
86     while(loop == true);
87 }
88
89 void clear(){
90     #ifdef _WIN32
91         std : system("cls");
92     #else
93         std : system("clear");
94     #endif
95 }
96
97 void pause(){
98     char pil[2];
99     while(1){
100         printf("Apakah anda ingin melanjutkan program?(y/t)\nPilihan: ");
101         scanf("%s",pil);
102         lower(pil);
103         if(strcmp(pil,"t") == 0 || strcmp(pil,"T") == 0){
104             exit(1);
105             break;
106         }else if(strcmp(pil,"y") == 0 || strcmp(pil,"Y") == 0) break;
107     }
108 }
109
110 void title(){
111     clear();
112     printf("
113     PROGRAM SIMULASI RUMAH SAKIT SPESIALIS
114     ");
115 }

```

3.1.2 File cekCovid.c

```

C cekCovid.c X
C cekCovid.c > cekCovid()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include "stdbool.h"
5
6  bool cekCovid()
7  {
8
9      int y, i, a, b, c;
10     float GejalaT1, GejalaT2, GejalaT3, sum;
11     char persen = '%';
12     char sym1[100][255] = {"1. Suhu diatas 37 derajat", "2. Batuk kering", "3. Kelelahan"};
13     char sym2[100][255] = {"4. Rasa tidak nyaman & nyeri", "5. Nyeri Tenggorokan", "6. Diare", "7. Mata merah",
14     "8. Sakit kepala", "9. Hilang Indra perasa atau penciuman", "10. Perubahan warna pada jari"};
15     char sym3[100][255] = {"11. Sesak nafas", "12. Nyeri dada", "13. Hilangnya kemampuan berbicara atau bergerak"};
16
17     printf("KETERANGAN\n");
18     printf("Masukkan 1 Apabila pernah atau sedang mengalami gejala berikut ");
19     printf("\nMasukkan 0 Apabila tidak pernah atau tidak sedang mengalami gejala berikut\n");
20     printf("GEJALA COVID\n");
21
22     y = 0;
23     for (i = 0; i < 3; i++)
24     {
25         printf("%s = ", sym1[i]);
26         scanf("%d", &a);
27         if (a == 1)
28             y++;
29     }
30     GejalaT1 = (100 / 3) * y * 0.2;
31
32     y = 0;
33     for (i = 0; i < 7; i++)
34     {
35         printf("%s = ", sym2[i]);
36         scanf("%d", &b);
37         if (b == 1)
38             y++;
39     }
40     GejalaT2 = (100 / 7) * y * 0.3;
41
42     y = 0;
43     for (i = 0; i < 3; i++)
44     {
45         printf("%s = ", sym3[i]);
46         scanf("%d", &c);

```

```

C cekCovid.c X
C cekCovid.c > cekCovid()
42 y = 0;
43 for (i = 0; i < 3; i++)
44 {
45     printf("%s = ", sym3[i]);
46     scanf("%d", &c);
47     if (c == 1)
48         y++;
49 }
50 GejalaT3 = (100 / 3) * y + 0.5;
51
52 sum = GejalaT1 + GejalaT2 + GejalaT3;
53
54 //getch();
55 // system("cls");
56
57 printf("HASIL CEK GEJALA COVID\n");
58 printf("Persentase terindikasi covid adalah %.2f %c\n", sum, persen);
59
60 if (sum <= 10)
61 {
62     printf(">>Tidak terindikasi covid/Sehat.\n");
63 }
64
65 else if (sum <= 30)
66 {
67     printf(">>Terindikasi gejala ringan!\n");
68     printf(">>Harap melakukan isolasi dan perawatan mandiri di rumah!\n");
69 }
70
71 else if (sum <= 60)
72 {
73     printf(">>Terindikasi gejala sedang!\n");
74     printf(">>Harap segera melakukan pemeriksaan!\n");
75 }
76
77 else
78 {
79     printf(">>Terindikasi gejala serius!\n");
80     printf(">>Anda memiliki potensi terkena covid!\n>>Harap segera melakukan pemeriksaan!\n");
81 }
82
83 if (sum <= 30)
84 {
85     printf(">>Akses masuk diterima!\n");
86     printf("\nSilahkan lanjut ke tahap berikutnya.\n");

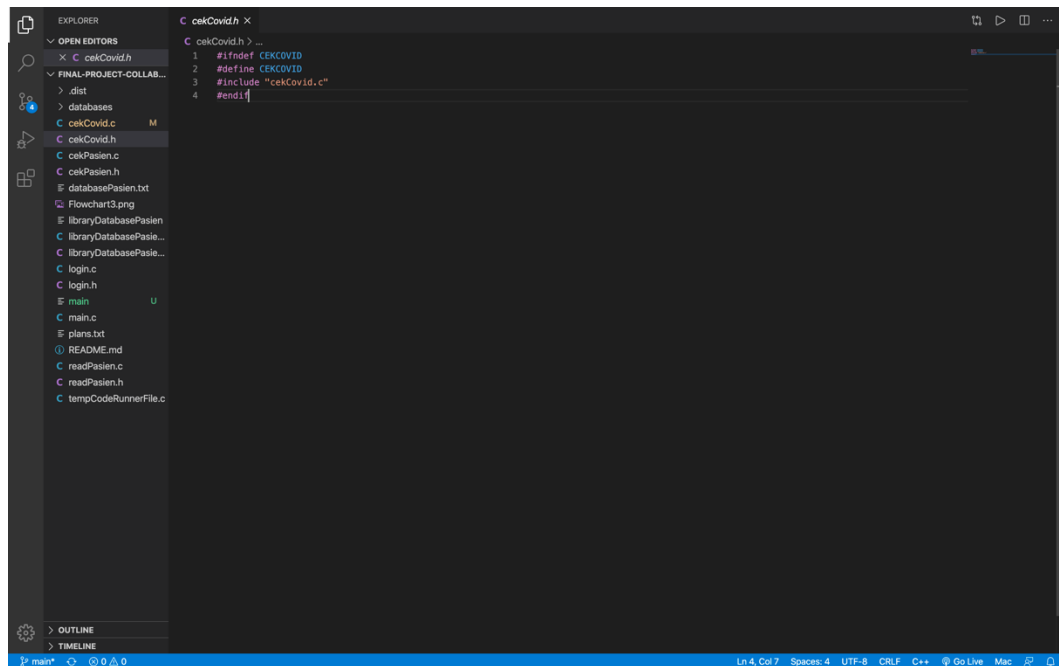
```

```

C cekCovid.c X
C cekCovid.c > cekCovid()
55 // system("cls");
56
57 printf("HASIL CEK GEJALA COVID\n");
58 printf("Persentase terindikasi covid adalah %.2f %c\n", sum, persen);
59
60 if (sum <= 10)
61 {
62     printf(">>Tidak terindikasi covid/Sehat.\n");
63 }
64
65 else if (sum <= 30)
66 {
67     printf(">>Terindikasi gejala ringan!\n");
68     printf(">>Harap melakukan isolasi dan perawatan mandiri di rumah!\n");
69 }
70
71 else if (sum <= 60)
72 {
73     printf(">>Terindikasi gejala sedang!\n");
74     printf(">>Harap segera melakukan pemeriksaan!\n");
75 }
76
77 else
78 {
79     printf(">>Terindikasi gejala serius!\n");
80     printf(">>Anda memiliki potensi terkena covid!\n>>Harap segera melakukan pemeriksaan!\n");
81 }
82
83 if (sum <= 30)
84 {
85     printf(">>Akses masuk diterima!\n");
86     printf("\nSilahkan lanjut ke tahap berikutnya.\n");
87 }
88 else
89 {
90     printf(">>Akses masuk ditolak!\n");
91 }
92
93 //getch();
94 if (sum <= 30)
95     return true;
96 else
97     return false;
98 }

```


3.1.3 cekCovid.h

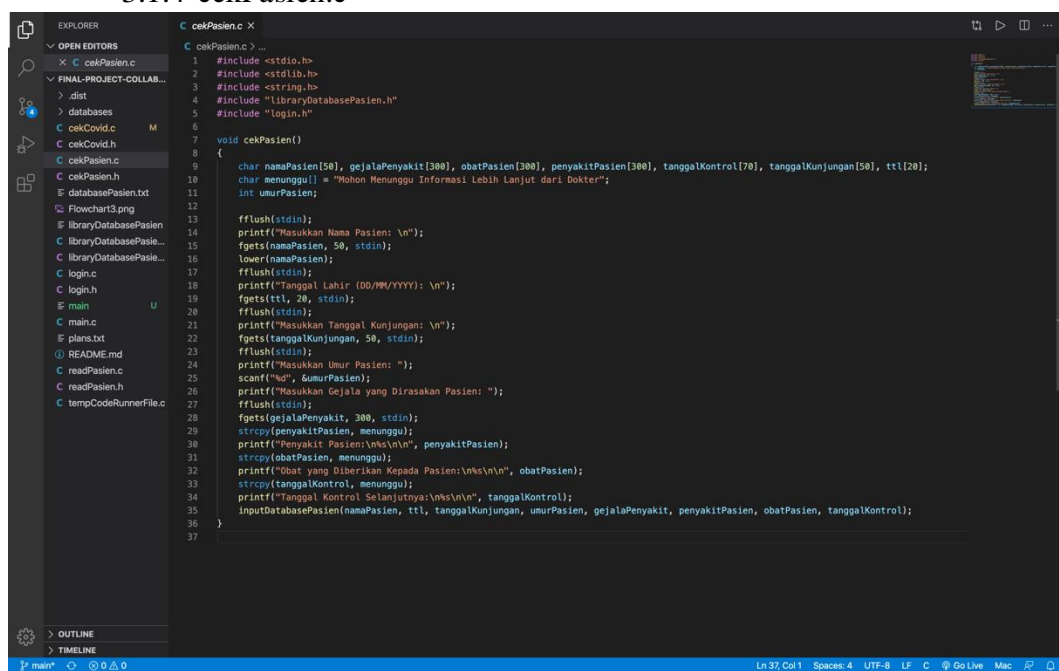


```

1 #ifndef CEKCOVID
2 #define CEKCOVID
3 #include "cekCovid.c"
4 #endif

```

3.1.4 cekPasien.c

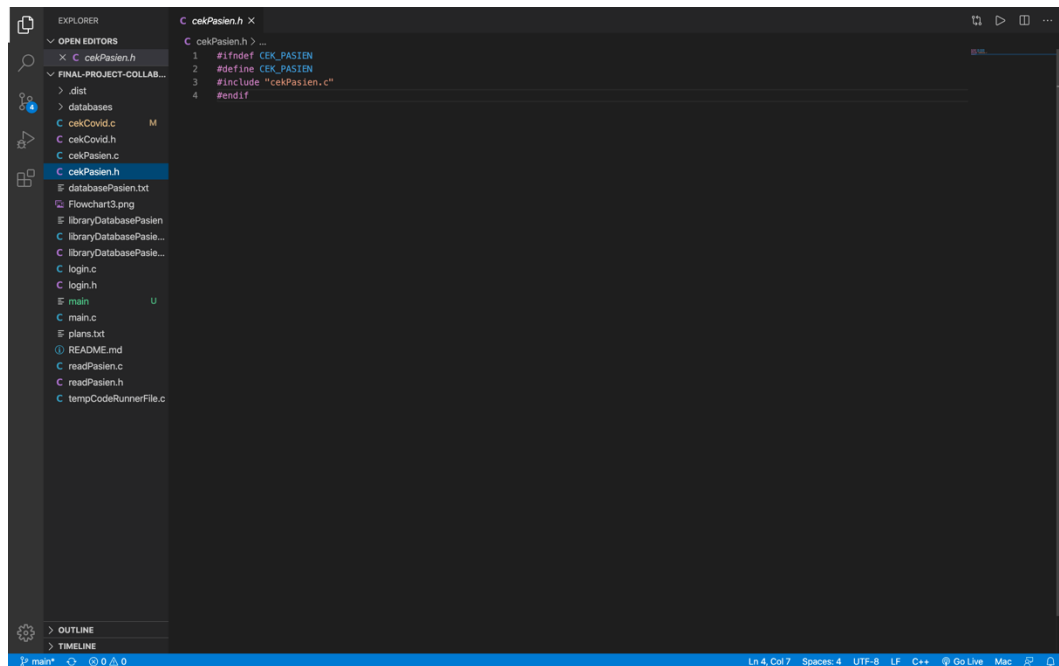


```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "libraryDatabasePasien.h"
5 #include "login.h"
6
7 void cekPasien()
8 {
9     char namaPasien[50], gejalaPenyakit[300], obatPasien[300], penyakitPasien[300], tanggalKontrol[70], tanggalKunjungan[50], ttl[20];
10     char menunggu[] = "Mohon Menunggu Informasi Lebih Lanjut dari Dokter";
11     int umurPasien;
12
13     fflush(stdin);
14     printf("Masukkan Nama Pasien: \n");
15     fgets(namaPasien, 50, stdin);
16     lower(namaPasien);
17     fflush(stdin);
18     printf("Tanggal Lahir (DD/MM/YYYY): \n");
19     fgets(ttl, 20, stdin);
20     fflush(stdin);
21     printf("Masukkan Tanggal Kunjungan: \n");
22     fgets(tanggalKunjungan, 50, stdin);
23     fflush(stdin);
24     printf("Masukkan Umur Pasien: ");
25     scanf("%d", &umurPasien);
26     printf("Masukkan Gejala yang Dirasakan Pasien: ");
27     fflush(stdin);
28     fgets(gejalaPenyakit, 300, stdin);
29     strcpy(penyakitPasien, menunggu);
30     printf("Penyakit Pasien:\n%s\n", penyakitPasien);
31     strcpy(obatPasien, menunggu);
32     printf("Obat yang Diberikan Kepada Pasien:\n%s\n", obatPasien);
33     strcpy(tanggalKontrol, menunggu);
34     printf("Tanggal Kontrol Selanjutnya:\n%s\n", tanggalKontrol);
35     inputDatabasePasien(namaPasien, ttl, tanggalKunjungan, umurPasien, gejalaPenyakit, penyakitPasien, obatPasien, tanggalKontrol);
36 }

```

3.1.5 cekPasien.h

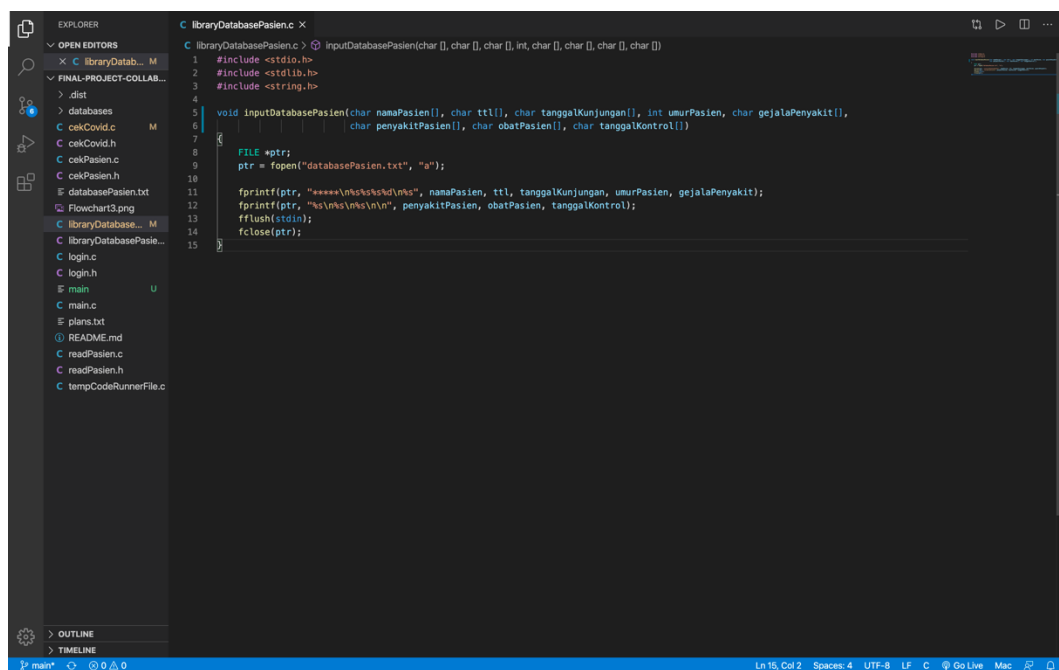


```

1 #ifndef CEK_PASIEN
2 #define CEK_PASIEN
3 #include "cekPasien.c"
4 #endif

```

3.1.6 libraryDatabasePasien.c

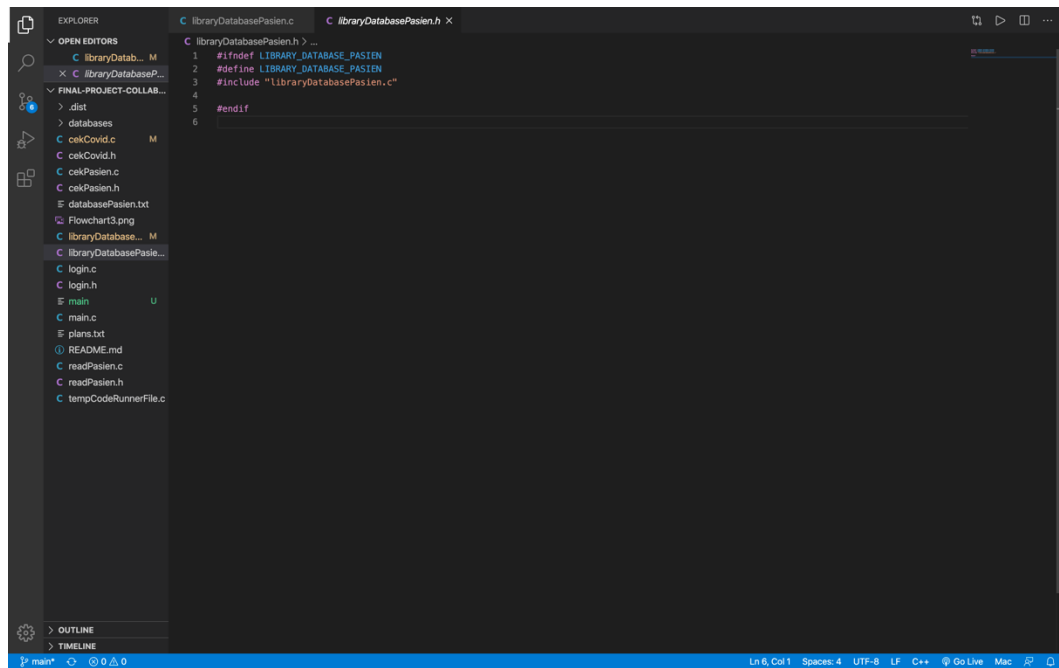


```

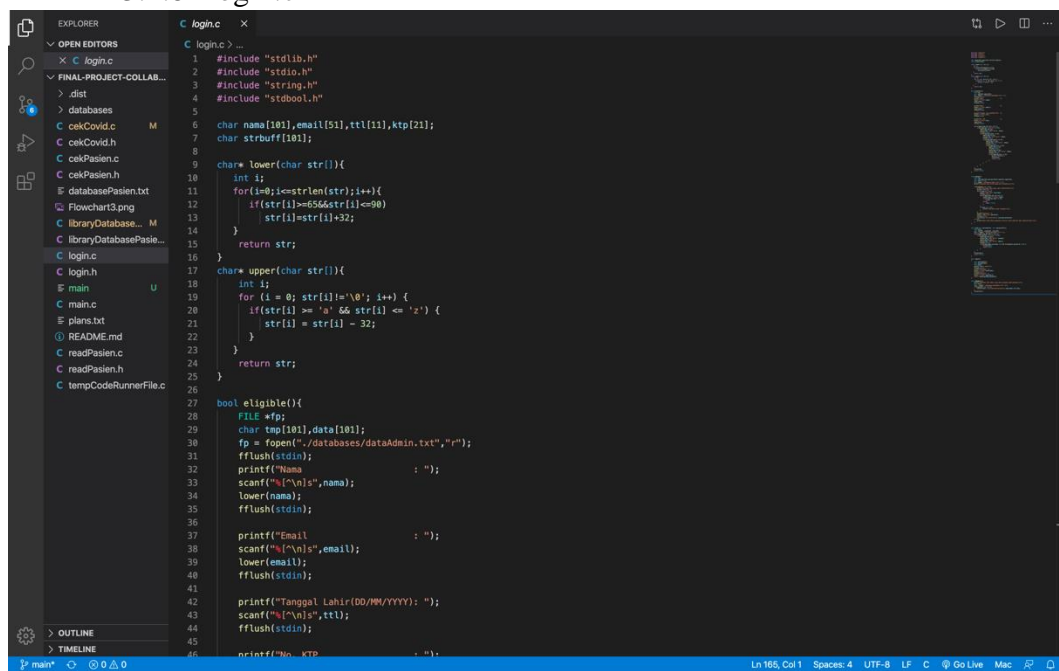
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 void inputDatabasePasien(char namaPasien[], char ttl[], char tanggalKunjungan[], int umurPasien, char gejalaPenyakit[],
6                           char penyakitPasien[], char obatPasien[], char tanggalKontrol[])
7
8 FILE *ptr;
9 ptr = fopen("databasePasien.txt", "a");
10
11 fprintf(ptr, "====\n%ks\n%ks\n%ks", namaPasien, ttl, tanggalKunjungan, umurPasien, gejalaPenyakit);
12 fprintf(ptr, "%s\n%ks\n%ks\n%ks", penyakitPasien, obatPasien, tanggalKontrol);
13 fflush(stdin);
14 fclose(ptr);
15

```

3.1.7 libraryDatabasePasien.h



3.1.8 Login.c



```

43  scanf("%i\n", &ttl);
44  fflush(stdin);
45
46  printf("No. KTP : ");
47  scanf("%i\n", &ktp);
48  lower(ktp);
49  fflush(stdin);
50
51  while(fgets(tmp, 101, fp) != NULL){
52      if(strcmp(tmp, "=====") == 0){
53          fgets(tmp, 101, fp);
54          sscanf(tmp, "%i\n", &tmp);
55          fflush(stdin);
56          if(strcmp(tmp, "=====") == 0){
57              fgets(tmp, 51, fp);
58              sscanf(tmp, "%i\n", &tmp);
59              fflush(stdin);
60              if(strcmp(tmp, email) == 0){
61                  fgets(tmp, 11, fp);
62                  sscanf(tmp, "%i\n", &tmp);
63                  fflush(stdin);
64                  if(strcmp(tmp, ttl) == 0){
65                      fgets(tmp, 21, fp);
66                      fflush(stdin);
67                      fgets(tmp, 21, fp);
68                      sscanf(tmp, "%i\n", &tmp);
69                      fflush(stdin);
70                      if(strcmp(tmp, ktp) == 0){
71                          printf("PASSSS\n");
72                          fclose(fp);
73                          return true;
74                      }
75                  }
76              }
77          }
78      }
79  }
80  fclose(fp);
81  return false;
82 }
83
84 void signup(){
85     FILE *fp;
86     char username[101], password[101], tmp[101], tmp2[101];
87
88     // ... (code continues)

```

```

88     // ... (code continues)
89     if(eligible() == true){
90         printf("Masukkan data yang ingin didaftarkan\n");
91         printf("Username: ");
92         while(loop == true){
93             scanf("%i\n", &username);
94             fflush(stdin);
95             while(fgets(tmp, 101, fp) != NULL){
96                 sscanf(tmp, "%i\n", &tmp2);
97                 if(strcmp(tmp, tmp2) == 0){
98                     break;
99                 }else{
100                     loop = false;
101                 }
102             }
103             if(loop == true){
104                 printf("Username telah terpakai\n");
105             }
106         }
107         printf("Password: ");
108         scanf("%i\n", &password);
109         fflush(stdin);
110         fprintf(fp, "=====\\n\\n", username, password);
111     } else{
112         printf("Maaf anda belum memenuhi kriteria untuk memiliki akun administrator\n");
113     }
114 }
115
116 bool read(char username[51], char password[51]){
117     FILE *fptr;
118     char tmp[60], uname[60], pass[60];
119     fptr = fopen("./databases/admin.txt", "r");
120     while(fgets(tmp, 60, fptr) != NULL){
121         if(strcmp(tmp, "=====\\n") == 0){
122             fgets(tmp, 60, fptr);

```

```

112     fflush(stdin);
113     fprintf(fp, "====\n%s\n", username, password);
114 } else{
115     printf("Maaf anda belum memenuhi kriteria untuk memiliki akun administrator!\n");
116 }
117 }
118
119 bool read(char username[51], char password[51]){
120     FILE *fptr;
121     char tmp[60], uname[60], pass[60];
122     fptr = fopen("./databases/admin.txt", "r");
123     while(fgets(tmp, 60, fptr) != NULL){
124         if(strcmp(tmp, "====\n") == 0){
125             fgets(tmp, 60, fptr);
126             sscanf(tmp, "%60[^\n]\n", &uname);
127             fgets(tmp, 60, fptr);
128             sscanf(tmp, "%60[^\n]\n", &pass);
129             fflush(stdin);
130             if(strcmp(uname, username) == 0 && strcmp(pass, password) == 0){
131                 fclose(fptr);
132                 return true;
133             }
134         }
135     }
136     fclose(fptr);
137     return false;
138 }
139
140 bool signin()
141 {
142     char username[51];
143     char password[51];
144     FILE *fptr;
145     printf("Admin Login\n");
146     fflush(stdin);
147     printf("Username: ");
148     scanf("%[^\n]s", &username);
149     fflush(stdin);
150     printf("Password: ");
151     scanf("%[^\n]s", &password);
152     return read(username, password);
153 }
154
155 void addAdmin(){
156     printf("Masukkan data admin yang akan ditambah pada database:\n");
157

```

```

148
149 bool signin()
150 {
151     char username[51];
152     char password[51];
153     FILE *fptr;
154     printf("Admin Login\n");
155     fflush(stdin);
156     printf("Username: ");
157     scanf("%[^\n]s", &username);
158     fflush(stdin);
159     printf("Password: ");
160     scanf("%[^\n]s", &password);
161     return read(username, password);
162 }
163
164 void addAdmin(){
165     printf("Masukkan data admin yang akan ditambah pada database:\n");
166     FILE *fptr;
167     fptr = fopen("./databases/dataAdmin.txt", "a");
168     if(!fptr){
169         printf("====\n%s\n", nama, email, ttl, ktp);
170     }
171     fclose(fptr);
172 }
173

```

3.1.9 Login.h

```

1 #ifndef LOGIN
2 #define LOGIN
3 #include "login.c"
4 #endif

```

3.1.10 readPasien.c

```

1 #include "stdio.h"
2 #include "stdlib.h"
3 #include "login.h"
4
5 char data1[700], data2[700], data3[700], data4[4], data5[700], data6[700], data7[700], data8[700];
6 char tmp[700];
7
8
9 bool lookupPasien(){
10     fflush(stdin);
11     char tmp[700];
12     printf("Masukkan Data Pasien yang ingin di lookup: \n");
13     printf("Nama          : ");
14     scanf("%s", &nama); //variable nama sudah ada pada login.c sebagai global variable
15     lower(nama);
16     fflush(stdin);
17
18     printf("Tanggal Lahir(DD/MM/YYYY): ");
19     scanf("%s", &tgl); //variable tgl sudah ada pada login.c sebagai global variable
20     fflush(stdin);
21
22     FILE *fp;
23     fp = fopen("databasePasien.txt", "r");
24     while(fgets(tmp, 700, fp) != NULL){
25         sscanf(tmp, "%s\n", data1);
26         if(strcmp(data1, nama) == 0){
27             fgets(tmp, 700, fp);
28             sscanf(tmp, "%s\n", data2);
29             if(strcmp(data2, tgl) == 0){
30
31                 fgets(tmp, 700, fp);
32                 sscanf(tmp, "%s\n", data3);
33                 fflush(stdin);
34
35                 fgets(tmp, 4, fp);
36                 sscanf(tmp, "%s\n", data4);
37                 fflush(stdin);
38
39                 fgets(tmp, 700, fp);
40                 sscanf(tmp, "%s\n", data5);
41                 fflush(stdin);
42
43                 fgets(tmp, 700, fp);
44
45
46

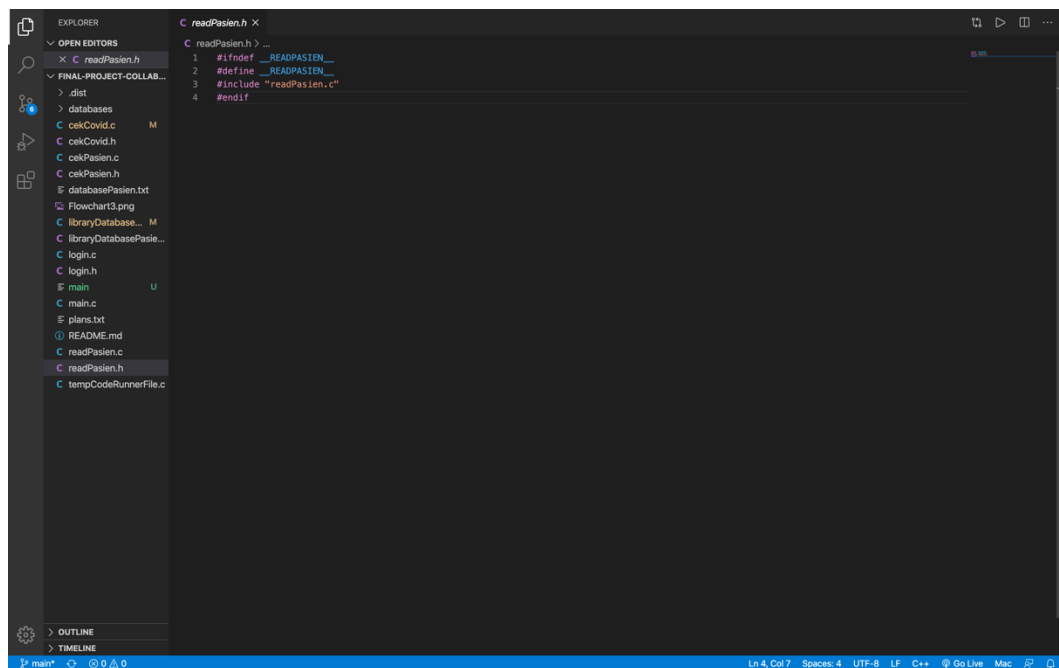
```

```
EXPLOLER      readPasienc.c >  updatePasienc()
OPEN EDITORS  readPasienc.c >  updatePasienc()
readPasienc.c 79      printf("Gejala          : %s\n", data5);
               80      printf("Penyakit           : %s\n", data6);
               81      printf("Obat              : %s\n", data7);
               82      printf("Tanggal Kontrol  : %s\n", data8);
               83      }
               84      }else printf("Maaf, data pasien yang anda cari tidak ditemukan!\n");
               85      }
               86      void updatePasienc()
               87      {
               88      int pil;
               89      char temp[500];
               90      if(lookupPasienc() == true){
               91      strcpy(temp, data1);
               92      upper(temp);
               93      printf("Nama          : %s\n", data1);
               94      strcpy(temp, data2);
               95      upper(temp);
               96      printf("Tanggal Lahir  : %s\n", data2);
               97      strcpy(temp, data3);
               98      upper(temp);
               99      printf("Tanggal Kunjungan : %s\n", data3);
               100      strcpy(temp, data4);
               101      upper(temp);
               102      printf("Jumur         : %s\n", data4);
               103      printf("Gejala          : %s\n", data5);
               104      printf("Penyakit           : %s\n", data6);
               105      printf("Obat              : %s\n", data7);
               106      printf("Tanggal Kontrol  : %s\n", data8);
               107      do{
               108      printf("Apakah anda ingin memperbaharui data ini?y/nl. Ya/n2. TidakkanPilihan: ");
               109      scanf("%c",&pil);
               110      switch(pil){
               111      case 1:
               112      {
               113      FILE *fp;
               114      FILE *fptr;
               115      fp = fopen("databasePasienc.txt","r");
               116      fptr = fopen("databases/tmp.txt","w");
               117      fflush(stdin);
               118      printf("Update data: \n");
               119      printf("Penyakit           : ");
               120      scanf("%[^\n]s",&data6);
               121      fflush(stdin);
               122      printf("Obat              : ");
               123      scanf("%[^\n]s",&data7);
               124      fflush(stdin);
               125      printf("Tanggal Kontrol  : ");
               126      scanf("%[^\n]s",&data8);
               127      fflush(stdin);
               128      printf("Tanggal Kunjungan : ");
               129      scanf("%[^\n]s",&data3);
               130      fflush(stdin);
               131      printf("Jumur         : ");
               132      scanf("%d",&data4);
               133      fflush(stdin);
               134      printf("Gejala          : ");
               135      scanf("%[^\n]s",&data5);
               136      fflush(stdin);
               137      printf("Nama          : ");
               138      scanf("%[^\n]s",&data1);
               139      fflush(stdin);
               140      printf("Tanggal Lahir  : ");
               141      scanf("%[^\n]s",&data2);
               142      fflush(stdin);
               143      printf("Tanggal Kunjungan : ");
               144      scanf("%[^\n]s",&data3);
               145      fflush(stdin);
               146      printf("Jumur         : ");
               147      scanf("%d",&data4);
               148      fflush(stdin);
               149      printf("Gejala          : ");
               150      scanf("%[^\n]s",&data5);
               151      fflush(stdin);
               152      printf("Penyakit           : ");
               153      scanf("%[^\n]s",&data6);
               154      fflush(stdin);
               155      printf("Obat              : ");
               156      scanf("%[^\n]s",&data7);
               157      fflush(stdin);
               158      printf("Tanggal Kontrol  : ");
               159      scanf("%[^\n]s",&data8);
               160      fflush(stdin);
               161      printf("Update data: \n");
               162      printf("Penyakit           : ");
               163      scanf("%[^\n]s",&data6);
               164      fflush(stdin);
               165      printf("Obat              : ");
               166      scanf("%[^\n]s",&data7);
               167      fflush(stdin);
               168      printf("Tanggal Kontrol  : ");
               169      scanf("%[^\n]s",&data8);
               170      fflush(stdin);
               171      printf("Tanggal Kunjungan : ");
               172      scanf("%[^\n]s",&data3);
               173      fflush(stdin);
               174      printf("Jumur         : ");
               175      scanf("%d",&data4);
               176      fflush(stdin);
               177      printf("Gejala          : ");
               178      scanf("%[^\n]s",&data5);
               179      fflush(stdin);
               180      printf("Nama          : ");
               181      scanf("%[^\n]s",&data1);
               182      fflush(stdin);
               183      printf("Tanggal Lahir  : ");
               184      scanf("%[^\n]s",&data2);
               185      fflush(stdin);
               186      printf("Tanggal Kunjungan : ");
               187      scanf("%[^\n]s",&data3);
               188      fflush(stdin);
               189      printf("Jumur         : ");
               190      scanf("%d",&data4);
               191      fflush(stdin);
               192      printf("Gejala          : ");
               193      scanf("%[^\n]s",&data5);
               194      fflush(stdin);
               195      printf("Penyakit           : ");
               196      scanf("%[^\n]s",&data6);
               197      fflush(stdin);
               198      printf("Obat              : ");
               199      scanf("%[^\n]s",&data7);
               200      fflush(stdin);
               201      printf("Tanggal Kontrol  : ");
               202      scanf("%[^\n]s",&data8);
               203      fflush(stdin);
               204      printf("Update data: \n");
               205      printf("Penyakit           : ");
               206      scanf("%[^\n]s",&data6);
               207      fflush(stdin);
               208      printf("Obat              : ");
               209      scanf("%[^\n]s",&data7);
               210      fflush(stdin);
               211      printf("Tanggal Kontrol  : ");
               212      scanf("%[^\n]s",&data8);
               213      fflush(stdin);
               214      printf("Tanggal Kunjungan : ");
               215      scanf("%[^\n]s",&data3);
               216      fflush(stdin);
               217      printf("Jumur         : ");
               218      scanf("%d",&data4);
               219      fflush(stdin);
               220      printf("Gejala          : ");
               221      scanf("%[^\n]s",&data5);
               222      fflush(stdin);
               223      printf("Nama          : ");
               224      scanf("%[^\n]s",&data1);
               225      fflush(stdin);
               226      printf("Tanggal Lahir  : ");
               227      scanf("%[^\n]s",&data2);
               228      fflush(stdin);
               229      printf("Tanggal Kunjungan : ");
               230      scanf("%[^\n]s",&data3);
               231      fflush(stdin);
               232      printf("Jumur         : ");
               233      scanf("%d",&data4);
               234      fflush(stdin);
               235      printf("Gejala          : ");
               236      scanf("%[^\n]s",&data5);
               237      fflush(stdin);
               238      printf("Penyakit           : ");
               239      scanf("%[^\n]s",&data6);
               240      fflush(stdin);
               241      printf("Obat              : ");
               242      scanf("%[^\n]s",&data7);
               243      fflush(stdin);
               244      printf("Tanggal Kontrol  : ");
               245      scanf("%[^\n]s",&data8);
               246      fflush(stdin);
               247      printf("Update data: \n");
               248      printf("Penyakit           : ");
               249      scanf("%[^\n]s",&data6);
               250      fflush(stdin);
               251      printf("Obat              : ");
               252      scanf("%[^\n]s",&data7);
               253      fflush(stdin);
               254      printf("Tanggal Kontrol  : ");
               255      scanf("%[^\n]s",&data8);
               256      fflush(stdin);
               257      printf("Tanggal Kunjungan : ");
               258      scanf("%[^\n]s",&data3);
               259      fflush(stdin);
               260      printf("Jumur         : ");
               261      scanf("%d",&data4);
               262      fflush(stdin);
               263      printf("Gejala          : ");
               264      scanf("%[^\n]s",&data5);
               265      fflush(stdin);
               266      printf("Nama          : ");
               267      scanf("%[^\n]s",&data1);
               268      fflush(stdin);
               269      printf("Tanggal Lahir  : ");
               270      scanf("%[^\n]s",&data2);
               271      fflush(stdin);
               272      printf("Tanggal Kunjungan : ");
               273      scanf("%[^\n]s",&data3);
               274      fflush(stdin);
               275      printf("Jumur         : ");
               276      scanf("%d",&data4);
               277      fflush(stdin);
               278      printf("Gejala          : ");
               279      scanf("%[^\n]s",&data5);
               280      fflush(stdin);
               281      printf("Penyakit           : ");
               282      scanf("%[^\n]s",&data6);
               283      fflush(stdin);
               284      printf("Obat              : ");
               285      scanf("%[^\n]s",&data7);
               286      fflush(stdin);
               287      printf("Tanggal Kontrol  : ");
               288      scanf("%[^\n]s",&data8);
               289      fflush(stdin);
               290      printf("Update data: \n");
               291      printf("Penyakit           : ");
               292      scanf("%[^\n]s",&data6);
               293      fflush(stdin);
               294      printf("Obat              : ");
               295      scanf("%[^\n]s",&data7);
               296      fflush(stdin);
               297      printf("Tanggal Kontrol  : ");
               298      scanf("%[^\n]s",&data8);
               299      fflush(stdin);
               300      printf("Tanggal Kunjungan : ");
               301      scanf("%[^\n]s",&data3);
               302      fflush(stdin);
               303      printf("Jumur         : ");
               304      scanf("%d",&data4);
               305      fflush(stdin);
               306      printf("Gejala          : ");
               307      scanf("%[^\n]s",&data5);
               308      fflush(stdin);
               309      printf("Nama          : ");
               310      scanf("%[^\n]s",&data1);
               311      fflush(stdin);
               312      printf("Tanggal Lahir  : ");
               313      scanf("%[^\n]s",&data2);
               314      fflush(stdin);
               315      printf("Tanggal Kunjungan : ");
               316      scanf("%[^\n]s",&data3);
               317      fflush(stdin);
               318      printf("Jumur         : ");
               319      scanf("%d",&data4);
               320      fflush(stdin);
               321      printf("Gejala          : ");
               322      scanf("%[^\n]s",&data5);
               323      fflush(stdin);
               324      printf("Penyakit           : ");
               325      scanf("%[^\n]s",&data6);
               326      fflush(stdin);
               327      printf("Obat              : ");
               328      scanf("%[^\n]s",&data7);
               329      fflush(stdin);
               330      printf("Tanggal Kontrol  : ");
               331      scanf("%[^\n]s",&data8);
               332      fflush(stdin);
               333      printf("Update data: \n");
               334      printf("Penyakit           : ");
               335      scanf("%[^\n]s",&data6);
               336      fflush(stdin);
               337      printf("Obat              : ");
               338      scanf("%[^\n]s",&data7);
               339      fflush(stdin);
               340      printf("Tanggal Kontrol  : ");
               341      scanf("%[^\n]s",&data8);
               342      fflush(stdin);
               343      printf("Tanggal Kunjungan : ");
               344      scanf("%[^\n]s",&data3);
               345      fflush(stdin);
               346      printf("Jumur         : ");
               347      scanf("%d",&data4);
               348      fflush(stdin);
               349      printf("Gejala          : ");
               350      scanf("%[^\n]s",&data5);
               351      fflush(stdin);
               352      printf("Nama          : ");
               353      scanf("%[^\n]s",&data1);
               35
```

The status bar at the bottom shows 'Ln 86, Col 21, Spaces: 4, UTF-8, LF, C, Go Live, Mac'."/>

The status bar at the bottom shows 'Ln 86, Col 21, Spaces: 4, UTF-8, LF, C, Go Live, Mac'."/>

3.1.11 readPasien.h



3.2 Penjelasan Singkat Kode Program

3.2.1 main()

Pada fungsi ini pertama-tama akan melakukan perulangan do while sehingga akan mengeksekusi terlebih dahulu perintah yang terdapat di dalam perulangan sebanyak 1 kali. Disajikan 4 buah menu, yaitu (1) Pengunjung, (2) Sign In Admin, (3) Sign Up Admin, serta (4) Exit. User akan diminta untuk memasukkan no pilihan menu. Bila user menginputkan no. 1, maka akan dieksekusi terlebih dahulu fungsi cekCovid, dan bila setelah diberikan akses, maka akan dilanjutkan dengan eksekusi fungsi cekPasien. Namun bila akses ditolak, maka user akan langsung dikeluarkan dari program dengan terlebih dahulu diberikan informasi bahwa akses ditolak. Bila user menginputkan no pilihan menu 2, maka user akan masuk ke bagian sign in admin dan mengeksekusi fungsi signin() terlebih dahulu. Bila berhasil sign in, user akan diberikan 3 pilihan, yaitu (1) Read Data Pasien, (2) Update Data Pasien, (3) Selesai. Bila memilih 1, maka

program akan mengeksekusi fungsi `printPasien()`, bila memilih 2, maka program akan mengeksekusi fungsi `updatePasien()`, dan bila memilih 3, maka program akan mengeksekusi fungsi `break`. Bila memilih 1 maupun 2, setelah fungsi dieksekusi, akan dijalankan kembali fungsi `pause()` dan menanyakan user apa ingin melanjutkan program atau tidak, seperti yang sudah dijelaskan sebelumnya. Selanjutnya kembali ke menu utama, bila user menginputkan angka 3, maka akan mengeksekusi fungsi `signup()`, dan bila user menginputkan angka 4, maka user akan diajak keluar dari perulangan dan keluar dari program. Selama user menginputkan nilai 1, 2, maupun 3, maka looping akan terus dijalankan.

3.2.2 cekCovid()

Mula-mula kita buat header yaitu `include stdio.h` untuk standar input output operasi, `include stdlib.h` untuk fungsi main, fungsi operator aritmatika pembandingan, selanjutnya `include conio.h` untuk menampilkan hasilnya, `include string.h` untuk membuat sebuah kalimat dalam array, dan `include stdbool.h` untuk memberikan nilai `return true` atau `false`.

Selanjutnya membuat fungsi bool `cekCovid`, bertipe bool agar dapat mereturn nilai `true` atau `false`. Di dalam fungsi ini perlu dideklarasikan variabel yang digunakan seperti integer `y`, `I`, `a`, `b`, `c` untuk bilangan bulat. Float `GejalaT1` sampai 3 dan `sum` untuk hasil akhir yaitu persentase terindikasi gejala yang bisa menghasilkan angka koma. Char `persen` untuk membuat karakter `%`. Char `sym1` sampai 3 untuk array yang didalamnya adalah sebuah string didefinisikan sebagai array 2 dimensi, 100 berarti maksimal banyak kata, dan 255 berarti maksimal banyak karakter dalam satu kata.

Dari gejala T1 sampai T3 memiliki perhitungan yang berbeda, karena tingkatannya semakin besar semakin serius. Maka dari itu mula-mula akan diberikan suatu perintah dimana masukkan 1 apabila pernah mengalami dan 0 apabila sebaliknya. Sebelum masuk

pada gejala T1 y kita inisialisasikan adalah 0. Kemudian kita tampilkan semua data pada gejala 1 yang ada di dalam array sebelumnya, dimana setiap data yang ditampilkan program akan meminta user untuk menginput nilai tersebut, apabila nilai 1 yang diinput maka percabangan if tersebut akan dijalankan dan statement akan dilakukan. Selanjutnya setelah semua data di Gejala T1 telah ditampilkan maka akan dilakukan proses perhitungan presentase pada gejala T1. Selanjutnya pada Gejala T2 sampai T3 dilakukan hal yang sama seperti Gejala T1 yaitu memulai dari menginisialisasikan nilai $y=0$ hingga menghitung presentase gejalanya. Setelah semua gejala sudah didapatkan, maka dilakukan lagi operasi aritmatika yaitu menjumlahkan ketiga gejalanya kemudian kita akan mendapatkan presentase akhir untuk cek covid.

Terakhir dilakukan penggolongan presentase terindikasi covid menggunakan percabangan if dengan operasi pembandingan. Maka presentase akhir tadi yaitu sum akan di cek satu satu dari kondisi awal sampai akhir. Apabila dalam percabangan tersebut kondisinya terpenuhi maka percabangan itu yang akan dikeluarkan statementnya. Dalam fungsi ini kita juga harus memberikan nilai true pada keluaran yang mendapat akses masuk yaitu pada gejala dibawah gejala ringan. Dengan memberikan nilai true ini maka program akan dilanjutkan ke tahap berikutnya yaitu menginput data kunjungan. Sebaliknya selain dari dibawah gejala serius atau apabila seseorang terindikasi gejala sedang atau serius maka program akan mereturn nilai false, kemudian program akan selesai dan kembali ke menu awal.

3.2.3 cekPasien()

Pada fungsi ini, terdapat variabel namaPasien, gejalaPenyakit, obatPasien, penyakitPasien, tanggalKontrol, tanggalKunjungan, serta ttl yang bertipe data char yang mampu menampung data berupa karakter dan berstruktur data array yang membuat masing-masing variabel memiliki batas maksimum karakter yang dapat ditampung.

Terdapat pula variabel data menunggu yang bertipe data char yang menampung array of char “Mohon Menunggu Informasi Lebih Lanjut dari Dokter”, serta variabel umurPasien yang bertipe data integer. Pada fungsi ini, user akan diminta untuk menginputkan data berupa nama pasien yang setelah user inputkan datanya, akan diproses dengan fungsi lower () yang didapat dari library login.h sehingga seluruh karakter dari nama yang diinputkan user akan berubah menjadi lowercase atau huruf kecil yang nantinya akan ditampung dalam variabel namaPasien. Lalu user juga akan diminta untuk menginputkan tanggal lahir dengan format (DD/MM/YYYY) yang ditampung dalam variabel ttl, memasukkan tanggal kunjungan yang ditampung dalam tanggalKunjungan, menginputkan umur pasien yang ditampung dalam variabel umurPasien, serta memasukkan gejala-gejala penyakit yang dirasakan pasien dan akan ditampung dalam variabel gejalaPenyakit. Setelah itu, keterangan mengenai Penyakit Pasien, Obat yang Diberikan kepada Pasien, serta Tanggal Kontrol Selanjutnya akan otomatis menghasilkan output berupa kalimat “Mohon Menunggu Informasi Lebih Lanjut dari Dokter” karena digunakan fungsi strcpy yang inputnya diambil dari variabel menunggu. Lalu, setelah semua data diinputkan, maka akan dicetak ke dalam databasePasien.txt karena digunakan fungsi inputDatabasePasien yang didapat dari library libraryDatabasePasien.h.

3.2.4 inputDatabasePasien()

Fungsi inputDatabasePasien() terdapat pada file libraryDatabasePasien.c. Pada fungsi ini, terdapat tipe data FILE dengan variabel pointer *ptr. Lalu variabel ptr akan membuka file dengan nama databasePasien.txt dengan mode Append, dimana dengan mode ini, maka akan menambahkan data-data baru dibawah data-data yang pernah diinputkan sebelumnya, dan apabila file dengan nama databasePasien.txt belum ada, maka mode ini akan secara otomatis membuat file dengan nama serupa. Lalu terdapat

fungsi `fprintf` yang akan mencetak data-data yang terdapat pada variabel `namaPasien`, `ttl`, `tanggalKunjungan`, `umurPasien`, `gejalaPenyakit`, `penyakitPasien`, `obatPasien`, serta `tanggalKontrol`. Lalu setelah mencetak data-data yang terdapat pada variabel tersebut, terdapat fungsi `fclose` yang berguna untuk menutup file yang dibuka oleh variabel `ptr` sebelumnya.

3.2.5 `Char* lower(char str[])`

Fungsi `char lower` ini berada pada file `login.c` yang berguna untuk mengecilkan seluruh karakter huruf apapun yang nantinya akan diinput. Untuk menggunakan `char lower` ini mula-mula kita harus membuat suatu variabel yang akan menampung sebuah string. Kemudian kita buat perulangan yang dimana akan mengambil semua karakter dari index 0 sampai akhir pada suatu kalimat atau kata dengan statement `i++` yang berarti karakter akan terus ditambahkan sampai akhir karakter. Kemudian diberikan perkondisian dimana apabila karakter yang diinput merupakan rentangan dari kode ASCII 65 sampai 90 yaitu huruf-huruf capital maka statement yang dijalankan untuk mengubahnya menjadi huruf kecil adalah dengan menambahkan sebanyak 32 pada indexnya sehingga akan didapatkan rentangan baru yaitu 97-122, dimana itu adalah kode ASCII untuk karakter a-z. Terakhir fungsi `lower` ini akan mereturn nilai variabel string yang akan diubah tersebut.

3.2.6 `Char* upper(char str[])`

Fungsi `char lower` ini berada pada file `login.c` yang berguna untuk mengkapitalkan seluruh karakter huruf apapun yang nantinya akan diinput. Tidak jauh berbeda dengan `char lower`. Perbedaannya adalah kita cukup mendeteksi bahwa apabila kita menginput suatu kata atau kalimat dimana berupa karakter yang lebih dari sama dengan a namun kurang dari sama dengan z dimana dalam kode ASCII rentangannya yaitu 97-122, untuk mengubahnya menjadi karakter dengan huruf capital maka cukup mengurangkan dengan 32 karena dalam ASCII untuk huruf capital rentangannya adalah 65-90.

Terakhir fungsi upper ini akan mereturn nilai variabel string yang akan diubah tersebut.

3.2.7 eligible()

Fungsi eligible terletak pada file “login.c”. Kegunaan dari fungsi eligible() ialah untuk mengecek apakah user yang dicek tersebut terdapat pada database “dataAdmin.txt”. Function memiliki tipe data boolean, jadi apabila data user memiliki kecocokan dengan data yang ada pada database “dataAdmin.txt” maka function tersebut akan mengembalikan nilai *true* dan apabila ada ketidakcocokan sedikitpun, maka function itu akan mengembalikan nilai *false*.

Cara kerja dari function ini yaitu, pertama tama, buka database “dataAdmin.txt” menggunakan fungsi fopen(). Data yang diambil dari user yaitu, nama, email, tanggal lahir, dan nomor ktp yang nantinya data data ini akan dicocokkan dengan data yang ada pada database “dataAdmin.txt”. Data dibuat menjadi huruf kecil terlebih dahulu sebelum dimasukkan ke variabel masing masing. Setelah mendapatkan data dari user, program memasukkan perulangan yang membaca tiap line pada database “dataAdmin.txt” menggunakan fgets() dan disimpan pada variabel “tmp”. Apabila data pada “tmp” sama dengan “*****\n” maka program akan memasukkan if statement. Lalu data dibaca lagi dengan fgets() dan disimpan pada “tmp”. Data tersebut diuraikan (*parse*) terlebih dahulu menggunakan fungsi sscanf untuk mendapatkan data “nama” dan simpan lagi pada tmp. Setelah itu data pada tmp dibandingkan dengan data “nama” yang diinputkan pada awal fungsi. Apabila memiliki kesamaan, maka akan memasukkan if statement selanjutnya. Proses yang terjadi pada bagian ini memiliki kesamaan dengan proses untuk mendapatkan nama dari database, hanya saja kali ini data yang diambil adalah “email” lalu dibandingkan dengan data email dari inputan user. Apabila memiliki kesamaan, program kembali mengambil data tanggal lahir lalu dibandingkan/ dicek. Apabila memiliki kesamaan dengan inputan user, maka program

kembali mengambil data sebagai data “ktp” lalu memasuki if statement yang terakhir pada if bersarang ini yaitu untuk mengecek apakah data nomor ktp sama dengan inputan user. Apabila memiliki kesamaan, maka file akan ditutup dengan `fclose()` lalu fungsi akan mengembalikan nilai *true*. Apabila data memiliki ketidakcocokan sedikitpun, maka proses pengecekan tersebut akan diulangi hingga akhir file (*NULL*). Apabila ada yang tidak cocok hingga line akhir dari database, maka akan menutup file dengan `fclose()` dan mengembalikan nilai *false*.

3.2.8 `signup()`

Fungsi `signup()` terdapat pada file `login.c`. program ini digunakan untuk melakukan proses registrasi user admin yang akan digunakan untuk melakukan login admin pada fungsi `signin()`. Sebelum melakukan registrasi user, terlebih dahulu dicek kelayakan user untuk mendaftar menggunakan fungsi `eligible()`. Jika fungsi `eligible()` bernilai *true* atau layak maka user akan diperbolehkan untuk melakukan registrasi dengan memasukkan Username dan Password yang ingin digunakan untuk `signin` admin.

3.2.9 `read()`

Fungsi `read()` terdapat pada file `login.c`. fungsi ini digunakan untuk mengecek kesesuaian data pada `admin.txt` dan data yang diinput oleh user pada fungsi `signin()` menggunakan perintah `strcmp`. Data yang dicek pada fungsi ini berupa Username dan Password, jika kedua data antara `admin.txt` dan input user pada fungsi `signin()` sama maka fungsi `read()` akan me-return nilai *true* dan jika tidak sama maka fungsi `read()` akan mereturn nilai *false()*

3.2.10 `signin()`

Fungsi `signin` terdapat pada file `login.c`. Pada fungsi ini, terdapat 2 variabel `char` yaitu `username` dan `password` dengan panjang array masing-masing 51. Akan dicetak kalimat “Admin Login” dan “Username: “ user akan diminta untuk memasukkan

username mereka, dan ada kalimat “Password: “ dan user juga akan diminta untuk menginputkan password mereka. Setelah menginputkan data berupa username dan password, maka akan dieksekusi fungsi read. Dan bila cocok, maka user akan diizinkan untuk login ke menu Login Admin.

3.2.11 addAdmin()

Pertama-tama akan dicetak kalimat “Masukkan data admin yang akan ditambah pada database:”. Lalu terdapat variabel pointer *fptr dengan tipe data FILE. Variabel fptr akan membuka folder database dan membuka file dataAdmin.txt dengan mode Append. Dengan mode ini, maka akan menambahkan data-data baru dibawah data-data yang pernah diinputkan sebelumnya. Akan dilakukan pemeriksaan terlebih dahulu dengan fungsi if yang mengeksekusi fungsi eligible(). Bila mereturn kondisi false, maka akan langsung dicetak data-data pada variabel nama, email, ttl, serta ktp pada file dataAdmin.txt

3.2.12 lookupPasien()

Fungsi ini terdapat pada file readPasien.c. fungsi ini digunakan untuk melakukan pengecekan pasien yang berkunjung pada databasePasien.txt sebelum melakukan atau mengeksekusi fungsi printPasien(). Data pasien yang dicek dalam fungsi ini berupa nama dan tanggal lahir (DD/MM/YYYY) pasien dan dicek menggunakan perintah strcmp, jika data sesuai maka fungsi lookupPasien() akan me-return nilai true, jika berbeda maka akan me-return nilai false.

3.2.13 printPasien()

Fungsi printPasien terdapat pada file readPasien.c. Pada fungsi ini, dicek terlebih dahulu perkondisian dengan fungsi lookupPasien(). Bila menghasilkan kondisi false, maka akan dicetak kalimat “Maaf, data pasien yang anda cari tidak ditemukan!”, namun bila bernilai true, maka akan mengeksekusi fungsi upper() terlebih dahulu untuk data yang dikandung dalam variabel data1, data2,

data3, serta data4. Lalu dengan fungsi printf akan dicetak data-data yang terdapat pada variabel data1, data2, data3, data4, data5, data6, data7, serta data8.

3.2.14 updatePasien()

Fungsi updatePasien() berada pada file “login.c” yang berguna untuk memperbaharui data dari pasien yang terdapat pada “databasePasien.txt”. Cara kerja dari fungsi ini yaitu, pertama tama fungsi lookupPasien() dipanggil. Apabila lookupPasien() bernilai true maka akan masuk ke dalam if statementnya. Hal yang pertama dilakukan adalah menampilkan data yang ada didatabase lalu menanyakan pada user apakah ia ingin memperbarui data tersebut. Apabila user memilih “Ya” maka akan memasuki switch case 1. Pada switch case 1, pertama tama file yang dibuka yaitu “databasePasien.txt” dalam mode read dan “tmp.txt” dalam mode write. Lalu user memasukkan data data yang diperbarui dan disimpan pada masing masing variabel. Lalu program memasuki while loop yang bertujuan untuk menyalin seluruh data dari “databasePasien.txt” kecuali data lama yang ingin diubah tadi ke “tmp.txt” dengan cara ditimpa (overwrite). Pada saat menyalin ke “tmp.txt”, apabila fgets() telah sampai pada line dari data yang ingin diperbarui tadi, data tersebut digantikan dengan melakukan fprintf() namun yang dicetak adalah data dari inputan user sebelumnya (data baru). Setelah selesai memasukkan data dari variabel, penyalinan dilanjutkan kembali hingga akhir file. Setelah itu kedua file ditutup dengann fclose(). Lalu file tadi dibuka lagi namun dengan mode yang berbeda. “databasePasien.txt” dibuka dengan mode write dan “tmp.txt” dibuka dengan mode read. Lalu memasukki perulangan while untuk menyalin seluruh data dari “tmp.txt” ke “databasePasien.txt” dengan cara ditimpa (overwrite) tanpa terkecuali. Setelah itu, kedua file ditutup dengan fclose() dan fungsi pun berakhir.

3.2.15 clear()

Fungsi ini terdapat pada main.c. Pada fungsi ini, berisikan dua macam perintah untuk membersihkan terminal. Bila user menggunakan sistem operasi windows, maka akan mengeksekusi perintah `system("cls")` sedangkan selain windows, maka akan mengeksekusi `system("clear")`.

3.2.16 pause()

Fungsi ini terdapat pada main.c. Pada fungsi ini, user akan ditanya apakah user ingin melanjutkan program. Bila iya, maka user akan diminta untuk menginputkan y atau Y, bila tidak, maka user cukup menginputkan t atau T dan otomatis akan mengeksekusi perintah `exit()` dan keluar dari program.

3.2.17 title()

Fungsi ini terdapat pada main.c. Fungsi ini hanya berfungsi untuk mencetak judul dari program ini, yaitu "Program Simulasi Rumah Sakit Spesialis"

3.3 Screenshot Run Program

```

PROGRAM SIMULASI RUMAH SAKIT SPESIALIS

KETERANGAN
>>Masukkan 1 Apabila pernah atau sedang mengalami gejala berikut
>>Masukkan 0 Apabila tidak pernah atau tidak sedang mengalami gejala berikut

GEJALA COVID
1. Suhu diatas 37 derajat = 1
2. Batuk kering = 1
3. Kelelahan = 1
4. Rasa tidak nyaman & nyeri = 1
5. Nyeri Tenggorokan = 1
6. Diare = 1
7. Mata merah = 1
8. Sakit kepala = 1
9. Hilang Indra perasa atau penciuman = 1
10. Perubahan warna pada jari = 1
11. Sesak nafas = 1
12. Nyeri dada = 1
13. Hilangnya kemampuan berbicara atau bergerak = 1

HASIL CEK GEJALA COVID
>>Persentase terindikasi covid adalah 98.70 %
>>Terindikasi gejala serius!
>>Anda memiliki potensi terkena covid!
>>Harap segera melakukan pemeriksaan!
>>Akses masuk ditolak!

PROGRAM SIMULASI RUMAH SAKIT SPESIALIS

KETERANGAN
>>Masukkan 1 Apabila pernah atau sedang mengalami gejala berikut
>>Masukkan 0 Apabila tidak pernah atau tidak sedang mengalami gejala berikut

GEJALA COVID
1. Suhu diatas 37 derajat = 0
2. Batuk kering = 0
3. Kelelahan = 0
4. Rasa tidak nyaman & nyeri = 0
5. Nyeri Tenggorokan = 0
6. Diare = 0
7. Mata merah = 0
8. Sakit kepala = 0
9. Hilang Indra perasa atau penciuman = 0
10. Perubahan warna pada jari = 0
11. Sesak nafas = 0
12. Nyeri dada = 0
13. Hilangnya kemampuan berbicara atau bergerak = 0

HASIL CEK GEJALA COVID
>>Persentase terindikasi covid adalah 0.00 %
>>Tidak terindikasi covid/Sehat.
>>Akses masuk diterima!

Silahkan lanjut ke tahap berikutnya.
Masukkan Nama Pasien:
Dirga
Tanggal Lahir (DD/MM/YYYY):
09/12/2001
Masukkan Tanggal Kunjungan:
25 Mei 2021
Masukkan Umur Pasien: 19
Masukkan Gejala yang Dirasakan Pasien: Sakit di perut bagian Kiri
Penyakit Pasien:
Mohon Menunggu Informasi Lebih Lanjut dari Dokter

Obat yang Diberikan Kepada Pasien:
Mohon Menunggu Informasi Lebih Lanjut dari Dokter

Tanggal Kontrol Selanjutnya:
Mohon Menunggu Informasi Lebih Lanjut dari Dokter

Apakah anda ingin melanjutkan program?(y/t)
Pilihan:

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+

Admin Login
Username: kuroyamii
Password: helloworld

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+

1. Tambah Admin
2. Menu Pasien
3. Log Out
Pilihan:

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+

Masukkan data admin yang akan ditambah pada database:
Nama          : evi
Email         : evi@gmail.com
Tanggal Lahir(DD/MM/YYYY): 12/12/2001
No. KTP       : 1234567890123456
Apakah anda ingin melanjutkan program?(y/t)
Pilihan:

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
1. Read Data Pasien
2. Update Data Pasien
3. Selesai
Pilihan: █

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
Masukkan Data Pasien yang ingin di lookup:
Nama                : dirga
Tanggal Lahir(DD/MM/YYYY): 09/12/2001
Nama                : DIRGA
Tanggal Lahir       : 09/12/2001
Tanggal Kunjungan   : 25 MEI 2021
Umur                : 19
Gejala              : Sakit di perut bagian kiri
Penyakit             : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Obat                 : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Tanggal Kontrol     : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Apakah anda ingin melanjutkan program?(y/t)
Pilihan: █

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
Masukkan Data Pasien yang ingin di lookup:
Nama                : dirga
Tanggal Lahir(DD/MM/YYYY): 09/12/2001
Nama                : dirga
Tanggal Lahir       : 09/12/2001
Tanggal Kunjungan   : 25 Mei 2021
Umur                : 19
Gejala              : Sakit di perut bagian kiri
Penyakit             : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Obat                 : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Tanggal Kontrol     : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Apakah anda ingin memperbaharui data ini?
1. Ya
2. Tidak
Pilihan: █

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
Masukkan Data Pasien yang ingin di lookup:
Nama                : dirga
Tanggal Lahir(DD/MM/YYYY): 09/12/2001
Nama                : dirga
Tanggal Lahir       : 09/12/2001
Tanggal Kunjungan   : 25 Mei 2021
Umur                : 19
Gejala              : Sakit di perut bagian kiri
Penyakit             : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Obat                 : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Tanggal Kontrol     : Mohon Menunggu Informasi Lebih Lanjut dari Dokter
Apakah anda ingin memperbaharui data ini?
1. Ya
2. Tidak
Pilihan: 1
Update data:
Penyakit            : Usus Buntu
Obat                 : metronidazole dan cefotaxime
Tanggal Kontrol     : 30 Mei 2021
Apakah anda ingin melanjutkan program?(y/t)
Pilihan: █

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
Masukkan data untuk memeriksa eligibility:
Nama                : evi
Email               : evi@gmail.com
Tanggal Lahir(DD/MM/YYYY): 12/12/2001
No. KTP             : 1234567890123456
Maaf anda belum memenuhi kriteria untuk memiliki akun administrator!
Apakah anda ingin melanjutkan program?(y/t)
Pilihan: █

```

```

+-----+
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |
+-----+
Masukkan data untuk memeriksa eligibility:
Nama                : evi
Email               : evi@gmail.com
Tanggal Lahir(DD/MM/YYYY): 12/12/2001
No. KTP             : 1234567890123456
PASS5
Masukkan data yang ingin didaftarkan:
Username: evi
Username telah terpakai!
█

```

```
+-----+  
|          PROGRAM SIMULASI RUMAH SAKIT SPESIALIS          |  
+-----+  
Admin Login  
Username: evi  
Password: evi  
Access Denied!  
Apakah anda ingin melanjutkan program?(y/t)  
Pilihan: █
```

BAB IV

PENUTUP

4.1 Kesimpulan

Kesimpulan yang dapat diambil adalah, program ini merupakan sebuah program dengan nama Simulasi Rumah Sakit Spesialis, seperti namanya program ini digunakan untuk mensimulasikan sebuah rumah sakit pada masa pandemic covid-19. menu yang terdapat pada program ini berupa menu: pengunjung, sign in admin, dan sign up admin. Pada menu pengunjung digunakan untuk pengujung yang ingin berobat dan sebelum melanjutkan untuk melakukan konsultasi penyakit yang diderita, pengunjung terlebih dahulu akan dicek gejala-gejala covid-19. Selanjutnya pada menu sign in admin terdapat sebuah menu untuk melakukan penambahan admin dan update data dari pasien yang berobat. Dan pada menu sign up admin digunakan untuk melakukan registrasi user admin baru.

4.2 Saran

Terkait dengan hal tersebut disarankan untuk penggunaan Bahasa C sebaiknya dipelajari semua orang karena bukan hanya mengasah logika, namun juga mengasah cara berpikir secara sistematis

DAFTAR PUSTAKA

- Aditya, Reza. “Teknik Pemrograman Terstruktur 1”. [Online] <http://rezaaditya.staff.gunadarma.ac.id/Downloads/files/42537/Struktur+Dasar+Bahasa+C+%28Materi+6+%29.pdf>. (Diakses 23 Februari 2021).
- Anonim. (2019). “Character functions in C”. [Online] <https://ladderpython.com/lesson/character-functions-in-c/>. (Diakses 3 April 2021).
- Anonim. (2020).” C Programming/String manipulation”. [Online] https://en.wikibooks.org/wiki/C_Programming/String_manipulation. (Diakses 3 April 2021).
- Anonim. (2021).” Computer Programming - File I/O”. [Online] https://www.tutorialspoint.com/computer_programming/computer_programming_file_io.htm. (Diakses 24 April 2021).
- Anonim. “Files in C”. [Online] <http://web.eng.fiu.edu/watsonh/EEL2880/FileIOHW.pdf>. (Diakses 24 April 2021).
- Anonim. “Pointer”. [Online] [http://yuliana.lecturer.pens.ac.id/Konsep%20Pemrograman/Teori/T12Pointer\(1\).pdf](http://yuliana.lecturer.pens.ac.id/Konsep%20Pemrograman/Teori/T12Pointer(1).pdf). (Diakses 2 April 2021).
- Anonim. “String functions in C with examples” [Online] <https://ashikur.buet.ac.bd/CSE115/StringDoc.pdf>. (Diakses 3 April 2021).
- Halimsetiawan, Hermanto, Jeffrey.(2008). “String dalam Bahasa C” [Online] <https://tutorialpemrograman.wordpress.com/2008/02/12/string-dalambahasa-c/>. (Diakses 3 April 2021).
- Muhardian, Ahmad. (2019). “Belajar Pemrograman C #11: Apa itu Pointer?”. [Online] <https://www.petanikode.com/c-pointer/>. (Diakses 2 April 2021).
- Muhardian, Ahmad. (2019). “Belajar Pemrograman C #5: Mengenal Variabel, Tipe Data, Konstanta”. [Online] <https://www.petanikode.com/c-variabel/>. (Diakses 12 Februari 2021).
- Muhardian, Ahmad. (2019). “Belajar Pemrograman C #9: Mengenal Struktur Data Array pada C”. [Online] <https://www.petanikode.com/c-array/>. (Diakses 14 Maret 2021).
- Muhardian, Ahmad. (2019).” Belajar Pemrograman C #15: Cara Membaca dan Menulis File di C”. [Online] <https://www.petanikode.com/c-file/>. (Diakses 24 April 2021).

- Muliantara, S.Kom, M.Kom, Agus. (2020). “Algoritma dan Pemograman File”. [Online]
http://sisil.dosen.ittelkompwt.ac.id/wpcontent/uploads/sites/3/2018/01/14_Pemrosesan-Arsip.pdf. (Diakses 23 April 2021).
- Pasuatmadi, Putu Audi, dan Rafif Jhordi. (2021). “Modul 3 Array/Larik”. [Online]
https://drive.google.com/open?id=1uTYjvAz7_rt868QgsPmsnDNW4lyuzFN_Ay9OWw19AE&authuser=1 (Diakses 14 Maret 2021).
- Purnomo, Eko. (2016). “Cara Melewatkan Parameter ke dalam Fungsi pada Bahasa C”. [Online] <https://www.nulis-ilmu.com/cara-melewatkanparameter-ke-dalam-fungsi-pada-bahasa-c/> (Diakses 4 April 2021).
- S.Kom., M.Kom, Fathushahib & Saintika S.T., M.T.I, Yudha. (2018). “Pengulangan Algoritma Dan Pemrograman [IS6110102]”. [Online]
https://yudha.dosen.ittelkom-pwt.ac.id/wp-content/uploads/sites/73/2018/10/Pertemuan-VI_Pengulangan.pdf. (Diakses 26 Februari 2021).
- S.Kom., M.Kom, Fathushahib & Saintika S.T., M.T.I, Yudha. (2018). “Larik/Array Algoritma Dan Pemrograman [IS6110102]”. [Online]
https://yudha.dosen.ittelkom-pwt.ac.id/wpcontent/uploads/sites/73/2018/10/Pertemuan-VIII_Array.pdf (Diakses 14 Maret 2021).
- S.Kom., M.Kom, Fathushahib & Saintika S.T., M.T.I, Yudha. (2019). “Algoritma Pemrograman (Semester 1 - IF6110202) Pertemuan 5 – Pemilihan/Percabangan”. [Online]
<https://dokumen.tech/document/algoritma-pemrograman-semester-1if6110202-pertemuan-yudhadosenittelkom-pwtacidpertemuanvpercabanganifsipdf.html>. (Diakses 25 Februari 2021).
- Suryaningrum. “BAB 7 File”. [Online]
<http://suryaningrum.staff.gunadarma.ac.id/Downloads/files/33516/m7.pdf>. (Diakses 23 April 2021).