

**LAPORAN TUGAS BESAR  
IF1210 ALGORITMA DAN PEMROGRAMAN 1  
SEMESTER 2 2024/2025**



Kelompok K

Anggota :

1. Faiq Azzam Nafidz (13524003)
2. Philipp Hamara (13524101)
3. Nicholas Luis Chandra (13524105)
4. Ega Luthfi Rais (13524115)
5. Varistha Devi (13524135)

**SEKOLAH TEKNIK ELEKTRO INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2025**

# Pernyataan Kelompok

Tabel 0.1 Pernyataan Kelompok

Nama	Pernyataan
Faiq Azzam Nafidz (13524003)	Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025
Philipp Hamara (13524101)	Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025
Nicholas Luis Chandra (13524105)	Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025
Ega Luthfi Rais (13524115)	Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025
Varistha Devi (13524135)	Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025

# Daftar Isi

Pernyataan Kelompok.....	i
Daftar Isi.....	ii
Daftar Gambar.....	v
Daftar Tabel.....	viii
Deskripsi Persoalan.....	1
Rencana Implementasi.....	3
Pembagian Kerja.....	6
Testing Primitif.....	8
Desain Command.....	9
Desain Kamus Data.....	15
1. ADT User (List).....	15
2. ADT Obat (List).....	18
3. ADT Penyakit (List).....	19
4. ADT Obat-Penyakit (Map).....	21
5. ADT Config.....	22
6. ADT Denah (Matriks).....	23
7. ADT Matriks.....	23
8. ADT Linked-List.....	24
9. ADT Stack-Linked-List.....	25
10. ADT Queue-Linked-List.....	25
Desain Dekomposisi.....	27
1. F01 - Login.....	27
2. F02 - Register Pasien.....	28
3. F03 - Logout.....	28
4. F04 - Lupa Password.....	29
5. F05 - Menu & Help.....	30
6. F06 - Denah Rumah Sakit.....	31
7. F07 - Lihat User.....	32
8. F08 - Cari User.....	32
9. F09 - Lihat Antrian.....	33
10. F10 - Tambah Dokter.....	34
11. F11 - Diagnosis.....	36
12. F12 - Ngobatin.....	37
13. F13 - Aku boleh pulang ga, dok 😊 ?.....	38
14. F14 - Daftar Check-Up.....	39
15. F15 - Antrian Saya!.....	41
16. F16 - Minum Obat + B05 - Dead or Alive.....	42
17. F17 - Minum Penawar.....	43
18. F18 - Exit.....	43
19. B02 - Denah Dinamis.....	44
20. B06 - Mainin Antrian.....	45
Spesifikasi Tiap Modul/Prosedur/Fungsi.....	46

1. F01 - Login.....	46
2. F02 - Register.....	47
3. F03 - Logout.....	48
4. F04 - Lupa Password.....	48
5. F05 - Help.....	50
6. F06 - Denah Rumah Sakit (Lihat Denah dan Lihat Ruangan).....	52
7. F07 - Lihat User.....	54
8. F08 - Cari User.....	56
9. F09 - Lihat Antrian.....	58
10. F10 - Tambah Dokter (Tambah Dokter dan Assign Dokter).....	60
11. F11 - Diagnosis.....	62
12. F12 - Ngobatin.....	64
13. F13 - Aku boleh pulang ga, dok 😊 ?.....	66
14. F14 - Daftar Checkup.....	69
15. F15 - Antrian Saya.....	72
16. F16 - Minum Obat + B05 - Dead or Alive.....	73
17. F17 - Minum Penawar.....	76
18. F18 - Exit.....	77
19. B02 - Denah dinamis.....	78
20. B06 - Mainin Antrian (Skip Antrian dan Cancel Antrian).....	79
Dokumentasi.....	84
F01 - Login.....	84
F02 - Register Pasien.....	86
F03 - Logout.....	87
F04 - Lupa Password.....	88
F05 - Menu & Help.....	91
F06 - Lihat Denah.....	93
F07 - Lihat User.....	94
F08 - Cari User.....	95
F09 - Lihat Antrian.....	96
F10 - Tambah Dokter & Assign Dokter.....	97
F11 - Diagnosis.....	99
F12 - Ngobatin.....	100
F13 - Aku boleh pulang ga, dok?.....	101
F14 - Daftar Check-Up.....	102
F15 - Antrian Saya!.....	103
F16 - Minum Obat + B05 - Dead or Alive.....	104
F17 - Minum Penawar.....	104
F18 - Exit.....	105
D03 - Load.....	105
D04 - Save.....	106
B02 - Denah Dinamis.....	107
B03 - Aura!.....	107
B05 - Dead or Alive.....	108

B06 - Mainin Antrian.....	108
Lampiran.....	109

# Daftar Gambar

Gambar 7.1 Flowchart F01 - Login.....	27
Gambar 7.2 Flowchart F02 - Register Pasien.....	28
Gambar 7.3 Flowchart F03 - Logout.....	28
Gambar 7.4 Flowchart F04 - Lupa Password.....	29
Gambar 7.5 Flowchart F05 - Menu & Help.....	30
Gambar 7.6 Flowchart F06 - Denah Rumah Sakit.....	31
Gambar 7.7 Flowchart F07 - Lihat User, Lihat Pasien, Lihat Dokter.....	32
Gambar 7.8 Flowchart F08 - Cari User, Cari Dokter, Cari Pasien.....	32
Gambar 7.9 Flowchart F09 - Lihat Antrian.....	33
Gambar 7.10a Flowchart F10 - Tambah Dokter.....	34
Gambar 7.10b Flowchart F10 - Assign Dokter.....	35
Gambar 7.11 Flowchart F11 - Diagnosis.....	36
Gambar 7.12 Flowchart F12 - Ngobatin.....	37
Gambar 7.13 Flowchart F13 - Aku Boleh Pulang Ga, Dok 😊 ?.....	38
Gambar 7.14 Flowchart F14 - Daftar Check-Up.....	41
Gambar 7.15 Flowchart F15 - Antrian Saya!.....	41
Gambar 7.16 Flowchart F16 - Minum Obat + B05 - Dead or Alive.....	42
Gambar 7.17 Flowchart F17 - Minum Penawar.....	43
Gambar 7.18 Flowchart F18 - Exit.....	43
Gambar 7.19a Flowchart B02 - Pindah dokter.....	44
Gambar 7.19b Flowchart B02 - Ubah Denah.....	44
Gambar 7.20a Flowchart B06 - Skip Antrian.....	45
Gambar 7.20b Flowchart B06 - Cancel Antrian.....	45
Gambar 9.1a Dokumentasi F01 - Login - Login Sukses.....	84
Gambar 9.1b Dokumentasi F01 - Login - Username Salah.....	85
Gambar 9.1c Dokumentasi F01 - Login - Password Salah.....	85
Gambar 9.1d Dokumentasi F01 - Login - Login Berbagai Peran.....	86
Gambar 9.2a Dokumentasi F02 - Register Pasien - Registrasi Sukses.....	86
Gambar 9.2b Dokumentasi F02 - Register Pasien - Username Sudah Ada.....	87
Gambar 9.3a Dokumentasi F03 - Logout - Logout Sukses.....	87
Gambar 9.3b Dokumentasi F03 - Logout - Akses Setelah Logout.....	88
Gambar 9.4a Dokumentasi F04 - Lupa Password - Username Terdaftar.....	88
Gambar 9.4b Dokumentasi F04 - Lupa Password - Username Tidak Terdaftar.....	89
Gambar 9.4c Dokumentasi F04 - Lupa Password - Kode Verifikasi Benar.....	89
Gambar 9.4d Dokumentasi F04 - Lupa Password - Kode Verifikasi Salah.....	90
Gambar 9.4e Dokumentasi F04 - Lupa Password - Buat Password Baru Valid.....	90
Gambar 9.4f Dokumentasi F04 - Lupa Password - Validasi Input.....	91
Gambar 9.5a Dokumentasi F05 - Menu & Help - Manager.....	91
Gambar 9.5b Dokumentasi F05 - Menu & Help - Menu Pengguna Belum Login.....	92
Gambar 9.5c Dokumentasi F05 - Menu & Help (Sesuai Peran).....	93
Gambar 9.6a Dokumentasi F06 - Lihat Denah - Tampil Denah.....	93
Gambar 9.6b Dokumentasi F06 - Lihat Denah - Lihat Ruangan Valid.....	94

Gambar 9.6c Dokumentasi F06 - Lihat Denah - Lihat Ruangan Tidak Valid.....	94
Gambar 9.7a Dokumentasi F07 - Lihat User - Tampil Semua User.....	95
Gambar 9.7b Dokumentasi F07 - Lihat User - Tampil User Sort.....	95
Gambar 9.8a Dokumentasi F08 - Cari User - Cari User ID Valid.....	96
Gambar 9.8b Dokumentasi F08 - Cari User - Cari User Nama Valid.....	96
Gambar 9.9a Dokumentasi F09 - Lihat Antrian - Antrian Ruangan Dokter.....	97
Gambar 9.10a Dokumentasi F10 - Tambah Dokter & Assign Dokter - Tambah Dokter Valid	97
Gambar 9.10b Dokumentasi F10 - Tambah Dokter & Assign Dokter - Tambah Dokter Username Ada.....	98
Gambar 9.10c Dokumentasi F10 - Tambah Dokter & Assign Dokter - Assign Dokter Valid... 99	99
Gambar 9.10d Dokumentasi F10 - Tambah Dokter & Assign Dokter - Assign Dokter Tidak Valid.....	99
Gambar 9.11a Dokumentasi F11 - Diagnosis - Diagnosis Valid.....	100
Gambar 9.12a Dokumentasi F12 - Ngobatin - Resep Berdasarkan Diagnosa.....	101
Gambar 9.12b Dokumentasi F12 - Ngobatin - Resep Tanpa Diagnosa.....	101
Gambar 9.13a Dokumentasi F13 - Pulangdok - Konsultasi Sebelum Pulang.....	102
Gambar 9.14a Dokumentasi F14 - Daftar Check-Up - Daftar Check-Up Valid.....	103
Gambar 9.14b Dokumentasi F14 - Daftar Check-Up - Daftar Check-Up Tidak Valid/Lengkap.. 103	103
Gambar 9.15a Dokumentasi F15 - Antrian Saya - Status Antrian Pasien.....	104
Gambar 9.16a Dokumentasi F16 - Minum Obat - Daftar Obat Pasien.....	104
Gambar 9.16b Dokumentasi F16 - Minum Obat - Pilih Obat.....	104
Gambar 9.16c Dokumentasi F16 - Minum Obat - Minum Obat Valid.....	104
Gambar 9.16d Dokumentasi F16 - Minum Obat - Cek Kondisi Kesehatan.....	104
Gambar 9.16e Dokumentasi F16 - Minum Obat - Hasil Kondisi Kesehatan Berbeda.....	104
Gambar 9.17a Dokumentasi F17 - Minum Penawar - Minum Penawar Salah Obat.....	105
Gambar 9.17b Dokumentasi F17 - Minum Penawar - Minum Penawar Tidak Salah Obat..	105
Gambar 9.17c Dokumentasi F17 - Minum Penawar - Efek Penawar.....	105
Gambar 9.18a Dokumentasi F18 - Exit - Simpan Data Sebelum Keluar.....	105
Gambar 9.18b Dokumentasi F18 - Exit - Keluar Tanpa Simpan.....	105
Gambar 9.18c Dokumentasi F18 - Exit - Proses Simpan Sukses.....	105
Gambar 9.19a Dokumentasi D03 - Load - Memuat Data Valid.....	106
Gambar 9.19b Dokumentasi D03 - Load - Memuat Data Tidak Valid.....	106
Gambar 9.20a Dokumentasi D04 - Save - Menyimpan Data Valid.....	107
Gambar 9.20b Dokumentasi D04 - Save - Menyimpan Data Tidak Valid.....	107
Gambar 9.20c Dokumentasi D04 - Save - Izin Folder Terbatas.....	107
Gambar 9.21a Dokumentasi B02 - Denah Dinamis - Ubah Ukuran Valid.....	107
Gambar 9.21b Dokumentasi B02 - Denah Dinamis - Ubah Ukuran Tidak Valid.....	107
Gambar 9.21c Dokumentasi B02 - Denah Dinamis - Pindah Dokter Valid.....	107
Gambar 9.21d Dokumentasi B02 - Denah Dinamis - Pindah Dokter Tidak Valid.....	107
Gambar 9.22a Dokumentasi B03 - Aura! - Urutkan Data.....	108
Gambar 9.22b Dokumentasi B03 - Aura! - Urutkan Data Kosong.....	108
Gambar 9.23a Dokumentasi B05 - Dead or Alive - Nyawa Lebih Dari 0.....	108
Gambar 9.23b Dokumentasi B05 - Dead or Alive - Nyawa 0.....	108
Gambar 9.24a Dokumentasi B06 - Mainin Antrian - Panggil Pasien Valid.....	108

Gambar 9.24b Dokumentasi B06 - Mainin Antrian - Panggil Pasien Kosong.....	108
Gambar 10.1a Hasil Pindai Form Asistensi 1 (Halaman Pertama).....	109
Gambar 10.1b Hasil Pindai Form Asistensi 1 (Halaman Kedua).....	110
Gambar 10.2 Hasil Pindai Form Asistensi 2.....	111

# **Daftar Tabel**

Tabel 0.1 Pernyataan Kelompok.....	1
Tabel 2.1 Rencana Implementasi.....	2
Tabel 3.1 Pembagian Kerja.....	5
Tabel 4.1 Testing primitif.....	7

# Deskripsi Persoalan

Tugas besar mata kuliah IF1210 Algoritma dan Pemrograman 1 menggunakan bahasa C sebagai bahasa utama dalam implementasi, serta mengaplikasikan teori ADT (Abstract Data Type) untuk setiap fitur program. Program yang dibuat merupakan suatu program rumah sakit Nimons. Beberapa fitur-fitur dalam program ini dapat mengubah kondisi dari data users dan config rumah sakit. Implementasi diawali dengan pembuatan fitur dasar yang mencakup **F01 - LOGIN** untuk memungkinkan pengguna masuk ke sistem dengan memasukkan username dan password, **F02 - REGISTER** untuk pendaftaran akun pasien baru dengan username unik, **F03 - LOGOUT** untuk keluar dari akun pengguna, dan **F04 - LUPA\_PASSWORD** digunakan jika user ter-logout dan hendak mengubah password mereka dengan verifikasi kode unik. Pengguna atau *user* dalam sistem ini terbagi menjadi tiga peran utama, yaitu dokter, pasien, dan manajer, di mana masing-masing peran memiliki hak akses fitur yang berbeda. Sebagai contoh, proses registrasi dokter hanya dapat dilakukan oleh manajer, sedangkan pasien dapat melakukan registrasi sendiri.

Dalam implementasi program terdapat beberapa fitur umum yang dapat diakses oleh seluruh role, seperti fitur **F05 - HELP** yang berfungsi untuk menampilkan seluruh perintah yang dapat digunakan oleh pengguna saat ini dan fitur **F06 - LIHAT\_DENAH** yang akan menampilkan denah rumah sakit. Di samping itu, terdapat fitur-fitur spesifik untuk masing-masing role. Untuk pasien, fitur-fitur yang tersedia antara lain **F14 - DAFTAR\_CHECKUP** untuk mendaftarkan diri ke pemeriksaan medis dengan memasukkan data kondisi tubuh, **F15 - ANTRIAN** untuk melihat status antrian pasien, **F13 - PULANGDOK** untuk berkonsultasi ulang dengan dokter sebelum dipulangkan dari rumah sakit, **F16 - MINUM\_OBAT** untuk melihat dan memilih obat yang akan diminum dari daftar obat yang dimiliki pasien, serta **F17 - PENAWAR** yang berfungsi untuk meminum penawar apabila pasien salah minum obat dan ingin membatalkan konsumsi terakhir. Sementara itu, role dokter memiliki dua fitur utama, yakni **F11 - DIAGNOSIS** untuk mendiagnosis pasien yang belum memiliki hasil diagnosis berdasarkan riwayat penyakit atau kondisi pasien, serta **F12 - NGOBATIN** untuk memberikan resep obat berdasarkan diagnosa yang telah dilakukan. Role terakhir adalah manajer, yang memiliki akses fitur paling luas dalam sistem. Manajer dapat menggunakan fitur **F10 - TAMBAH\_DOKTER** untuk mendaftarkan dokter baru, **F07 - LIHAT\_USER**, **F08 - CARI\_USER** yang memungkinkan pencarian berdasarkan ID atau nama, serta fitur **F09 - LIHAT\_SEMUA\_ANTRIAN** untuk memantau semua antrian ruangan yang aktif di rumah sakit. Program juga memiliki fitur **LOAD** (D03), di mana pada awal dijalankan program, bisa diberikan argumen untuk mengambil data dari folder mana, juga terdapat fitur **SAVE** (D04), di mana pengguna dapat menyimpan kondisi rumah sakit dan users di rumah sakit pada saat tersebut ke dalam folder mereka. Selain itu, terdapat **F18 - EXIT** untuk menyimpan data dan keluar dari program.

Setiap dari implementasi tersebut akan memanfaatkan ADT Sederhana, ADT List, ADT Linked List, ADT Matrix, ADT Set, ADT Map, ADT Stack, ADT Queue,

pemrosesan file external, fungsi dan prosedur, dan algoritma array search, sort, dan filter.

# Rencana Implementasi

Tabel 2.1 Rencana Implementasi

IMPLEMENTASI ADT	FITUR	DESKRIPSI IMPLEMENTASI	ALASAN IMPLEMENTASI
ADT List Dinamis	F01 - Login	Digunakan untuk menampung data users dari user.csv	Sebagai tempat penyimpanan data yang berada di dalam program
ADT List Dinamis ADT Set	F02 - Register	Memasukkan data pasien baru ke dalam list dinamis data  Memastikan keunikan dari username pasien yang diinput	Sebagai tempat penyimpanan data yang berada di dalam program  Karena ADT Set cocok untuk validasi string unik
-	F03 - Logout	-	-
ADT List Dinamis	F04 - Lupa Password	Digunakan untuk menampung data users dari user.csv	Sebagai tempat penyimpanan data yang berada di dalam program
-	F05 - Help	-	-
ADT Matriks	F06 - Lihat Denah, Lihat Ruangan	Digunakan untuk menampung data denah dari config.txt	Cocok dalam menyimpan data dua dimensi
ADT List Dinamis	F07 - Lihat User, Lihat Pasien, Lihat Dokter	Digunakan untuk menampung data users dari user.csv	Sebagai tempat penyimpanan data yang berada di dalam program
ADT List Dinamis	F08 - Cari User, Cari Pasien, Cari Dokter	Digunakan untuk menampung data users dari user.csv	Sebagai tempat penyimpanan data yang berada di dalam program
ADT Queue + Linked List	F09 - Lihat Antrian	Digunakan menampung antrian untuk data	Cocok karena sifat Queue yaitu Last In First Out, yang sesuai dengan konsep antrian
ADT List Dinamis ADT Set	F10 - Tambah Dokter, Assign Dokter	Memasukkan data dokter baru ke dalam list dinamis data  Memastikan keunikan dari username dokter yang diinput	Sebagai tempat penyimpanan data yang berada di dalam program  Karena ADT Set cocok untuk validasi string unik

ADT List Dinamis	F11 - Diagnosis	Digunakan untuk menampung data users dari user.csv dan data penyakit dari penyakit.csv	Sebagai tempat penyimpanan data yang berada di dalam program
ADT List Dinamis ADT Map	F12 - Ngobatin	Digunakan untuk menampung data users dari user.csv, data obat dari obat.csv, dan data penyakit dari penyakit.csv  Digunakan untuk menghubungkan penyakit dengan obat yang sesuai	Sebagai tempat penyimpanan data yang berada di dalam program  Cocok sebagai ADT yang memiliki pair key dan value, sehingga hubungan obat-penyakit jelas
ADT List Dinamis ADT Stack	F13 - Pulangdok	Digunakan untuk menampung data users dari user.csv, data obat dari obat.csv, dan data penyakit dari penyakit.csv  Digunakan untuk merepresentasikan riwayat obat / isi perut pasien	Sebagai tempat penyimpanan data yang berada di dalam program  Cocok karena sifat Stack Last In Last Out yang sama dengan konsep isi perut pasien
ADT List Dinamis ADT Queue + Linked List	F14 - Daftar Checkup	Digunakan untuk menampung data users dari user.csv,  Digunakan untuk menampung antrian	Sebagai tempat penyimpanan data yang berada di dalam program  Cocok karena sifat Queue yaitu First In First Out, yang sesuai dengan konsep antrian
ADT Queue + Linked List	F15 - Antrian	Digunakan untuk menampung antrian	Cocok karena sifat Queue yaitu First In First Out, yang sesuai dengan konsep antrian
ADT List Dinamis ADT Stack	F16 - Minum Obat + B05 - Dead or Alive	Digunakan untuk menampung data users dari user.csv, data obat dari obat.csv, dan data penyakit dari penyakit.csv  Digunakan untuk merepresentasikan riwayat obat / isi perut pasien	Sebagai tempat penyimpanan data yang berada di dalam program  Cocok karena sifat Stack Last In First Out yang sama dengan konsep isi perut pasien

ADT List Dinamis ADT Stack	F17 - Minum Penawar	Digunakan untuk menampung data users dari user.csv, data obat dari obat.csv, dan data penyakit dari penyakit.csv  Digunakan untuk merepresentasikan riwayat obat / isi perut pasien	Sebagai tempat penyimpanan data yang berada di dalam program  Cocok karena sifat Stack Last In First Out yang sama dengan konsep isi perut pasien
ADT List Dinamis	F18 - Exit	Digunakan sebagai tempat penyimpanan data sebelum disave	Sebagai tempat penyimpanan data yang berada di dalam program

# Pembagian Kerja

Tabel 3.1 Pembagian Kerja

Fitur	Implementasi	NIM Desainer	NIM Coder	NIM Tester
F01 - Login	ADT List Dinamis (user)	13524101 13524135	13524101	13524101 13524115
F02 - Register Pasien	ADT List Dinamis (user), ADT Set, function to_lower	13524101 13524003 13524135	13524101	13524101 13524115
F03 - Logout	ADT List Dinamis (user)	13524101	13524101	13524101 13524115
F04 - Lupa Password	ADT List Dinamis (user)	13524101 13524135	13524101	13524101 13524115
F05 - Help	ADT List Dinamis (user)	13524105	13524105	13524105 13524115
F06/D01 - Lihat Denah, Lihat Ruangan	ADT Matriks (denah)	13524003	13524105	13524105 13524115
F07 - Lihat User, Lihat Pasien, Lihat Dokter	ADT List Dinamis (user), procedure sortBased, alphabetSort, auraSort, tampilList	13524101	13524101	13524101 13524115
F08 - Cari User, Cari Pasien, Cari Dokter	ADT List Dinamis (user), procedure findBased, findId, findName, tampilPenyakit, findPenyakit, function binSearchId, seqSearchName	13524101	13524101	13524101 13524115
F09/D02 - Lihat Antrian	ADT Queue + Linked List (antrian)	13524135	13524135	13524115 13524135
F10 - Tambah Dokter, Assign Dokter	ADT List Dinamis (user), ADT Set, ADT Matriks (denah)	13524101 13524003 13524135	13524101	13524101 13524115
F11 - Diagnosis	ADT List Dinamis (user), ADT List Dinamis (penyakit), fungsi cekPenyakit, assignPenyakit, punyaRiwayat	13524003	13524105	13524105 13524115
F12 - Ngobatin	ADT List Dinamis (obat), ADT Map (obat-penyakit)	13524003 13524135	13524003	13524003 13524115
F13 - Aku boleh pulang ga, dok?	ADT List Dinamis (user), ADT Stack + Linked List, ADT Queue + Linked List, fungsi cekPerutDenganUrutan	13524003 13524135	13524003	13524003 13524115
F14 - Daftar	ADT List Dinamis (user), ADT	13524003	13524135	13524115

Check-Up	Matriks (denah), ADT Queue (antrian)	13524135		13524135
F15 - Antrian Saya!	ADT Queue + Linked List (antrian)	13524135	13524135	13524115 13524135
F16 - Minum Obat	ADT List Dinamis (obat), ADT Map (obat-penyakit), ADT Stack + Linked List (perut)	13524003 13524135	13524003	13524003 13524115
F17 - Minum Penawar	ADT Stack (perut), ADT Matriks (inventory)	13524003	13524003	13524003 13524115
F18 - Exit	ADT List Dinamis (user, obat, penyakit), ADT Map (obat-penyakit), ADT Matriks (denah), ADT Queue + Linked List (antrian), ADT Stack + Linked List (perut)	13524105	13524105	13524105 13524115
D03 - Load	ADT List Dinamis (user, obat, penyakit), ADT Map (obat-penyakit), ADT Matriks (denah), ADT Queue + Linked List (antrian), ADT Stack + Linked List (perut)	13524003 13524101 13524105 13524135	13524003 13524101 13524105	13524101 13524115
D04 - Save	ADT List Dinamis (user, obat, penyakit), ADT Map (obat-penyakit), ADT Matriks (denah), ADT Queue + Linked List (antrian), ADT Stack + Linked List (perut)	13524003 13524101 13524105 13524135	13524003 13524101 13524105	13524101 13524115
B02 - Denah Dinamis	ADT ADT Matriks (denah)	13524105	13524105	13524105 13524115
B03 - Aura!	ADT List Dinamis (user), procedure auraSort	13524101	13524101	13524101 13524115
B05 - Dead or Alive	ADT List Dinamis (user, obat) ADT Stack (perut), ADT Matriks (inventory)	13524003	13524003	13524003 13524115
B06 - Mainin Antrian	ADT Queue + Linked List	13524003 13524135	13524003	13524003 13524115

# Testing Primitif

Tabel 4.1 Testing primitif

Fitur	Desain	Implementasi	Testing
F01	V	V	V
F02	V	V	V
F03	V	V	V
F04	V	V	V
F05	V	V	V
F06	V	V	V
F07	V	V	V
F08	V	V	V
F09	V	V	V
F10	V	V	V
F11	V	V	V
F12	V	V	V
F13	V	V	V
F14	V	V	V
F15	V	V	V
F16	V	V	V
F17	V	V	V
F18	V	V	V
D03	V	V	V
D04	V	V	V
B02	V	V	V
B03	V	V	V
B05	V	V	V
B06	V	V	V

# Desain Command

## 1. F01 - Login

- Masukan:
  - username : string
  - password : string
- Keluaran:
  - Jika `username` dan `password` cocok: Pesan sambutan sesuai role (Manager, Dokter, Pasien) dan status login berubah.
  - Jika `username` tidak ditemukan: Pesan kesalahan "Tidak ada user di database..."
  - Jika `password` salah: Pesan kesalahan "Password yang dimasukkan salah!"

## 2. F02 - Register Pasien

- Masukan:
  - `username`: String (maksimal 20 karakter, hanya alfabet)
  - `password`: String (maksimal 20 karakter)
- Keluaran:
  - Jika registrasi berhasil: Pesan "Pasien [username] berhasil ditambahkan!" dan data pasien baru ditambahkan ke sistem.
  - Jika `username` sudah terdaftar: Pesan "Registrasi gagal! User dengan nama [username] sudah terdaftar."

## 3. F03 - Logout

- Masukan: -
- Keluaran:
  - Jika user sudah login: Status login direset dan pesan "Sampai Jumpa".
  - Jika user belum login: Pesan kesalahan "Logout gagal! Anda belum login..."

## 4. F04 - Lupa Password

- Masukan:
  - `username`: String (maksimal 20 karakter, hanya alfabet)
  - `KodeUnik`: String (hasil RLE dari username)
- Keluaran:
  - Jika `username` tidak ditemukan: Pesan kesalahan "Tidak ada user di database..."
  - Jika `KodeUnik` cocok: Meminta `password` baru dan memperbarui `password` user. Pesan sambutan sesuai role.
  - Jika `KodeUnik` salah: Pesan "Kode unik salah!"

## 5. F05 - Menu & Help

- Masukan: -

- Keluaran:
  - Jika belum login maka akan ditampilkan menu untuk login dan register
  - Jika sudah login dan merupakan manajer maka akan ditampilkan menu logout, tambah dokter, lihat denah, lihat user, lihat pasien, lihat dokter, cari user, cari pasien, cari dokter, lihat semua antrian, tambah dokter, dan assign dokter.
  - Jika sudah login dan merupakan dokter maka akan ditampilkan menu logout, diagnosis, ngobatin, dan lihat denah
  - Jika sudah login dan merupakan pasien maka akan ditampilkan menu logout, daftar check up, antrian, lihat denah, pulang dok, minum obat, dan minum penawar

6. F06 - Denah Rumah Sakit

- Lihat\_Denah:
  - Masukan : -
  - Keluaran : Mengeluarkan denah dari ruangan
- Lihat\_Ruangan
  - Masukan : Menerima masukan kode ruangan
  - Keluaran : Menampilkan data dari ruangan dengan kode ruangan yang telah diinput sebelumnya, seperti id dokter dan id pasien

7. F07 - Lihat User

- Masukan: Menerima jenis sort dan urutan sort (untuk dokter bisa berdasarkan aura)
- Keluaran: Menampilkan daftar seluruh user.

8. F08 - Cari User

- Masukan:
  - Pilihan pencarian: ID atau Nama atau penyakit (bagi pasien)
  - Nilai pencarian (ID atau Nama)
- Keluaran: Menampilkan user yang sesuai dengan kriteria pencarian.

9. F09 - Lihat Antrian

- Masukan : -
- Keluaran :
  - Gambar denah rumah sakit
  - Untuk tiap ruangan yang memiliki dokter, tampilkan nama-nama pasien dan orang-orang dalam antrian untuk ruangan tersebut

10. F10 - Tambah Dokter

- Masukan: Data dokter baru (username, password, dll.).
- Keluaran: Menambahkan dokter baru ke sistem.

11. F11 - Diagnosis

- Masukan: -
- Keluaran:
  - Jika pasien punya riwayat penyakit dan masih sakit maka akan ditampilkan informasi bahwa pasien masih sakit.
  - Jika pasien punya riwayat penyakit dan sudah sembuh maka akan ditampilkan pasien tidak terdiagnosa penyakit.
  - Jika pasien tidak punya riwayat penyakit dan terdiagnosa penyakit maka akan ditampilkan penyakit yang diderita dan di assign ke riwayat penyakit.
  - Jika pasien tidak punya riwayat penyakit dan tidak memiliki penyakit apapun maka akan ditampilkan bahwa pasien tidak terdiagnosa penyakit apapun.

#### 12. F12 - Ngobatin

- Masukan: -
- Keluaran:
  - Jika dokter belum terassign ruangan, tampilkan pesan belum terassign.
  - Jika dokter tidak memiliki pasien yang mengantri, tampilkan pesan tidak ada pasien.
  - Jika pasien pertama yang mengantri belum didiagnosis, tampilkan pesan pasien belum diagnosis.
  - Jika pasien pertama sudah menerima obat, tampilkan pesan pasien sudah menerima obat.
  - Selain itu, obat yang sesuai dengan penyakit pasien dimasukkan ke dalam inventory pasien, lalu tampilkan konfirmasi bahwa hal tersebut dilakukan.

#### 13. F13 - Aku boleh pulang ga, dok 😊 ?

- Masukan: -
- Keluaran:
  - Jika pasien belum check-up, tampilkan pesan belum check-up.
  - Jika pasien belum didiagnosis, tampilkan pesan belum didiagnosis.
  - Jika pasien belum diberi obat, tampilkan pesan belum menerima obat.
  - Jika pasien belum menghabiskan obat, tampilkan pesan belum menghabiskan obat.
  - Jika pasien sudah meminum semua obatnya namun dengan urutan yang salah, tampilkan pesan untuk memperbaiki urutan minumnya.
  - Selain itu, pasien boleh pulang (dihilangkan dari antrian dan direset datanya dalam program) serta tampilkan konfirmasi telah dilakukan hal tersebut.

#### 14. F14 - Daftar Check-Up

- Masukan :

- Data suhu tubuh : float
- Data tekanan darah : integer
- Data detak jantung : integer
- Data saturasi oksigen : float
- Data kadar gula darah : integer
- Data berat badan : float
- Data tinggi badan : integer
- Data kadar kolesterol : integer
- Data trombosit : integer
- Pilihan dokter : integer
- Keluaran :
  - Jika pasien sudah berada dalam antrian suatu ruangan, tampilkan pesan bahwa pasien sudah terdaftar dalam antrian check-up dan untuk menyelesaikan check-up yang sudah terdaftar terlebih dahulu.
  - Jika pasien memasukkan data kondisi tubuh yang invalid, tampilkan pesan kesalahan lalu mengulangi lagi proses input.
  - Jika pasien memasukkan pilihan dokter yang invalid, tampilkan pesan kesalahan lalu mengulangi lagi proses input
  - Jika pasien berhasil memasukkan data kondisi tubuh, memilih dokter, dan antrian tidak kosong, pasien akan masuk ke dalam antrian dan tampil pesan yang memberitahu bahwa pasien terdaftar di antrian dokter tersebut pada ruangan sekian serta posisi antrian sang pasien.
  - Jika pasien berhasil memasukkan data kondisi tubuh, memilih dokter, antrian kosong, dan ada pasien lain dalam ruangan, pasien akan masuk ke dalam ruangan dan tampil pesan yang memberitahu bahwa pasien dipersilahkan untuk langsung masuk.
  - Jika pasien berhasil memasukkan data kondisi tubuh, memilih dokter, antrian kosong, dan tidak ada pasien lain dalam ruangan, pasien akan masuk ke dalam ruangan dan tampil pesan yang memberitahu bahwa pasien dipersilahkan untuk langsung masuk dan akan diperiksa oleh dokter.

#### 15. F15 - Antrian Saya!

- Masukan : -
- Keluaran :
  - Jika pasien tidak terdapat dalam ruangan dokter maupun antrian, tampilkan pesan pasien belum melakukan terdaftar untuk check-up dan dapat melakukannya dengan command DAFTAR\_CHECKUP.
  - Jika pasien sudah berada dalam ruangan dokter, tampilkan pesan bahwa pasien sedang berada di dalam ruangan dokter.
  - Jika pasien sedang berada dalam antrian suatu ruangan, tampilkan pesan status antrian sang pasien yang berisikan nama dokter, ruangan, dan posisi antriannya di ruangan tersebut.

16. F16 - Minum Obat + B05 - Dead or Alive

- Masukan:
  - Pilihan obat: integer
- Keluaran:
  - Jika pasien tidak punya obat, tampilkan pesan pasien tidak punya obat.
  - Jika pasien punya obat, tampilkan list obat yang dimiliki.
  - Jika pilihan obat yang diberikan pasien tidak valid, tampilkan pesan pilihan obat tidak tepat.
  - Jika pilihan valid, obat yang dipilih dihilangkan pada inventory dan dimasukkan ke dalam perut pasien serta tampilkan pesan konfirmasi.
  - Jika pilihan obat tidak sesuai urutan, nyawa user berkurang sebanyak 1 serta tampilkan pesan konfirmasi.
  - Jika nyawa user bernilai 0, user dikeluarkan dari antrian, data user dihapus, dan program melakukan logout serta tampilkan pesan konfirmasi.

17. F17 - Minum Penawar

- Masukan: -
- Keluaran:
  - Jika tidak ada obat di dalam perut, tampilkan pesan perut pasien tidak memiliki obat.
  - Selain itu, obat dalam perut yang terakhir dimasukkan dikeluarkan dan dimasukkan kembali ke inventory serta tampilkan konfirmasi.

18. F18 - Exit

- Masukan: -
- Keluaran: Menyimpan data dan keluar dari program.

19. B02 - Denah dinamis

- Masukan:
  - ubahDenah: baris baru : integer  
kolom baru : integer
  - pindahDok : kode ruangan lama : string  
kode ruangan baru : string
- Keluaran:
  - ubahDenah :
    - Mengubah data configurasi baris dan kolom sesuai konfigurasi dari inputan user
  - pindahDok :
    - Memindahkan data dokter dan antrian ke ruangan baru jika ruangan baru tersebut valid dan kosong.



# Desain Kamus Data

Berikut adalah desain dari seluruh Abstract Data Type yang ada di program dalam notasi algoritmik.

## 1. ADT User (List)

```
type
    User : RECORD <
        id : integer,
        username : string,
        password : string,
        role : string,
        riwayat_penyakit : string,
        suhu_tubuh : real,
        tekanan_darah_sistolik : integer,
        tekanan_darah_diastolik : integer,
        detak_jantung : integer,
        saturasi_oksigen : real,
        kadar_gula_darah : integer,
        berat_badan : real,
        tinggi_badan : integer,
        kadar_kolesterol : integer,
        trombosit : integer,
        nyawa : integer,
        ruang : string,
        antrian : string,
        aura : integer
    >

{ Prosedur untuk menampilkan data User }
procedure printUser(input user : User)
{ I.S. user terdefinisi }
{ F.S. Data user ditampilkan }

{ Definisi Indeks }
constant
    IDX_MIN = 0
    IDX_UNDEF = -1

{ Definisi ElType dan IdxType }
type
    ElType : User
    IdxType : integer

{ Definisi ListDinUser }
type
    ListDinUser : RECORD <
        buffer : ARRAY [*] OF ElType, { Pointer ke array of ElType }
        nEff : integer, { Jumlah elemen efektif }
        capacity : integer { Kapasitas elemen }
    >

{ Fungsi Selektor }
```

```

function NEFF(l : ListDinUser) -> integer
function BUFFER(l : ListDinUser) -> ARRAY [*] OF ElType
function ELMT(l : ListDinUser, i : integer) -> ElType
function CAPACITY(l : ListDinUser) -> integer

{ Prosedur Konstruktor: Membuat list kosong }
procedure createListDinUser(input/output l : ListDinUser, input
capacity : integer)
{ I.S. l sembarang, capacity > 0 }
{ F.S. Terbentuk list dinamis l kosong dengan kapasitas
capacity }

{ Prosedur Dealokasi List }
procedure deallocateList(input/output l : ListDinUser)
{ I.S. l terdefinisi; }
{ F.S. (l) dikembalikan ke sistem, CAPACITY(l)=0; NEFF(l)=0 }

{ Fungsi Selektor (Tambah) }
function listLength(l : ListDinUser) -> integer
{ Mengirimkan banyaknya elemen efektif list }
{ Mengirimkan nol jika list l kosong }

{ Fungsi Selektor Indeks }
function getFirstIdx(l : ListDinUser) -> IdxType
{ Prekondisi : List l tidak kosong }
{ Mengirimkan indeks elemen l pertama }

function getLastIdx(l : ListDinUser) -> IdxType
{ Prekondisi : List l tidak kosong }
{ Mengirimkan indeks elemen l terakhir }

{ Fungsi Test Indeks yang Valid }
function isIdxValid(l : ListDinUser, i : IdxType) -> boolean
{ Mengirimkan true jika i adalah indeks yang valid utk
kapasitas list l }
{ yaitu antara indeks yang terdefinisi utk container}

function isIdxEff(l : ListDinUser, i : IdxType) -> boolean
{ Mengirimkan true jika i adalah indeks yang terdefinisi utk
list }
{ yaitu antara 0..NEFF(l) }

{ Fungsi Test Kosong/Penuh }
function isListDinEmpty(l : ListDinUser) -> boolean
{ Mengirimkan true jika list l kosong, mengirimkan false jika
tidak }

function isFull(l : ListDinUser) -> boolean
{ Mengirimkan true jika list l penuh, mengirimkan false jika
tidak }

{ Fungsi Mengembalikan indeks dengan username tertentu }

```

```

function indexOfUsername(l : ListDinUser, username : string) ->
IdxType
{ I.S. list dan username terdefinisi }
{ F.S. Mengembalikan indeks user dengan username tertentu, atau
IDX_UNDEF jika tidak ditemukan }

{ Prosedur Operasi Lain }
procedure copyList(input lIn : ListDinUser, input/output lOut :
ListDinUser)
{ I.S. lIn terdefinisi tidak kosong, lOut sembarang }
{ F.S. lOut berisi salinan dari lIn (identik, nEff dan capacity
sama) }
{ Proses : Menyalin isi lIn ke lOut }

{ Prosedur Menambah dan Menghapus Elemen di Akhir }
procedure insertLast(input/output l : ListDinUser, val : ElType)
{ Proses: Menambahkan val sebagai elemen terakhir list }
{ I.S. List l boleh kosong, tetapi tidak penuh }
{ F.S. val adalah elemen terakhir l yang baru }

procedure deleteLast(input/output l : ListDinUser, output val :
ElType)
{ Proses : Menghapus elemen terakhir list }
{ I.S. List tidak kosong }
{ F.S. val adalah nilai elemen terakhir l sebelum penghapusan,
}
{ Banyaknya elemen list berkurang satu }
{ List l mungkin menjadi kosong }

{ Prosedur Memasukkan item pada elemen pada list sesuai ID user }
procedure insertUserByID(input/output list : ListDinUser, input
item : ElType)
{ I.S. list dan item terdefinisi. item.id bernilai dan dapat
menjadi indeks yang valid }
{ F.S. list.buffer[item.id] terisi dengan data pada item }
{ jika item.id >= list.nEff, maka list.nEff = item.id + 1 }

{ Prosedur Menampilkan list data user }
procedure printList(input l : ListDinUser)
{ I.S. list user terdefinisi }
{ F.S. Ditampilkan list data user }

{ Prosedur Menyortir List user berdasarkan id }
procedure sortUser(input/output list : ListDinUser)
{ I.S. list terdefinisi }
{ F.S. list tersortir ascending menurut id }

{ Fungsi Mengembalikan indeks user dengan id tertentu }
function cariIdxUser(input/output UserData : ListDinUser, id :
integer) -> integer
{ I.S. UserData dan id terdefinisi }
{ F.S. Mengembalikan indeks user dengan id tertentu, atau
IDX_UNDEF jika tidak ditemukan }

```

```

{ Fungsi Mengembalikan nilai bertipe User yang memiliki
komponen-komponen default. }
function createEmptyUser() -> User
    { I.S. - }
    { F.S. Mengembalikan nilai bertipe User yang memiliki
komponen-komponen default. }

{ Prosedur Mereset data user, digunakan ketika ingin memulangkan
pasien. }
procedure resetUserData(input/output user : User)
    { I.S. user terdefinisi, user merupakan pasien. }
    { F.S. Semua data user kecuali ID, username, password, dan role
direset. }

{ Menghilangkan user dengan indeks tertentu dari list. }
procedure deleteUser(input/output list: ListDinUser, input idx:
integer);
    { I.S. list dan idx terdefinisi. }
    { F.S. Semua komponen yang ada pada elemen indeks idx direset
menjadi default. }

{ Prosedur Menuliskan List User ke dalam file }
procedure writeListUser(input folderPath : string, input/output
list : ListDinUser)
    { I.S. list terdefinisi dan terisi }
    { F.S. File user.csv berada pada folder yang ditujukan }

```

## 2. ADT Obat (List)

```

constant MAX_NAMA_OBAT : integer = 21

type Obat : <
    obat_id : integer,
    nama_obat : string
>

type ListObat : <
    items : pointer to array of Obat,
    nEff : integer,
    capacity : integer
>

procedure CreateListObat(output list : ListObat, input capacity
: integer)
{ I.S. list terdefinisi sembarang, capacity terdefinisi }
{ F.S. Memori dialokasikan untuk list; list.nEff ← 0;
list.capacity ← capacity }

procedure PrintListObat(input list : ListObat)
{ I.S. list boleh kosong }
{ F.S. Jika tidak kosong, seluruh elemen list dicetak ke layar }

```

```

procedure InsertObat(input/output list : ListObat, input item :
Obat)
{ I.S. list dan item terdefinisi; item.obat_id belum bernilai
(default) }
{ F.S. item dimasukkan di indeks list.nEff; list.nEff ← list.nEff
+ 1 }

procedure InsertObatByID(input/output list : ListObat, input
item : Obat)
{ I.S. list dan item terdefinisi; item.obat_id adalah indeks yang
valid }
{ F.S. item dimasukkan di indeks item.obat_id;
jika item.obat_id ≥ list.nEff maka list.nEff ← item.obat_id
+ 1 }

procedure FreeListObat(input/output list : ListObat)
{ I.S. list terdefinisi sembarang }
{ F.S. Memori list dilepas; list.nEff dan list.capacity bernilai
0 }

procedure SortObat(input/output list : ListObat)
{ I.S. list terdefinisi }
{ F.S. list tersortir secara ascending berdasarkan obat_id }

function CariIdxObat(input list : pointer to ListObat, input id
: integer) → integer
{ Mengembalikan indeks dalam list sesuai dengan id obat }

procedure WriteListObat(input folderPath : string, input list :
pointer to ListObat)
{ I.S. list terdefinisi dan terisi }
{ F.S. Isi list ditulis ke file 'obat.csv' dalam folder
folderPath }

```

### 3. ADT Penyakit (List)

```

type Penyakit : <
    id : integer,
    nama_penyakit : string,
    suhu_tubuh_min : real,
    suhu_tubuh_max : real,
    tekanan_darah_sistolik_min : integer,
    tekanan_darah_sistolik_max : integer,
    tekanan_darah_diastolik_min : integer,
    tekanan_darah_diastolik_max : integer,
    detak_jantung_min : integer,
    detak_jantung_max : integer,
    saturasi_oksigen_min : real,
    saturasi_oksigen_max : real,
    kadar_gula_darah_min : integer,
    kadar_gula_darah_max : integer,
    berat_badan_min : real,
    berat_badan_max : real,
    tinggi_badan_min : integer,

```

```

tinggi_badan_max : integer,
kadar_kolesterol_min : integer,
kadar_kolesterol_max : integer,
trombosit_min : integer,
trombosit_max : integer
>

type ListPenyakit : <
    items : pointer to array of Penyakit,
    nEff : integer,
    capacity : integer
>

procedure CreateListPenyakit(output list : ListPenyakit, input
capacity : integer)
{ I.S. list terdefinisi dan sembarang, capacity terdefinisi }
{ F.S. Memori untuk list sudah teralokasikan, list.nEff ← 0,
list.capacity ← capacity }

procedure PrintListPenyakit(input list : ListPenyakit)
{ I.S. list boleh kosong }
{ F.S. Jika list tidak kosong, tampilkan seluruh elemen ke layar
}

procedure InsertPenyakit(input/output list : ListPenyakit, input
item : Penyakit)
{ I.S. list dan item terdefinisi; item.id belum bernilai
(default) }
{ F.S. list[nEff] terisi data item, list.nEff ← list.nEff + 1 }

procedure InsertPenyakitByID(input/output list : ListPenyakit,
input item : Penyakit)
{ I.S. list dan item terdefinisi; item.id merupakan indeks yang
valid }
{ F.S. list[item.id] ← item; jika item.id ≥ list.nEff, maka
list.nEff ← item.id + 1 }

function SearchPenyakitIDByName(input list : ListPenyakit, input
nama : string) → integer
{ I.S. list dan nama terdefinisi }
{ F.S. Mengembalikan id penyakit jika ditemukan nama yang sesuai,
-1 jika tidak ditemukan }

procedure FreeListPenyakit(input/output list : ListPenyakit)
{ I.S. list terdefinisi dan sembarang }
{ F.S. Memori dalam list terdeallocate; list.nEff dan
list.capacity ← 0 }

procedure SortPenyakit(input/output list : ListPenyakit)
{ I.S. list terdefinisi }
{ F.S. list tersortir ascending berdasarkan id }

```

```

function CariIdxPenyakit(input list : pointer to ListPenyakit,
input id : integer) → integer
{ Mengembalikan indeks dalam list yang memiliki id penyakit
sesuai }

procedure WriteListPenyakit(input folderPath : string, input
list : pointer to ListPenyakit)
{ I.S. list terdefinisi dan terisi }
{ F.S. File penyakit.csv berada pada folder yang dituju dan
berisi isi list }

4. ADT Obat-Penyakit (Map)

constant MAX_OBAT_PER_PENYAKIT : integer = 10
constant MAX_MAP_SIZE : integer = 100

type ValueObat : <
    obat_id : integer,
    urutan_minum : integer
>

type MapPairObatPenyakit : <
    penyakit_id : integer,
    value : array [0..MAX_OBAT_PER_PENYAKIT - 1] of ValueObat
>

type MapObatPenyakit : <
    items : array [0..MAX_MAP_SIZE - 1] of MapPairObatPenyakit,
    size : integer
>

procedure CreateMapObatPenyakit(output map : MapObatPenyakit)
{ I.S. map terdefinisi dan sembarang }
{ F.S. map.size ← 0; setiap map.items[i].penyakit_id ← 0;
      setiap value dalam map.items[i] memiliki value.obat_id ← 0
}

procedure InsertObatPenyakitByID(input/output map :
MapObatPenyakit,
procedure InsertObatPenyakitByRawData(input/output map :
MapObatPenyakit,
integer,
integer,
integer)

```

```

{ I.S. map, obatId, penyakitId, urutanMinum terdefinisi dan valid
}
{ F.S. map.items[penyakitId].penyakit_id ← penyakitId;
      map.items[penyakitId].value[urutanMinum].urutan_minum ←
      urutanMinum;
      map.items[penyakitId].value[urutanMinum].obat_id ← obatId;
      jika penyakitId ≥ map.size maka map.size ← penyakitId + 1 }

```

**procedure** PrintMapObatPenyakit(input map : MapObatPenyakit)

{ I.S. map terdefinisi }

{ F.S. Seluruh isi data map ditampilkan ke layar }

**procedure** WriteMapObatPenyakit(input folderPath : string,
input mapObatPenyakit : pointer to MapObatPenyakit)
{ I.S. mapObatPenyakit terdefinisi dan terisi }
{ F.S. Isi map ditulis ke file yang sesuai pada folder yang dituju }

## 5. ADT Config

```

constant MAX_PASIEN : integer = 1000
constant MAX_OBAT : integer = 10

```

```

type Config : <
    denah : MatriksDenah,
    kapasitasRuang : integer,
    kapasitasAntrian : integer,
    jumlahPemilikObat : integer,
    inventoryPasien : Matriks,
    jumlahPerutPasien : integer,
    perutPasien : array [0..MAX_PASIEN - 1] of Stack
>

```

**procedure** CreateConfig(output config : Config)

{ I.S. config terdefinisi sembarang }

{ F.S. Seluruh komponen dalam config terinisialisasi dengan nilai default }

**procedure** PrintConfig(input config : Config)

{ I.S. config terdefinisi dan terisi }

{ F.S. Seluruh isi data dalam config ditampilkan ke layar }

**procedure** WriteConfig(input folderPath : string, input config : pointer to Config)

{ I.S. config terdefinisi dan terisi }

{ F.S. Data config dituliskan ke file config.txt di dalam folder folderPath }

## 6. ADT Denah (Matriks)

```

constant MAX_KODE_RUANG_LENGTH : integer = 10
constant MAX_ROWS : integer = 100
constant MAX_COLS : integer = 100

```

```

type KontenDenah : <
    kodeRuangan : string,
    dokterID : integer,
    antrian : pointer to Queue
>

type MatriksDenah : <
    contents : array [0..MAX_ROWS - 1, 0..MAX_COLS - 1] of
KontenDenah,
    rows : integer,
    cols : integer
>

procedure InitializeMatriksDenah(output matriks : MatriksDenah)
{ I.S. matriks sembarang, rows dan cols sudah ditentukan }
{ F.S. matriks terinisialisasi sebagai array 2D berukuran rows ×
cols }

procedure FreeMatriksDenah(input/output matriks : MatriksDenah)
{ I.S. matriks terdefinisi dan mungkin berisi pointer antrian }
{ F.S. Seluruh alokasi dinamis dalam matriks (termasuk antrian)
dibebaskan }

function ConvertNumberToKodeRuangan(input row : integer, input
col : integer) → string
{ I.S. row dan col terdefinisi }
{ F.S. Mengembalikan string kode ruangan hasil konversi dari
indeks baris dan kolom }

```

## 7. ADT Matriks

```

type
    Matriks : RECORD <
        contents : ARRAY [*][*] OF integer, { Pointer ganda ke array
integer 2D }
        rows : integer,
        cols : integer
    >

{ Prosedur Inisialisasi Matriks }
procedure initializeMatriks(input/output arr : Matriks, input
rows : integer, input cols : integer)
{ I.S. matriks bertipe Matriks (sembarang) terdefinisi, rows
dan cols terdefinisi }
{ F.S. matriks berbentuk array 2D dengan ukuran rows × cols }

{ Prosedur Membebaskan Memori Matriks }
procedure freeMatriks(input/output arr : Matriks)
{ I.S. arr terdefinisi }
{ F.S. Memori yang dialokasikan untuk arr dibebaskan }

{ Prosedur Memasukkan Data ke dalam Matriks Berdasarkan Indeks }
procedure insertMatrixByIndex(input/output matriks : Matriks,
input row : integer, input col : integer, input data : integer)

```

```

    { I.S. Semua argumen terdefinisi, matriks.contents[row][col]
valid }
    { F.S. matriks.contents[row][col] terisi data.
        matriks.rows = row + 1 jika row >= matriks.rows
        matriks.cols = col + 1 jika col >= matriks.cols }

{ Fungsi Memeriksa Apakah Matriks Kosong }
function isMatriksEmpty(matriks : Matriks) -> integer
    { I.S. matriks terdefinisi }
    { F.S. Mengembalikan 1 jika kosong (semua elemen bernilai 0), 0
jika tidak }

{ Fungsi Memeriksa Apakah Suatu Baris pada Matriks Kosong }
function isMatriksRowEmpty(matriks : Matriks, input row :
integer) -> integer
    { I.S. matriks dan row terdefinisi }
    { F.S. Mengembalikan 1 jika baris kosong (semua elemen bernilai
0), 0 jika tidak }

{ Fungsi Mengembalikan Banyak Ruangan pada Denah }
function matriksSize(input/output arr : Matriks) -> integer
    { I.S. arr terdefinisi }
    { F.S. Mengembalikan banyak ruangan pada denah (ukuran matriks)
}

```

## 8. ADT Linked-List

```

type Node : <
    data : integer,
    next : pointer to Node
>

function CreateLinked(data : integer) -> pointer to Node
{ Menghasilkan simpul baru dengan nilai data, next di-set ke NIL
}

procedure LinkedInsertHead(input/output headPointer : pointer to
Node, input data : integer)
{ I.S. headPointer terdefinisi sembarang, data terdefinisi }
{ F.S. Simpul baru berisi data ditambahkan di awal linked list }

procedure LinkedDeleteHead(input/output head : pointer to Node)
{ I.S. head terdefinisi }
{ F.S. Simpul pertama dihapus jika tidak kosong;
    jika kosong, tampilkan pesan kesalahan }

procedure LinkedInsertAt(input/output headPointer : pointer to
Node,
                        input position : integer, input data :
integer)
{ I.S. headPointer terdefinisi dan tidak kosong;
    position dalam rentang yang valid; data terdefinisi }

```

```

{ F.S. Simpul baru disisipkan di posisi ke-position dalam linked
list }

procedure LinkedDeleteAt(input/output headPointer : pointer to
Node,
                           input position : integer)
{ I.S. headPointer terdefinisi dan tidak kosong;
    position dalam rentang yang valid }
{ F.S. Simpul di posisi ke-position dihapus dari linked list }

```

## 9. ADT Stack-Linked-List

```

type Stack : <
    head : pointer to Node,
    length : integer
>

procedure CreateStack(output stack : Stack)
{ I.S. stack terdefinisi sembarang }
{ F.S. stack.head ← NIL dan stack.length ← 0 }

function IsStackEmpty(stack : Stack) → boolean
{ Menghasilkan true jika stack kosong (head = NIL), false jika
tidak }

procedure Push(input/output stack : Stack, input data : integer)
{ I.S. stack dan data terdefinisi }
{ F.S. jika stack tidak penuh, data masuk pada bagian atas stack;
    jika gagal, keluarkan pesan kesalahan }

function Pop(input/output stack : Stack) → integer
{ I.S. stack terdefinisi }
{ F.S. elemen paling atas dihapus dari stack dan nilainya
dikembalikan }

function PeekStack(stack : Stack) → integer
{ Menghasilkan nilai elemen paling atas stack tanpa menghapusnya;
    jika stack kosong, hasilnya -1 }

procedure PrintStack(input stack : Stack)
{ I.S. stack terdefinisi }
{ F.S. semua elemen dalam stack ditampilkan ke layar }

```

## 10. ADT Queue-Linked-List

```

type Queue : <
    front : pointer to Node,
    rear : pointer to Node,
    counter : integer
>

function CreateNode(data : integer) → pointer to Node
{ Menghasilkan simpul baru dengan nilai data, next ← NIL }

function CreateQueue() → pointer to Queue

```

```

{ Menghasilkan queue baru yang kosong (front = rear = NIL,
counter = 0) }

function IsQueueEmpty(q : pointer to Queue) → boolean
{ Menghasilkan true jika queue kosong (front = NIL), false jika
tidak }

procedure Enqueue(input/output q : pointer to Queue, input data
: integer)
{ I.S. q dan data terdefinisi }
{ F.S. data dimasukkan ke bagian belakang queue (rear) }

function Dequeue(input/output q : pointer to Queue) → integer
{ I.S. q terdefinisi dan tidak kosong }
{ F.S. elemen pada bagian depan queue dihapus dan nilainya
dikembalikan }

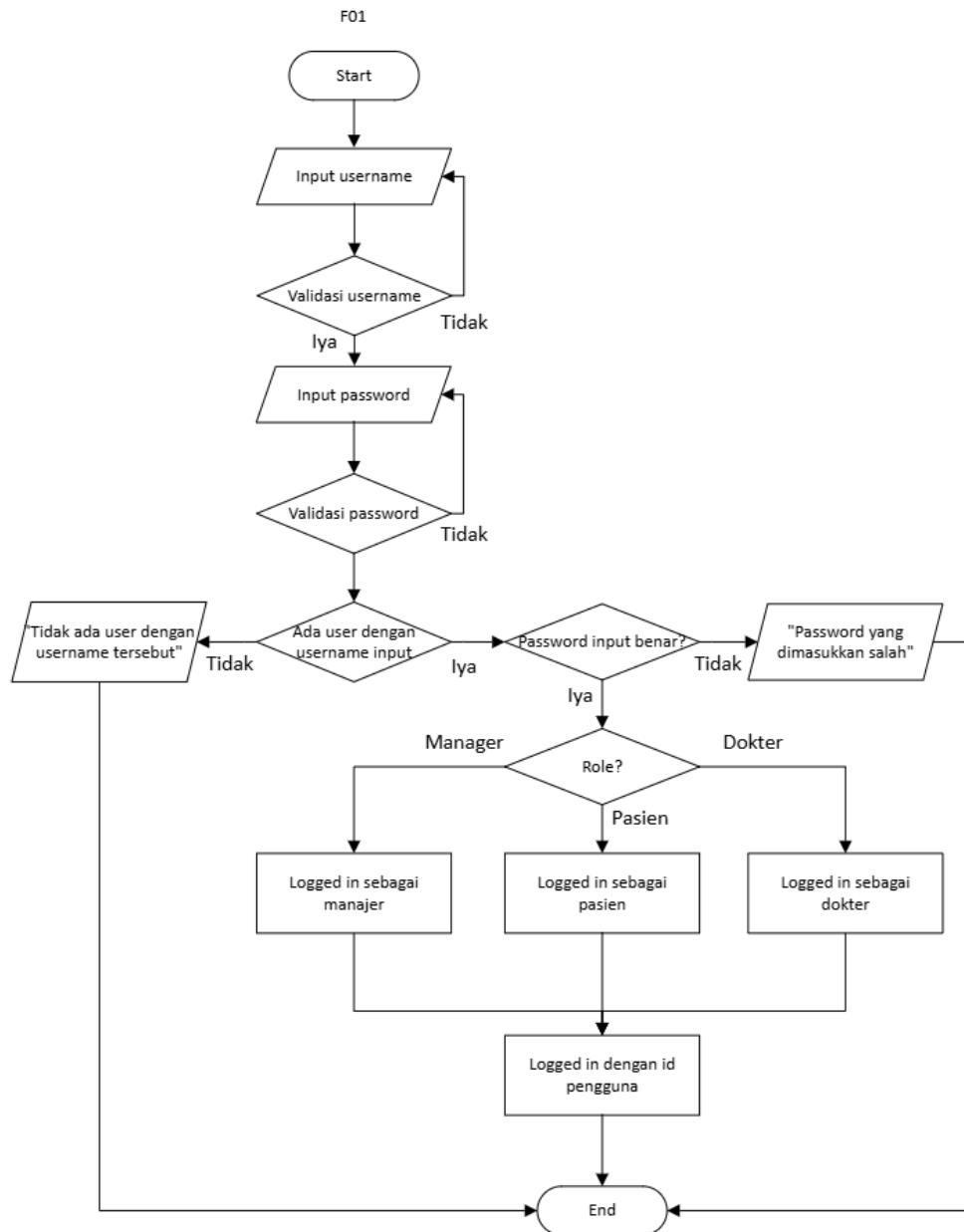
function PeekQueue(q : pointer to Queue) → integer
{ I.S. q terdefinisi }
{ F.S. Menghasilkan nilai elemen paling depan tanpa menghapusnya;
jika queue kosong, hasilnya -1 }

procedure PrintQueue(input q : pointer to Queue)
{ I.S. q terdefinisi }
{ F.S. Menampilkan semua elemen dalam queue dari front ke rear ke
layar }

```

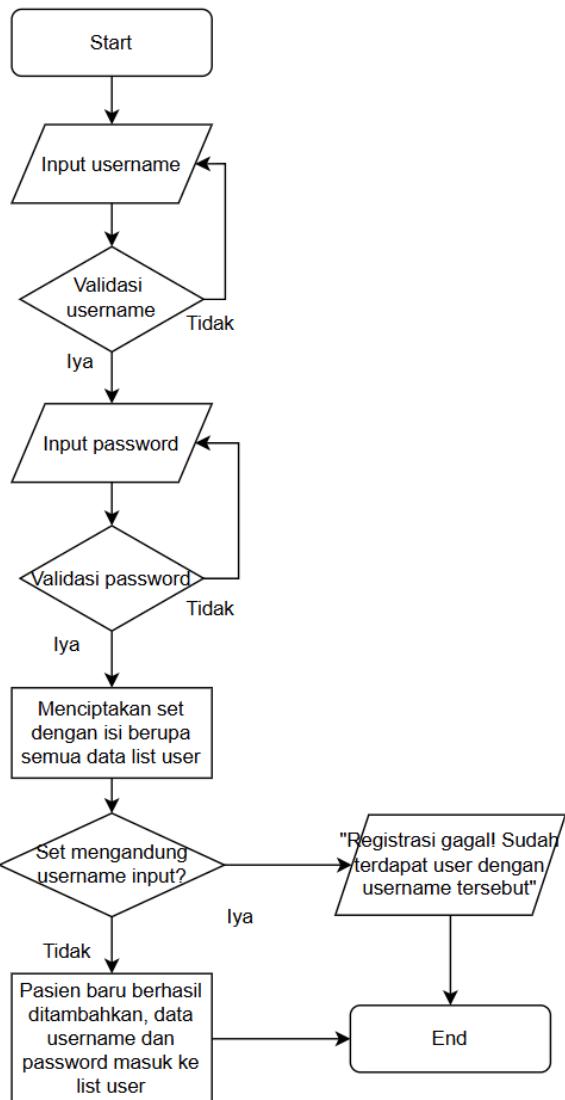
# Desain Dekomposisi

## 1. F01 - Login



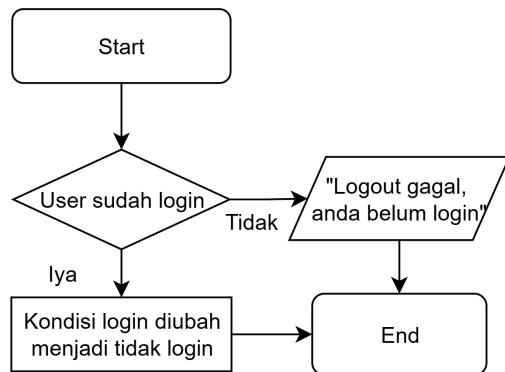
Gambar 7.1 Flowchart F01 - Login

## 2. F02 - Register Pasien



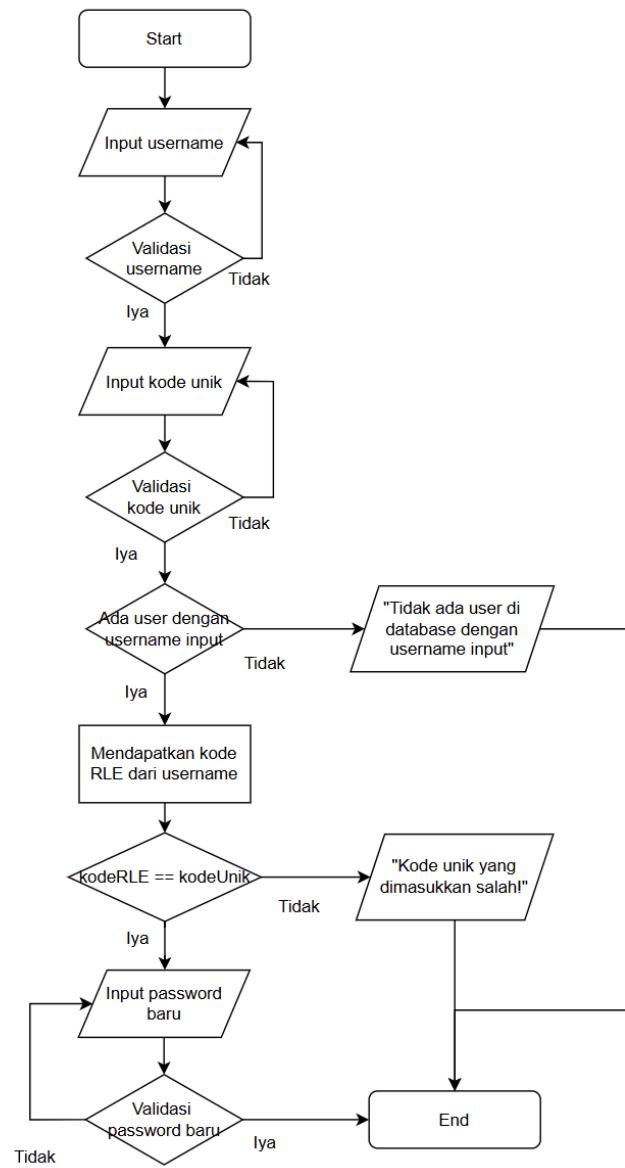
Gambar 7.2 Flowchart F02 - Register Pasien

## 3. F03 - Logout



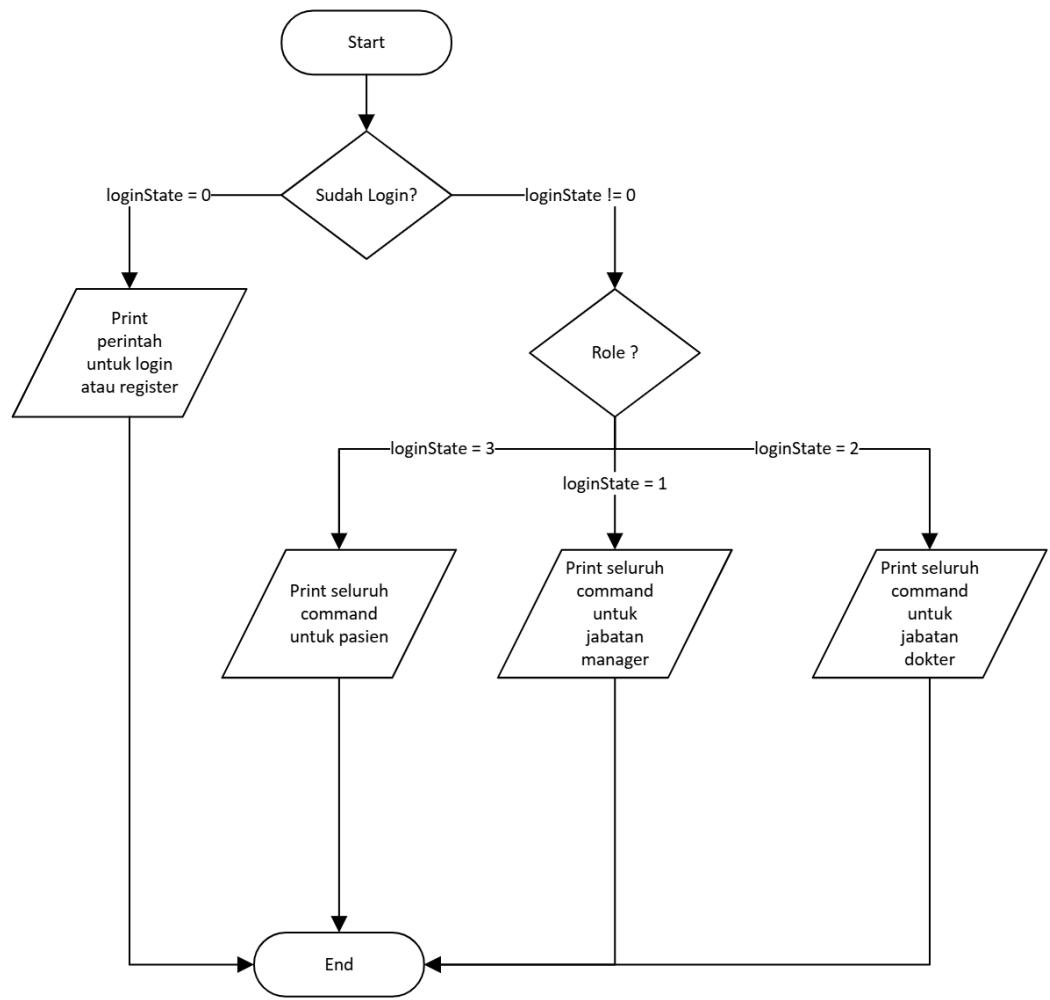
Gambar 7.3 Flowchart F03 - Logout

#### 4. F04 - Lupa Password



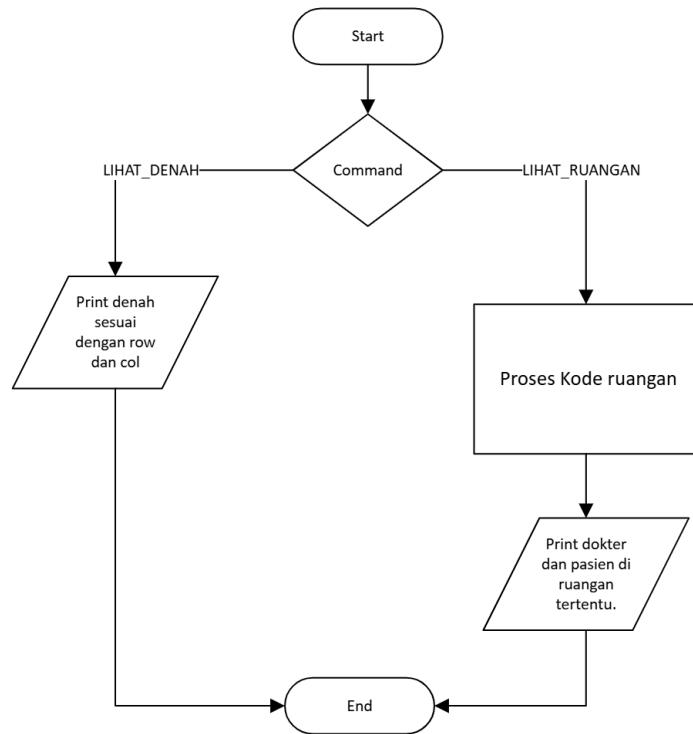
Gambar 7.4 Flowchart F04 - Lupa Password

## 5. F05 - Menu & Help



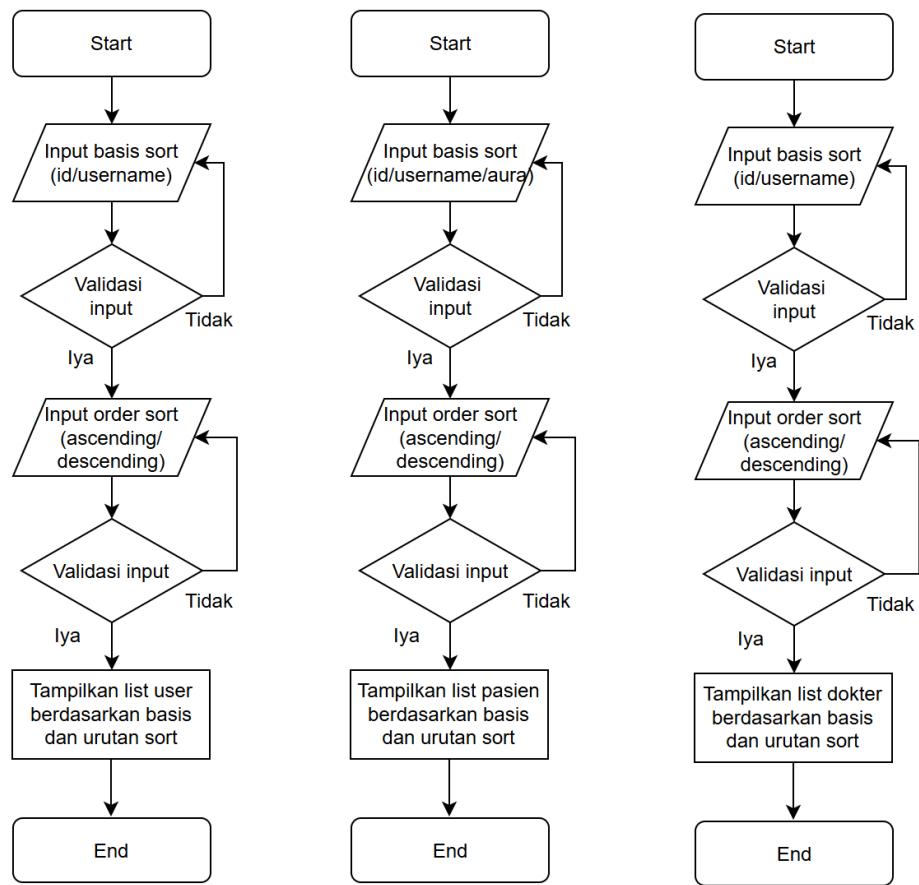
Gambar 7.5 Flowchart F05 - Menu & Help

## 6. F06 - Denah Rumah Sakit



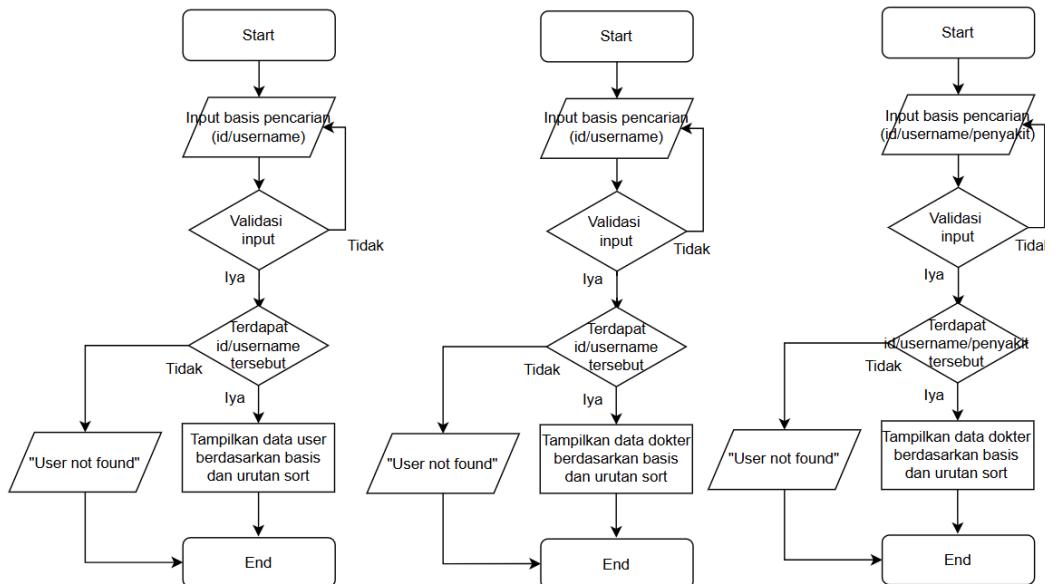
Gambar 7.6 Flowchart F06 - Denah Rumah Sakit

## 7. F07 - Lihat User



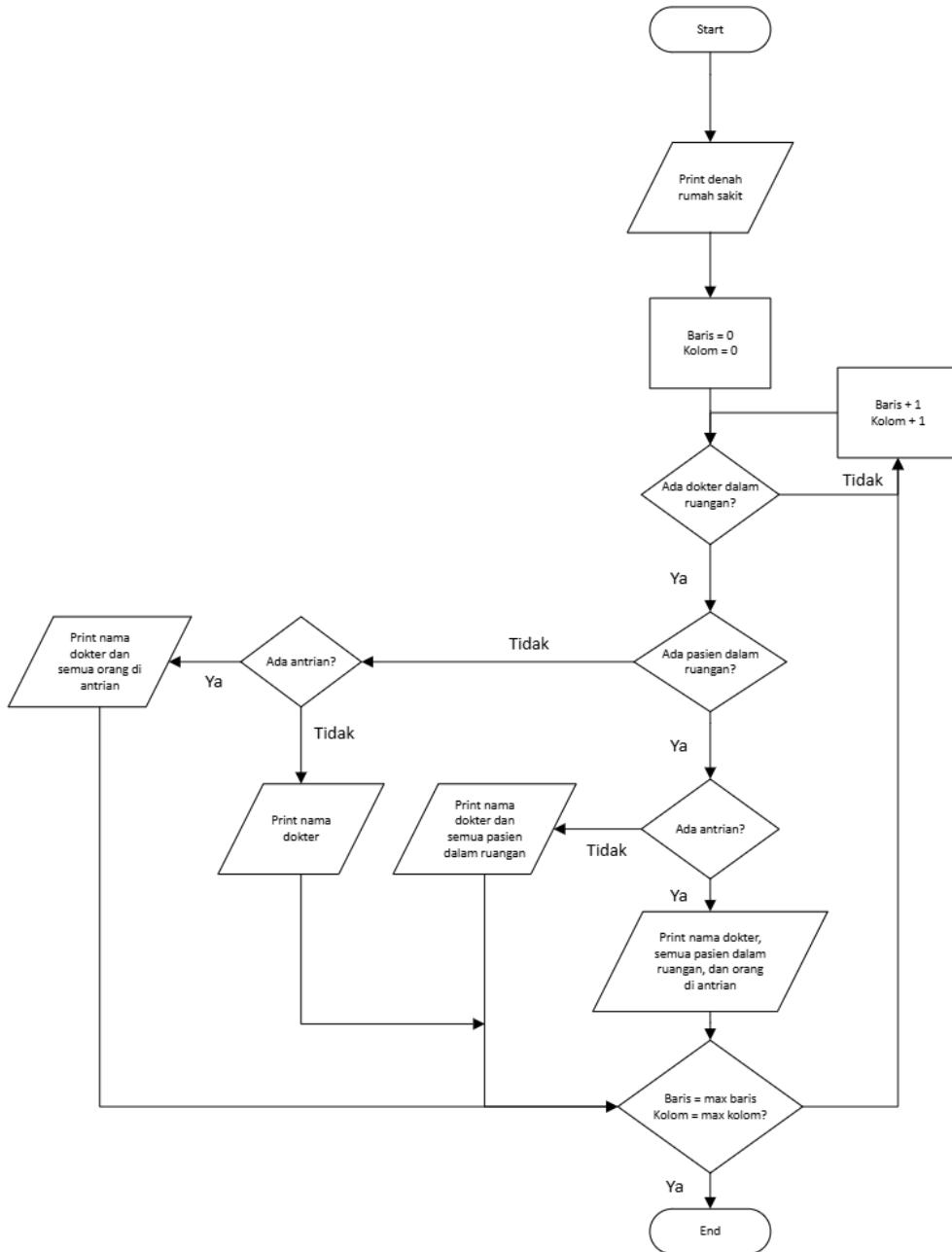
Gambar 7.7 Flowchart F07 - Lihat User, Lihat Pasien, Lihat Dokter

## 8. F08 - Cari User



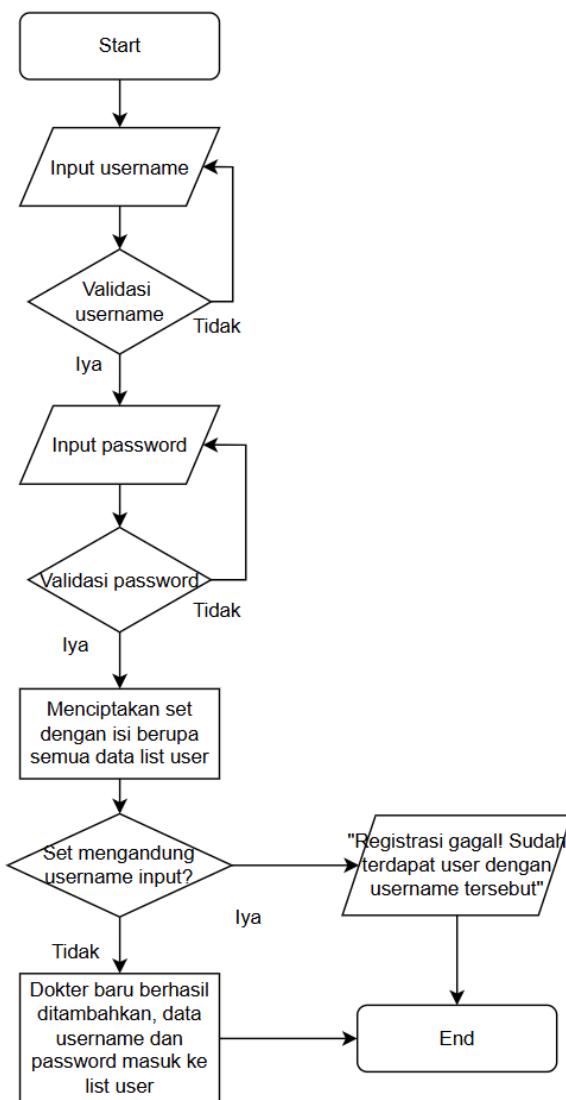
Gambar 7.8 Flowchart F08 - Cari User, Cari Dokter, Cari Pasien

## 9. F09 - Lihat Antrian

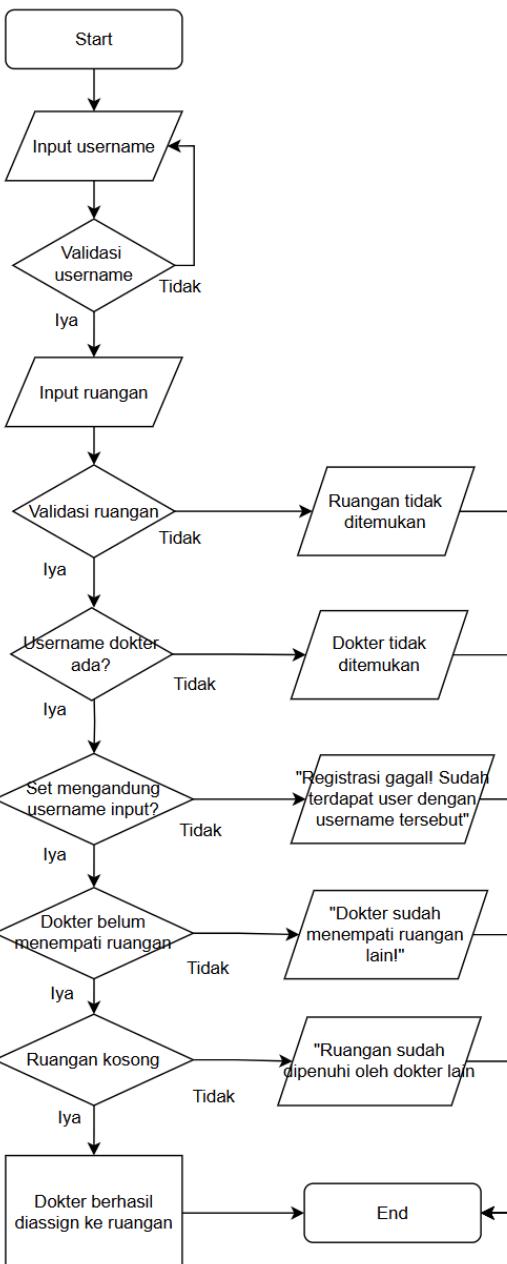


Gambar 7.9 Flowchart F09 - Lihat Antrian

#### 10. F10 - Tambah Dokter

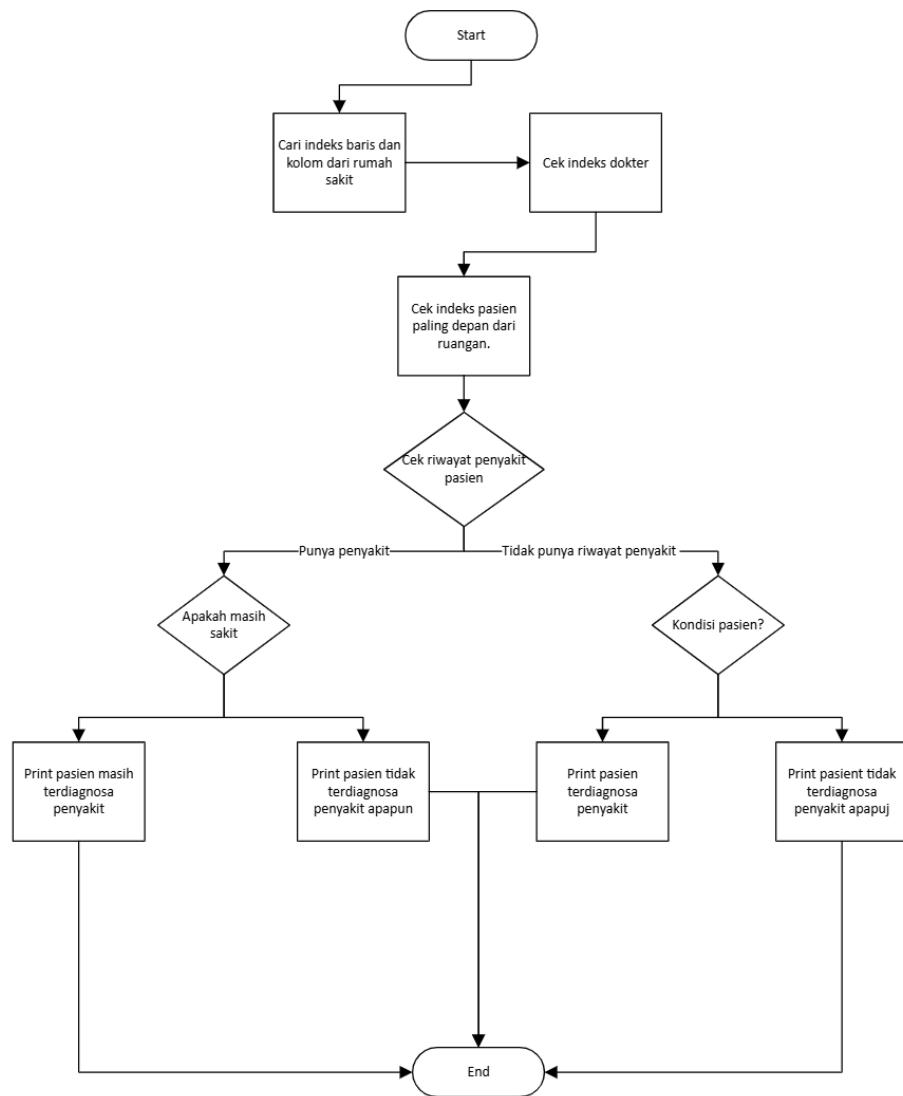


Gambar 7.10a Flowchart F10 - Tambah Dokter



Gambar 7.10b Flowchart F10 - Assign Dokter

## 11. F11 - Diagnosis



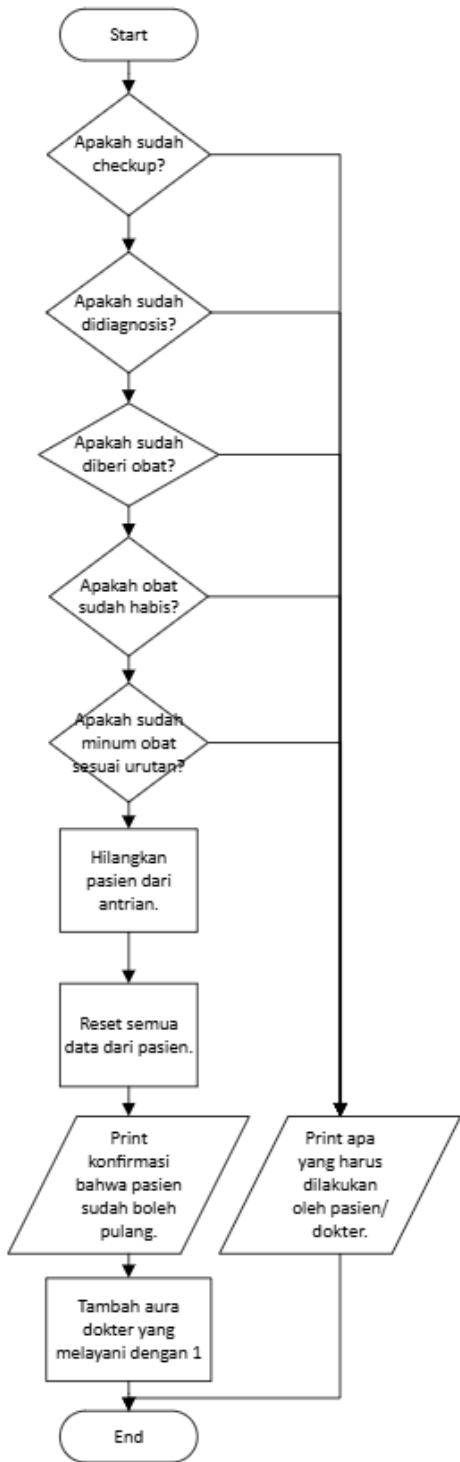
Gambar 7.11 Flowchart F11 - Diagnosis

## 12. F12 - Ngobatin



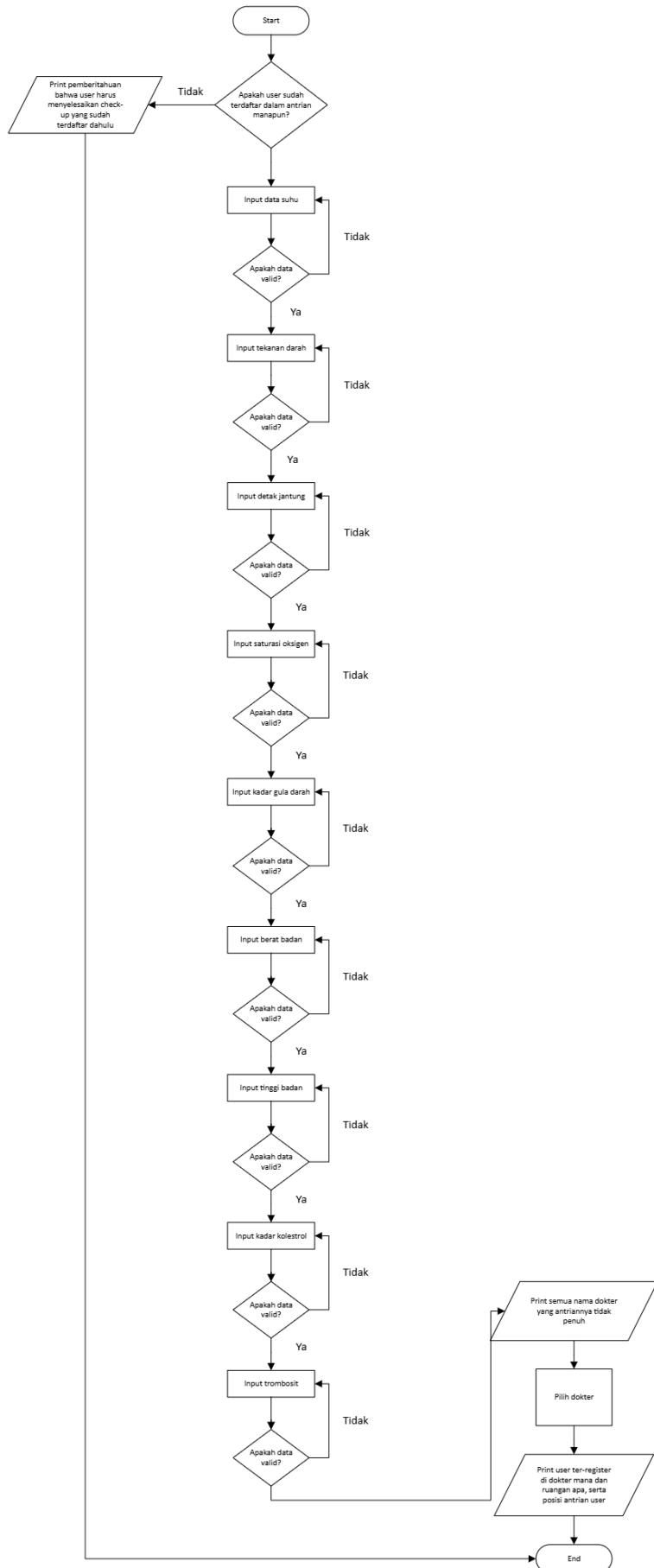
Gambar 7.12 Flowchart F12 - Ngobatin

13. F13 - Aku boleh pulang ga, dok 😊 ?



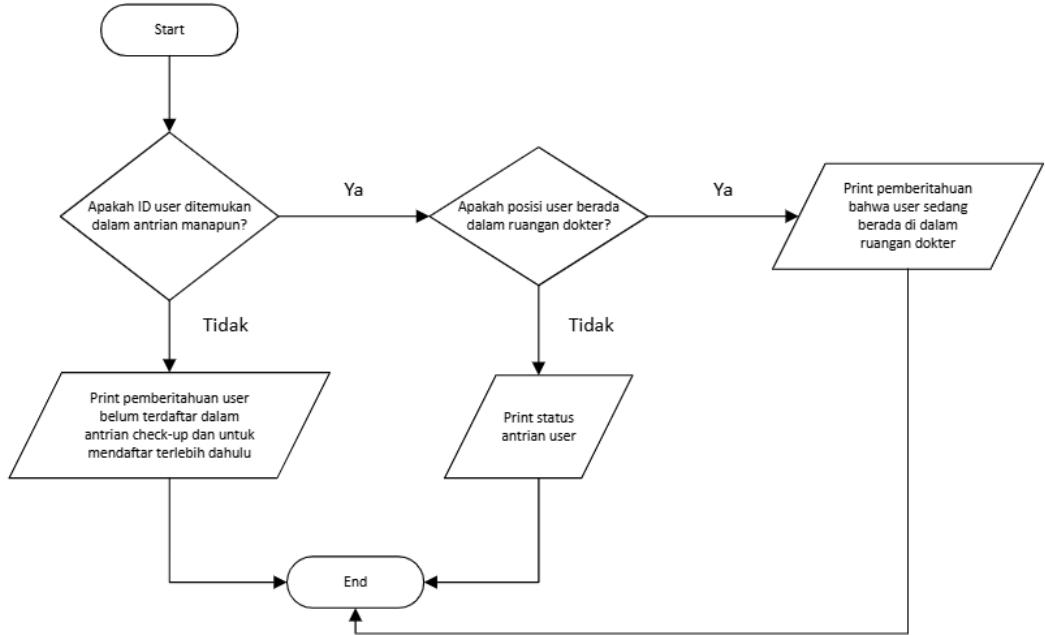
Gambar 7.13 Flowchart F13 - Aku Boleh Pulang Ga, Dok 😊 ?

14. F14 - Daftar Check-Up



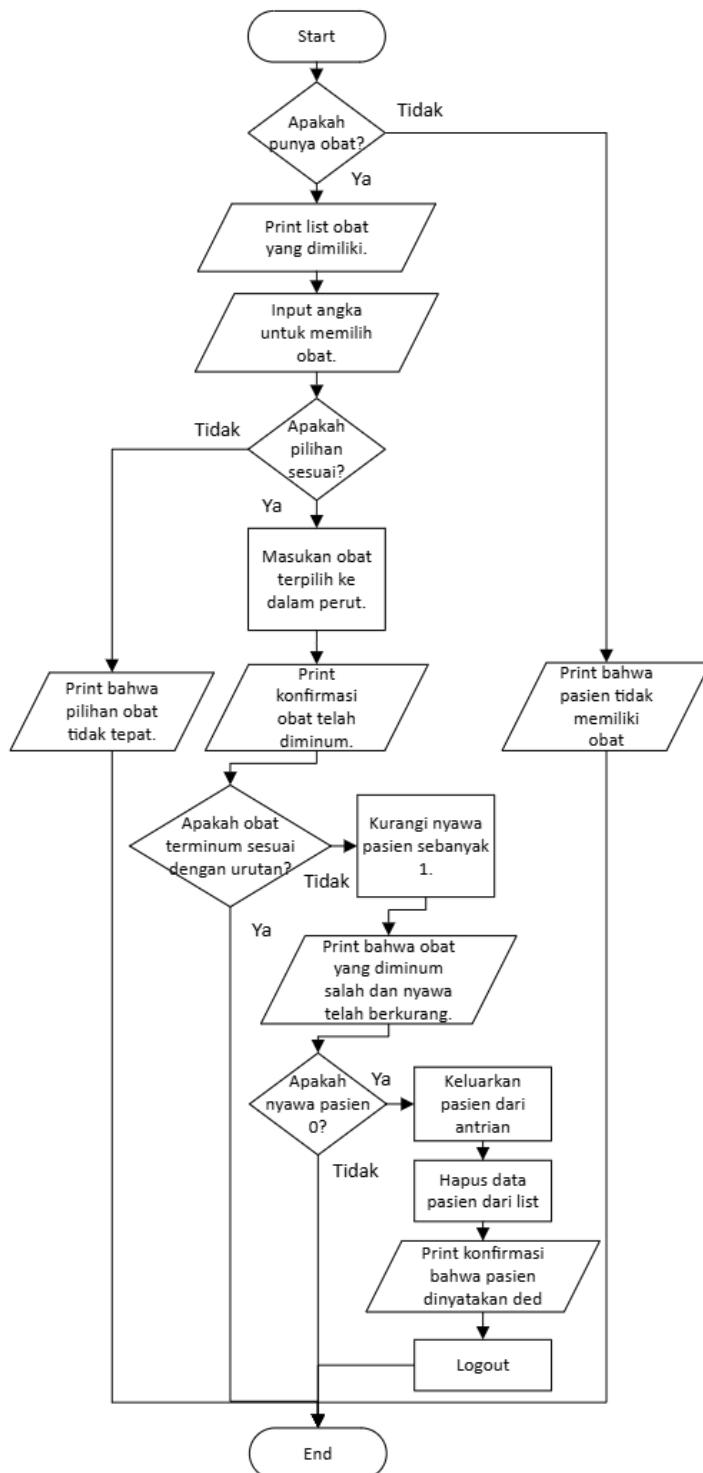
Gambar 7.14 Flowchart F14 - Daftar Check-Up

15. F15 - Antrian Saya!



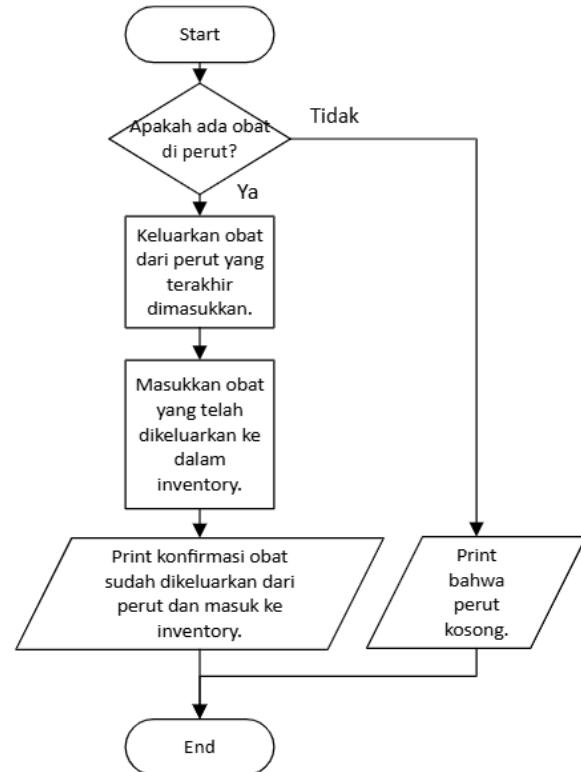
Gambar 7.15 Flowchart F15 - Antrian Saya!

### 16. F16 - Minum Obat + B05 - Dead or Alive



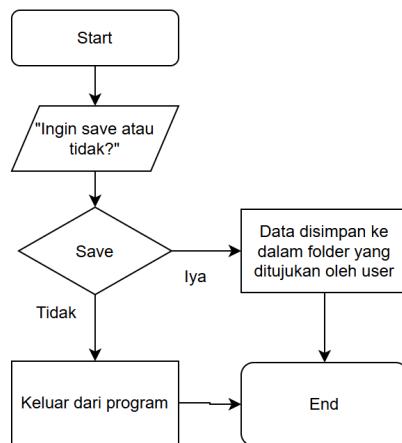
Gambar 7.16 Flowchart F16 - Minum Obat + B05 - Dead or Alive

### 17. F17 - Minum Penawar



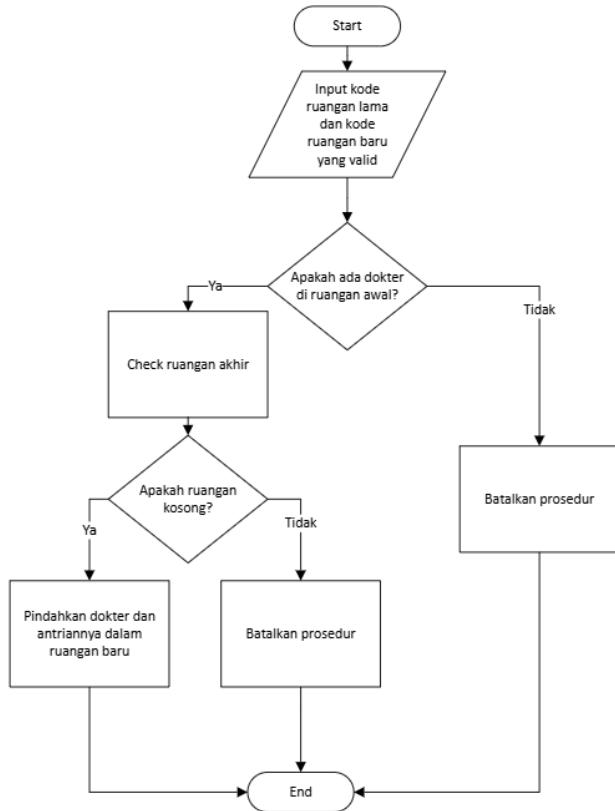
Gambar 7.17 Flowchart F17 - Minum Penawar

### 18. F18 - Exit

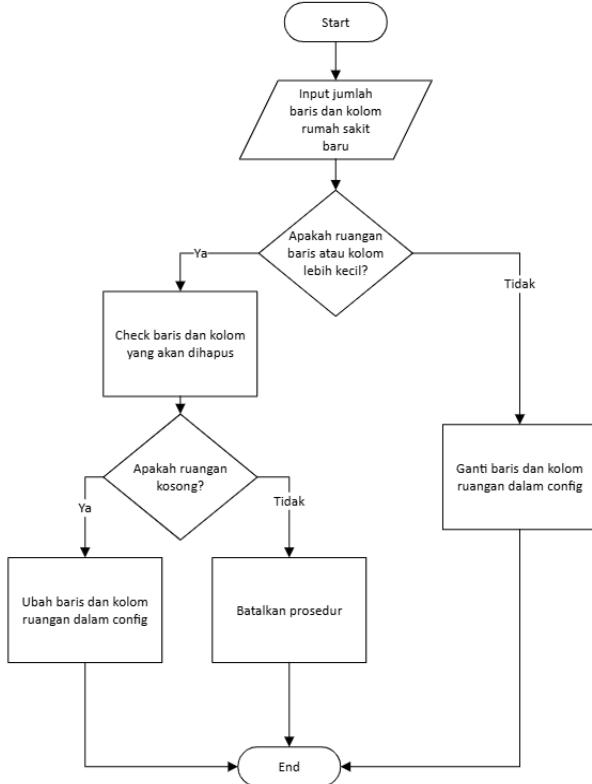


Gambar 7.18 Flowchart F18 - Exit

## 19. B02 - Denah Dinamis

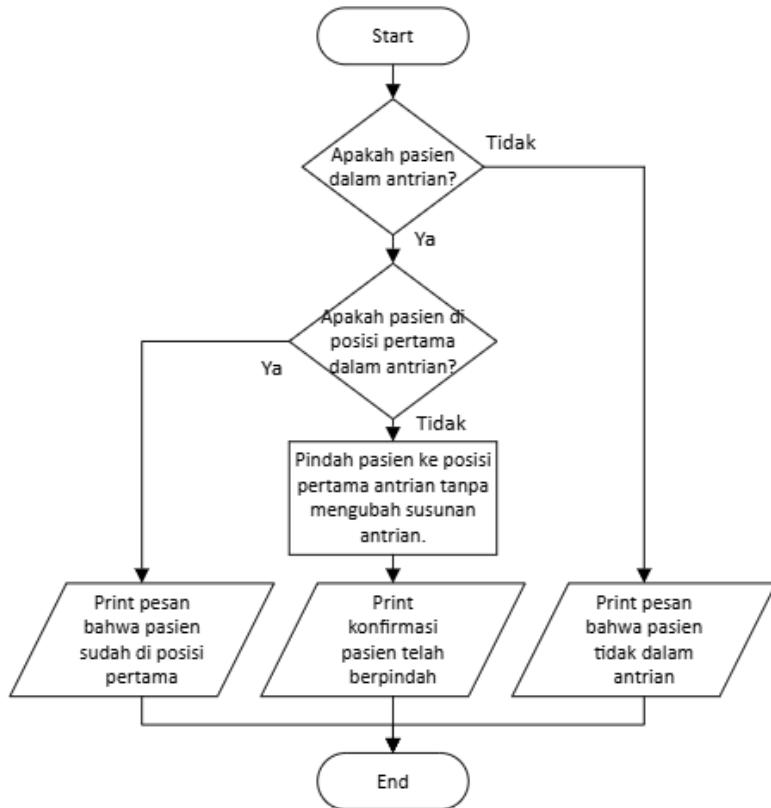


Gambar 7.19a Flowchart B02 - Pindah dokter

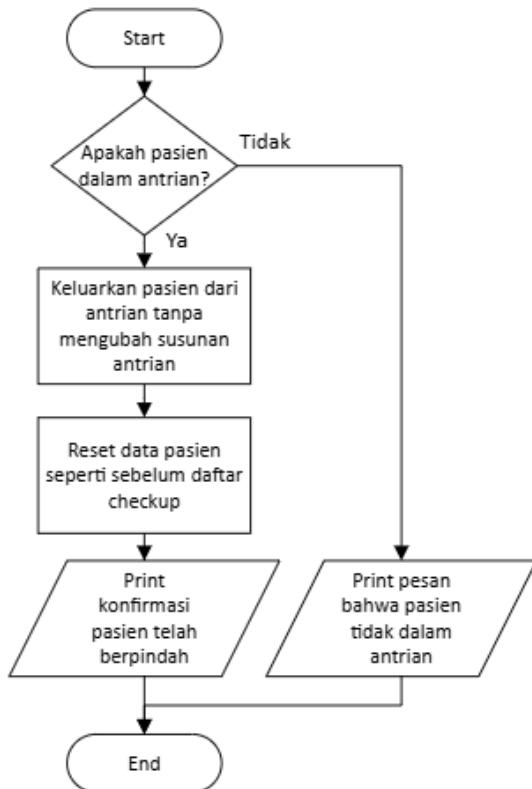


Gambar 7.19b Flowchart B02 - Ubah Denah

## 20. B06 - Mainin Antrian



Gambar 7.20a Flowchart B06 - Skip Antrian



Gambar 7.20b Flowchart B06 - Cancel Antrian

# Spesifikasi Tiap Modul/Prosedur/Fungsi

## 1. F01 - Login

```
procedure login(input/output loginState : integer,
                  input/output loginId : integer,
                  input UserData : ListDinUser);
{ I.S. Seluruh argumen terdefinisi dan user belum login sebelumnya
  F.S. User terlogin sebagai rolenya dan berdasarkan idnya }
```

### KAMUS LOKAL

```
Username, Password: string
idxUsername: integer
```

### ALGORITMA

```
{ Input username dengan validasi }
repeat
    output("Username (max 20 characters): ")
    input(Username)
    if (length(Username) > 20) then
        output("Error: Username melebihi 20 characters.")
    until (length(Username) <= 20)

{ Input password dengan validasi }
repeat
    output("Password (max 20 characters): ")
    input(Password)
    if (length>Password) > 20) then
        output("Error: Password melebihi 20 characters.")
    until (length>Password) <= 20

idxUsername ← indexOfUsername(UserData, Username)

if (idxUsername = -1) then
    output("Tidak ada user di database dengan username ",
    Username, "!")
    else if (UserData.buffer[idxUsername].password ≠ Password)
then
    output("Password yang dimasukkan salah!")
else
    loginId ← UserData.buffer[idxUsername].id

if (UserData.buffer[idxUsername].role = "manager") then
    output("Halo Manager ", Username)
    loginState ← 1
else if (UserData.buffer[idxUsername].role = "dokter")
then
    output("Halo Dokter ", Username)
    loginState ← 2
else if (UserData.buffer[idxUsername].role = "pasien")
then
```

```

    output("Halo ", Username, "! Ada keluhan apa?")
    loginState ← 3

```

## 2. F02 - Register

```

procedure registerPasien(input/output UserData: ListDinUser)
{ I.S. UserData terdefinisi, mungkin kosong }
{ F.S. Jika registrasi berhasil, pasien baru ditambahkan ke
UserData}

```

### KAMUS LOKAL

```

Username, Password: string
isValid: boolean
i: integer
setUsername: Set
tempUsername: string

```

### ALGORITMA

```

{ Input username dengan validasi }
repeat
    output("Username (max 20 characters): ")
    input(Username)

    isValid ← true
    i ← 0
    while (i < length(Username)) and (isValid) do
        if (ord(Username[i]) < 65) or (ord(Username[i]) >
122) or
            ((ord(Username[i]) > 90) and (ord(Username[i]) <
97)) then
                isValid ← false
                output("Username hanya boleh terdiri atas huruf
alfabet!")
        i ← i + 1

        if (length(Username) > 20) then
            output("Error: Username melebihi 20 characters.")
        until (length(Username) <= 20) and (isValid)

{ Input password dengan validasi }
repeat
    output("Password (max 20 characters): ")
    input(Password)
    if (length(Password) > 20) then
        output("Error: Password melebihi 20 characters.")
    until (length(Password) <= 20)

{ Cek username unik }
createSet(setUsername, 100)
i traversal [0..UserData.nEff-1]
setInsert(setUsername,
to_lower(UserData.buffer[i].username))

```

```

    if setContain(setUsername, to_lower(Username)) then
        output("Registrasi gagal! User dengan nama ", Username, " "
sudah terdaftar.")
    else
        output("Pasien ", Username, " berhasil ditambahkan!")

    { Inisialisasi user baru }
    UserData.buffer[UserData.nEff] ← createEmptyUser()
    UserData.buffer[UserData.nEff].id ←
    UserData.buffer[UserData.nEff-1].id + 1
    UserData.buffer[UserData.nEff].username ← Username
    UserData.buffer[UserData.nEff].password ← Password
    UserData.buffer[UserData.nEff].role ← "pasien"
    UserData.buffer[UserData.nEff].ruang ← ""
    UserData.buffer[UserData.nEff].antrian ← ""
    UserData.nEff ← UserData.nEff + 1

```

### 3. F03 - Logout

```

procedure logout(input/output loginState: integer,
                  input/output loginId: integer)
{ I.S. loginState dan loginId terdefinisi }
{ F.S. Jika user sudah login, status login direset. Jika belum
login, tampilkan pesan error }

```

#### **KAMUS LOKAL**

#### **ALGORITMA**

```

    if (loginState = 0) then
        output("Logout gagal!")
        output("Anda belum login, silahkan login terlebih dahulu
sebelum melakukan logout")
    else
        loginState ← 0
        loginId ← -1
        output("Sampai Jumpa")

```

### 4. F04 - Lupa Password

```

procedure lupaPassword(input/output UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Jika verifikasi berhasil, password user diupdate }

```

#### **KAMUS LOKAL**

```

    uname, KodeUnik: string
    isValid: boolean
    i, j, count, idxUsername, RLEidx: integer
    RLEKode: string
    currentChar: char
    NewPassword: string

```

#### **ALGORITMA**

```

    { Input username dengan validasi }
    repeat
        output("Username (max 20 characters): ")
        input(uname)

        isValid ← true

```

```

        i ← 0
        while (i < length(uname)) and (isValid) do
            if (ord(uname[i]) < 65) or (ord(uname[i]) > 122) or
                ((ord(uname[i]) > 90) and (ord(uname[i]) < 97))
        then
            isValid ← false
            output("Username hanya boleh terdiri atas huruf
alfabet!")
            i ← i + 1

            if (length(uname) > 20) then
                output("Error: Username melebihi 20 characters.")
            until (length(uname) <= 20) and (isValid)

            output("KodeUnik: ")
            input(KodeUnik)
            output() { newline }

        idxUsername ← indexOfUsername(UserData, uname)

        if (idxUsername = -1) then
            output("Tidak ada user di database dengan username ",
uname, "!")
        else
            { Generate RLE Kode }
            RLEidx ← 0
            i ← 0
            while (i < length(uname)) do
                currentChar ← uname[i]
                count ← 1

                while (i + 1 < length(uname)) and (uname[i + 1] =
currentChar) do
                    count ← count + 1
                    i ← i + 1

                if (count > 1) then
                    RLEKode[RLEidx..RLEidx+length(intToStr(count))-1]
← intToStr(count)
                    RLEidx ← RLEidx + length(intToStr(count))
                    RLEKode[RLEidx] ← currentChar
                    RLEidx ← RLEidx + 1
                    i ← i + 1

            RLEKode[RLEidx] ← '\0' { Terminate string }

            if (KodeUnik = RLEKode) then
                { Verifikasi berhasil }
                depend on(UserData.buffer[idxUsername].role)
                    "manager": output("Halo Manager ", uname)
                    "dokter": output("Halo Dokter ", uname)
                    "pasien": output("Halo ", uname, "! Ada keluhan
apa?")

```

```

    output("Password Baru: ")
    input(NewPassword)
    UserData.buffer[idxUsername].password ← NewPassword
    output() { newline }
else
    output("Kode unik salah!")

```

## 5. F05 - Help

```

procedure help (input loginState : integer, input loginId :
integer,
                  input UserData : ListDinUser)
{I.S. loginID valid}
{F.S. menampilkan seluruh command yang bisa dilakukan oleh user
sesuai dengan posisinya}

KAMUS LOKAL
ALGORITMA
    output("n===== HELP =====\n")

    depend on loginState
        loginState = 0:
            output("Kamu belum login sebagai role apapun.
Silahkan login terlebih dahulu.\n\n")
            output("1. LOGIN      : Masuk ke dalam akun yang sudah
terdaftar")
            output("2. REGISTER : Membuat akun baru")
            output("3. SAVE       : Menyimpan kondisi rumah sakit")
            output("4. EXIT       : Keluar dari program")

        loginState = 1:
            output("Halo Manager " +
UserData.buffer[loginId-1].username + ".
Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n\n")
            output(" 1. LOGOUT   : Keluar dari akun yang sedang
digunakan")
            output(" 2. TAMBAH_DOKTER: Mendaftarkan dokter baru
ke sistem")
            output(" 3. LIHAT_DENAH      : Membuka denah rumah
sakit")
            output(" 4. LIHAT_RUANGAN     : Melihat isi ruangan pada rumah
sakit")
            output(" 5. LIHAT_USER       : Melihat data seluruh
pengguna")
            output(" 6. LIHAT_PASIEN     : Melihat data seluruh
pasien")
            output(" 7. LIHAT_DOKTER     : Melihat data seluruh
dokter")
            output(" 8. CARI_USER       : Mencari pengguna
tertentu
berdasarkan ID atau Nama")
            output(" 9. CARI_PASIEN     : Mencari pasien
tertentu
berdasarkan ID, Nama, atau Penyakit")

```

```

        output("10. CARI_DOKTER      : Mencari dokter
tertentu
berdasarkan ID atau Nama")
        output("11. LIHAT_SEMUA_ANTRIAN: Melihat rincian
seluruh
ruangan saat ini")
        output("12. TAMBAH_DOKTER      : Menambah dokter
baru dengan
username dan password yang ditentukan")
        output("13. ASSIGN_DOKTER      : Melakukan assign
ruangan ke
dokter tertentu yang belum memiliki ruangan")
output("14. UBAH_DENAH      : Mengubah ukuran ruangan dari rumah
sakit")
        output("15. PINDAH_DOK      : Melakukan pemindahan
ruangan dokter ke ruangan lain yang kosong beserta dengan
antriannya")
        output("16. SAVE      : Menyimpan kondisi rumah
sakit")
output("17. EXIT      : Keluar dari program")

loginState = 2:
        output("Halo Dokter " +
UserData.buffer[loginId-1].username +
". Kamu memanggil HELP. Berikut command yang tersedia:\n\n")
        output("1. LOGOUT      : Keluar dari akun")
        output("2. DIAGNOSIS      : Melakukan diagnosis
pasien")
        output("3. NGOBATIN      : Meresepkan obat")
        output("4. LIHAT_DENAH      : Membuka denah rumah
sakit")
        output("5. LIHAT_RUANGAN      : Melihat isi ruangan pada
rumah
sakit")
output("6. SAVE      : Menyimpan kondisi rumah sakit")
output("7. EXIT      : Keluar dari program")

loginState = 3:
        output("Selamat datang, " +
UserData.buffer[loginId-1].username + ".
Berikut command yang tersedia:\n\n")
        output(" 1. LOGOUT      : Keluar dari akun")
        output(" 2. DAFTAR_CHECKUP      : Mendaftarkan diri
untuk
pemeriksaan")
        output(" 3. ANTRIAN      : Melihat status
antrian")
        output(" 4. LIHAT_DENAH      : Membuka denah rumah
sakit")
        output(" 5. LIHAT_RUANGAN      : Melihat isi ruangan pada
rumah
sakit")

```

```

        output(" 6. PULANGDOK      : Berkonsultasi kembali
ke
dokter untuk menanyakan apakah kondisi sudah baik
dan boleh pulang")
        output(" 7. MINUM_OBAT      : Melihat daftar obat
dan
memilih obat yang akan diminum")
        output(" 8. PENAWAR      : Meminum penawar jika
salah
minum obat dan mengeluarkan obat terakhir yang
diminum")
        output(" 9. SKIP_ANTRIAN      : Langsung maju ke posisi
terdepan")
output("10. CANCEL_ANTRIAN      : Keluar sepenuhnya dari
ruangan")
        output("11. SAVE      : Menyimpan kondisi rumah
sakit")
output("12. EXIT      : Keluar dari program")

output("`nFootnote:")
output("1. Untuk menggunakan aplikasi, silahkan masukkan nama
fungsi yang terdaftar")
output("2. Jangan lupa untuk memasukkan input yang valid`n")

```

## 6. F06 - Denah Rumah Sakit (Lihat Denah dan Lihat Ruangan)

```

function cariUsername (UserData : ListDinUser, idPasien
: integer) → string
{ I.S. idPasien valid }
{ F.S. Menghasilkan output berupa nama username }

```

### KAMUS

ruangan : array [A..Z] of char

```

procedure lihatDenah(input rumahsakit : Config)
{ I.S. rumahsakit terdefinisi }
{ F.S. Menghasilkan output berupa denah rumah sakit }

```

### KAMUS LOKAL

i, j : integer

### ALGORITMA

```

output("      ")
i traversal[0..rumahsakit.denah.cols - 1]
output(" " + (j+1) + " ")

output("`n      ")
j traversal[0..rumahsakit.denah.cols - 1]
output("----+")

output("`n")

```

```

    i traversal [0..rumahsakit.denah.rows - 1]
        output(" " + ruangan[i] + " |")
        j traversal [0..rumahsakit.denah.cols]
            output(" " + ruangan[i] + (j+1) + " |")

            output("    +")
        j traversal [0..rumahsakit.denah.cols]
            output("-----+")

procedure lihatRuangan(input rumahsakit : Config,
input Userdata : ListDinUser)
{ I.S. rumahsakit terdefinisi }
{ F.S. Menerima inputan kode ruangan yang valid dan
menampilkan isinya }

```

#### **KAMUS LOKAL**

```

kodeRuang : string
baris, kolom : integer
idDokter, pasienCount, id_pasien : integer
temp : pointer to Node

```

#### **ALGORITMA**

```

iterate
    output("Ruang: ")
    input(kodeRuang)

    stop(length(kodeRuang) = 2 OR kodeRuang[0] not
in ruangan OR kodeRuang[1] not in ['1'..'9'])

        output("Input tidak valid! Format harus 1 huruf
kapital
dan 1 angka (contoh : A1)")

    baris ← ASCII(kodeRuang[0]) - ASCII('A')
    kolom ← ASCII(kodeRuang[1]) - ASCII('1')

    if (baris < 0 OR baris ≥ rumahsakit.denah.rows OR
        kolom < 0 OR kolom ≥ rumahsakit.denah.cols)then
        output("Ruang tidak ditemukan!")

    else
        idDokter ←
        rumahsakit.denah.contents[baris][kolom].dokterID
        pasienCount ← 0
        temp
        ←rumahsakit.denah.contents[baris][kolom].antrian.front

```

```

        output("\\n--- Detail Ruangan " + kodeRuangan +
" ---")
        output("Kapasitas : " +
rumahsakit.kapasitasRuangan)
        if idDokter = 0 then
            output("Dokter      : -")
        else
            output("Dokter      : " +
cariUsername(UserData,
idDokter))

        output("Pasien di dalam ruangan:")

        while (temp ≠ NULL AND
pasienCount rumahsakit.kapasitasRuangan AND
cariUsername(UserData, idDokter) ≠ "-") do
            id_pasien ← temp↑.data
            if (id_pasien ≠ 0) then
                pasienCount ← pasienCount + 1

            output(" " + pasienCount + ". " +
cariUsername(UserData, id_pasien))
            temp↑ ← temp↑.next

            if (pasienCount = 0) then
                output(" Tidak ada pasien")

        output("-----")

```

## 7. F07 - Lihat User

```

{ Menanyakan user sort berdasarkan id atau nama dan urutan sort }
procedure sortBased(input/output base: integer,
                     input/output order: integer)
{ I.S. base dan order masih 0 }
{ F.S. base dan order memiliki nilai berdasarkan jenis sort yang
diinginkan }

{ Melakukan sort berdasarkan nama/alfabet }
procedure alphabetSort(input/output ptrs: array [0..99] of
^User,
                     input length: integer)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Pointers-pointers ptrs disortir berdasarkan alfabet }

{ Melakukan sort berdasarkan aura }
procedure auraSort(input/output ptrs: array [0..99] of ^User,
                     input length: integer)

{ I.S. Seluruh argumen terdefinisi }

```

```

{ F.S. Pointers-pointers ptrs disortir berdasarkan aura }

{ Menampilkan list berdasarkan jenis pengurutan }
procedure tampilList(input ptrs: array [0..99] of ^User,
                     input type: integer,
                     input base: integer,
                     input order: integer,
                     input length: integer)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Ditampilkan list berdasarkan jenis pengurutan yang
diberikan user }

procedure lihatUser(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan semua user dengan sorting tertentu }

```

**KAMUS LOKAL**

```

base, order, length, i: integer
ptrs: array[0..99] of ^User

```

**ALGORITMA**

```

sortBased(base, order)
length ← listLength(UserData)

i traversal [0..length-1]
ptrs[i] ← @UserData.buffer[i]

tampilList(ptrs, 1, base, order, length)

```

```

procedure lihatPasien(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan pasien saja dengan sorting tertentu }

```

**KAMUS LOKAL**

```

base, order, length, i: integer
ptrs: array[0..99] of ^User

```

**ALGORITMA**

```

sortBased(base, order)
length ← listLength(UserData)

i traversal [0..length-1]
ptrs[i] ← @UserData.buffer[i]

tampilList(ptrs, 2, base, order, length)

```

```

procedure lihatDokter(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan dokter saja dengan sorting tertentu }

```

**KAMUS LOKAL**

```

base, order, length, i: integer
ptrs: array[0..99] of ^User

ALGORITMA
sortBased(base, order)
length ← listLength(UserData)

i traversal [0..length-1]
ptrs[i] ← @UserData.buffer[i]

tampillList(ptrs, 3, base, order, length)

```

## 8. F08 - Cari User

```

{ Menanyakan user pencarian berdasarkan id atau nama atau
penyakit (bagi pasien) }
procedure findBased(input/output base: integer,
                     input type: integer)
{ I.S. base adalah 0 dan type terdefinisi }
{ F.S. base memiliki nilai berdasarkan jenis pencarian yang
diinginkan }

{ Melakukan binary search untuk mencari index dengan id tertentu
}
function binSearchId(input UserData: ListDinUser,
                      input id: integer): integer
{ Mengembalikan index user dengan id tertentu atau -1 jika tidak
ditemukan }

{ Melakukan sequential search untuk mencari index dengan username
tertentu }
function seqSearchName(UserData: ListDinUser,
                        name: string) → integer
{ Mengembalikan index user dengan username tertentu atau -1 jika
tidak ditemukan }

{ Menampilkan data dari pencarian id }
procedure findId(input UserData: ListDinUser,
                  input type: integer)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Ditampilkan data dari pencarian id }

{ Menampilkan data dari pencarian nama }
procedure findName(input UserData: ListDinUser,
                   input type: integer)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Ditampilkan data dari pencarian username }

{ Menampilkan list data dari pencarian penyakit }

```

```

procedure tampilPenyakit(input ptrs: array [0..99] of ^User,
                        input base: integer,
                        input order: integer,
                        input length: integer,
                        input penyakit: string)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Ditampilkan list data dari pencarian penyakit }

{ Menampilkan data dari pencarian penyakit }
procedure findPenyakit(input UserData: ListDinUser)
{ I.S. Seluruh argumen terdefinisi }
{ F.S. Ditampilkan data dari pencarian penyakit }

procedure cariUser(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan hasil pencarian user berdasarkan kriteria }

```

**KAMUS LOKAL**

base: integer

**ALGORITMA**

```

base ← 0
findBased(base, 1)

depend on(base)
    1: findId(UserData, 1)
    2: findName(UserData, 1)

```

```

procedure cariPasien(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan hasil pencarian pasien berdasarkan kriteria }

```

**KAMUS LOKAL**

base: integer

**ALGORITMA**

```

base ← 0
findBased(base, 2)

depend on (base)
    1: findId(UserData, 2)
    2: findName(UserData, 2)
    3: findPenyakit(UserData)

```

```

procedure cariDokter(input UserData: ListDinUser)
{ I.S. UserData terdefinisi }
{ F.S. Menampilkan hasil pencarian dokter berdasarkan kriteria }

```

**KAMUS LOKAL**

base: integer

**ALGORITMA**

```
base ← 0
findBased(base, 3)

depend on(base)
1: findId(UserData, 3)
2: findName(UserData, 3)
```

## 9. F09 - Lihat Antrian

```
function cari_user (input UserData: ListDinUser, input ID:
integer) → string
{ Mencari username berdasarkan ID user }
{ I.S. UserData dan ID terdefinisi }
{ F.S. Mengembalikan username jika ditemukan, "-" jika tidak }
```

```
procedure lihatAntrian (input rumahsakit: Config, input
UserData: ListDinUser)
{ Menampilkan denah ruangan dan antrian pasien untuk setiap
dokter }
{ I.S. rumahsakit dan UserData terdefinisi }
{ F.S. Menampilkan informasi antrian ke layar }
```

**KAMUS LOKAL**

```
ruang: array [0..25] of characters ['A', 'B', 'C', ..., 'Z']
baris, kolom, j, i: integer
pasien_count, nomor_urut, antrian_count, nomor: integer
temp: Node
nama: string
```

**ALGORITMA**

```
output(" ")
j traversal [0.. rumahsakit.denah.cols-1]
    output(" ", j+1, " ")
output("\\n")

output(" +")
j traversal [0..rumahsakit.denah.cols-1]
    output("----+")
output("\\n")

i traversal [0..rumahsakit.denah.rows-1]
    output(ruang[i], " |")
    j traversal [0..rumahsakit.denah.cols-1]
        output(" ", ruang[i], j+1, " |")
        output("\\n")
        output(" +")
        j traversal [0..rumahsakit.denah.cols-1]
            output("----+")
            output("\\n")

baris traversal [0..rumahsakit.denah.rows-1]
```

```

        kolom traversal [0..rumahsakit.denah.cols-1]
        if
        (rumahsakit.denah.contents[baris][kolom].dokterID = 0) then
            continue
            output("\\n===== ",
            rumahsakit.denah.contents[baris][kolom].kodeRuangan,
            " =====")
            output("Kapasitas : ",
            rumahsakit.kapasitasRuangan)
            output("Dokter : ",
            cari_user(UserData,
            rumahsakit.denah.contents[baris][kolom].dokterID))
            output("Pasien di dalam ruangan:")
            pasien_count ← 0
            nomor_urut ← 1
            temp ←
            rumahsakit.denah.contents[baris][kolom].antrian.front
            while (temp ≠ NULL) and (pasien_count <
            rumahsakit.kapasitasRuangan) do
                nama ← cari_user(UserData, temp.data)
                if (nama ≠ "-") and (nama ≠ "") then
                    output(" ", nomor_urut, ". ", nama)
                    nomor_urut ← nomor_urut + 1
                    pasien_count ← pasien_count + 1
                    temp ← temp.next
                if (pasien_count = 0) then
                    output(" Tidak ada pasien")
                output("Pasien di antrian:")
                nomor ← 1
                antrian_count ← 0
                while (temp ≠ NULL) and (antrian_count <
                rumahsakit.kapasitasAntrian) do
                    nama ← cari_user(UserData,
                    temp.data)
                    if (nama ≠ "-") and (nama ≠ "")
                    then
                        output(" ", nomor, ". ", nama)
                        nomor ← nomor + 1
                        antrian_count ← antrian_count + 1
                        temp ← temp.next
                    if (antrian_count = 0) then
                        output(" Tidak ada pasien di antrian
                        luar ruangan")
                        output("-----")

```

#### 10. F10 - Tambah Dokter (Tambah Dokter dan Assign Dokter)

```

procedure tambahDokter(input/output UserData: ListDinUser)
{ I.S. UserData terdefinisi, mungkin kosong }

```

```
{ F.S. Jika registrasi berhasil, dokter baru ditambahkan ke  
UserData}
```

#### KAMUS LOKAL

```
Username, Password: string  
isValid: boolean  
i: integer  
setUsername: Set
```

#### ALGORITMA

```
{ Input username dengan validasi }  
repeat  
    output("Username (max 20 characters): ")  
    input(Username)  
  
    isValid ← true  
    i traversal [0..length(Username) - 1]  
        if (ord(Username[i]) < 65) or (ord(Username[i]) >  
122) or  
            ((ord(Username[i]) > 90) and (ord(Username[i]) <  
97)) then  
                isValid ← false  
                output("Username hanya boleh terdiri atas huruf  
alfabet!")  
                break  
  
        if (length(Username) > 20) then  
            output("Error: Username melebihi 20 characters.")  
until (length(Username) ≤ 20) and (isValid)  
  
{ Input password dengan validasi }  
repeat  
    output("Password (max 20 characters): ")  
    input(Password)  
    if (length(Password) > 20) then  
        output("Error: Password melebihi 20 characters.")  
until (length(Password) ≤ 20)  
  
{ Cek username unik }  
createSet(setUsername, 100)  
i traversal [0..UserData.nEff - 1]  
    setInsert(setUsername,  
to_lower(UserData.buffer[i].username))  
  
if setContain(setUsername, to_lower(Username)) then  
    output("Penambahan dokter gagal! Sudah ada user bernama  
, Username)  
else  
    output("Dokter ", Username, " berhasil ditambahkan!")  
  
{ Inisialisasi dokter baru }  
UserData.buffer[UserData.nEff].id ←  
UserData.buffer[UserData.nEff - 1].id + 1
```

```

        UserData.buffer[UserData.nEff].username ← Username
        UserData.buffer[UserData.nEff].password ← Password
        UserData.buffer[UserData.nEff].role ← "dokter"
        UserData.buffer[UserData.nEff].ruang ← ""
        UserData.buffer[UserData.nEff].antrian ← ""
        UserData.nEff ← UserData.nEff + 1

procedure assignDokter(input/output UserData: ListDinUser,
input/output rumahsakit: Config)
{ I.S. UserData dan denah rumahsakit terdefinisi }
{ F.S. Dokter diassign ke ruangan tertentu jika valid }

```

#### KAMUS LOKAL

```

    Username: string
    input: string
    baris, kolom, idx, idxOldDok: integer

```

#### ALGORITMA

```

{ Input username dokter }
repeat
    output("Username (max 20 characters): ")
    input(Username)
    if (length(Username) > 20) then
        output("Error: Username melebihi 20 characters.")
    until (length(Username) ≤ 20)

{ Input ruangan dengan validasi format (contoh: A1) }
repeat
    output("Ruang: ")
    input(input)
    input ← hapusNewline(input) { Menghapus karakter '\n' }

    if (length(input) ≠ 2) or
        (ord(input[0]) < 65 or ord(input[0]) > 90) or
        (ord(input[1]) < 49 or ord(input[1]) > 57) then
            output("Input tidak valid! Format harus 1 huruf
kapital dan 1 angka (contoh: A1).")
    until (input memenuhi format)

{ Konversi input ruangan ke indeks matriks }
baris ← ord(input[0]) - ord('A')
kolom ← int(input[1]) - 1

{ Cek keberadaan dokter }
idx ← -1
i traversal [0..UserData.nEff - 1]
if (UserData.buffer[i].username = Username) then
    idx ← i
    break

if (idx = -1) then
    output("Tidak ada dokter bernama ", Username)
    return

```

```

    { Validasi ruangan }
    if (baris < 0 or kolom < 0 or
        kolom ≥ rumahsakit.denah.cols or baris ≥
        rumahsakit.denah.rows) then
        output("Ruangan tidak ditemukan!")
        return

    { Cek konflik: dokter sudah punya ruangan atau ruangan sudah
    terisi }
    if (length(UserData.buffer[idx].ruang) > 0) then
        output("Dokter ", Username, " sudah menempati ruangan ",
        UserData.buffer[idx].ruang)

    if (rumahsakit.denah.contents[baris][kolom].dokterID ≠ 0)
then
    idxOldDok ← cariIdxUser(UserData,
    rumahsakit.denah.contents[baris][kolom].dokterID)
    output("Ruangan ", input, " sudah ditempati oleh Dokter
    ", UserData.buffer[idxOldDok].username)

    { Assign dokter ke ruangan jika semua kondisi terpenuhi }
    if (length(UserData.buffer[idx].ruang) = 0) and
        (rumahsakit.denah.contents[baris][kolom].dokterID = 0)
then
    output("Dokter ", Username, " berhasil diassign ke
    ruangan ", input)
    UserData.buffer[idx].ruang ← input
    rumahsakit.denah.contents[baris][kolom].dokterID ←
    UserData.buffer[idx].id
    rumahsakit.denah.contents[baris][kolom].antrian.front ← 0

```

## 11. F11 - Diagnosis

```

function cekPenyakit (kriteriaPenyakit : ListPenyakit, UserData
: ListDinUser, idPasien : integer) → integer
{I.S. idPasien merupakan id yang valid}
{F.S. Mengeluarkan output integer(sebagai boolean)}

function assignPenyakit(kriteriaPenyakit : ListPenyakit,
    UserData : ListDinUser, idPasien : integer) → boolean
{I.S. idPasien merupakan id yang valid}
{F.S. Mengganti nilai dari UserData jika pasien memenuhi syarat
penyakit dan melakukan return true jika berhasil dan false jika
gagal}

function punyaRiwayat(riwayatPenyakit : string) → boolean
{I.S. -}
{F.S. Melakukan pengecekan apakah pasien punya riwayat penyakit}

void diagnosisPenyakit(input kriteriaPenyakit : ListPenyakit,
    input rumahsakit : Config, input/output
    UserData : ListDinUser, input loginId : integer) ->
    integer

```

```

{I.S. -
{F.S. Mengeluarkan diagnosis dari pasien dan mengganti data
riwayat penyakit pasien paling depan dari ruangan yang dijalan
oleh dokter dengan id tertentu}

```

**KAMUS LOKAL**

```

idxKolom, idxBaris : integer
dokterFoundruangan : integer ← 0
idxDokter, idxPasien, idPenyakitSekarang : integer
found : boolean ← false
i, j : integer ← 0

```

**ALGORITMA**

```

while (not found AND i < rumahsakit.denah.rows) do
    j ← 0
    while (not found AND j < rumahsakit.denah.cols) do
        if(rumahsakit.denah.contents[i][j].dokterID = loginId)
then
        idxBaris ← i
        idxKolom ← j
        found ← true
        dokterFoundruangan ← 1

        j ← j + 1
        i ← i + 1

    idxDokter ← cariIdxUser(UserData, loginId)
    if (idxDokter ≠ -1 AND dokterFoundruangan = 1) then
        idxPasien ←
    cariIdxUser(UserData, rumahsakit.denah.contents[idxBaris][idxKolom]
    ].antrian.front.data)

    if (idxPasien ≠ -1) then
        idPenyakitSekarang ← cekPenyakit(kriteriaPenyakit,
    UserData, idxPasien)

        if(punyaRiwayat(UserData.buffer[idxPasien].riwayat_p
    enyakit))then
            if (idPenyakitSekarang = -1) then
                output(UserData.buffer[idxPasien].username + "
                    tidak terdiagnosa penyakit apapun!")
                UserData.buffer[idxPasien].riwayat_penyakit ←
            "_

```

"

```

        else
            output(UserData.buffer[idxPasien].username + "
                masih menderita "
                +
                UserData.buffer[idxPasien].riwayat_penyakit)

    else
        if assignPenyakit(kriteriaPenyakit,
            UserData, idxPasien) then

```

```

        printAsciiDiagnosis()
        output(UserData.buffer[idxPasien].username +
            " terdiagnosa penyakit " +
            UserData.buffer[idxPasien].riwayat_penyakit + "!")
    else
        output(UserData.buffer[idxPasien].username +
            " tidak terdiagnosa penyakit apapun!")
        UserData.buffer[idxPasien].riwayat_penyakit ←
            "--"

else
    output("Tidak ada pasien di ruangan Anda saat ini")

else
    output("Anda tidak sedang bertugas di ruangan manapun")

```

## 12. F12 - Ngobatin

```

procedure Ngobatin(input loginID: integer,
                    input/output config: Config,
                    input listUser: ListDinUser,
                    input listObat: ListObat,
                    input listPenyakit: ListPenyakit,
                    input mapObatPenyakit: MapObatPenyakit)

{ I.S. Semua parameter terdefinisi sesuai tipe }
{ F.S. Jika dokter ter-assign ke ruangan dan terdapat pasien,
      maka pasien diobati dan diberi daftar obat sesuai
      penyakit }

```

### KAMUS LOKAL

```

found : boolean
indeksRuang : array[0..1] of integer
i, j : integer
pasienID, pasienIdx : integer
penyakitID, penyakitIdx : integer
obatID, obatIdx : integer

```

### ALGORITMA

```

found ← false

i ← 0
while (i < config.denah.rows) and (not found) do
    j ← 0
    while (j < config.denah.cols) and (not found) do
        if config.denah.contents[i][j].dokterID = loginID
    then
        found ← true
        indeksRuang[0] ← i

```

```

        indeksRuangan[1] ← j
        j ← j + 1
        { j = config.denah.cols or ditemukan }
        i ← i + 1
        { i = config.denah.rows or ditemukan }

    if (not found) then
        output("Anda belum terassign pada ruangan apapun.")
        return

    if
(config.denah.contents[indeksRuangan[0]][indeksRuangan[1]].antria
.n.counter = 0) then
    output("Tidak ada pasien dalam ruangan.")
    return

    pasienID ←
config.denah.contents[indeksRuangan[0]][indeksRuangan[1]].antrian
.front.data
    pasienIdx ← cariIdxUser(listUser, pasienID)

    output("Dokter sedang mengobati pasien!")

    penyakitID ← searchPenyakitIDByName(listPenyakit,
listUser.buffer[pasienIdx].riwayat_penyakit)

    if (penyakitID = -1) then
        output("Pasien tidak memiliki penyakit!")
        output("Pasien belum didiagnosis!")
        return

    penyakitIdx ← cariIdxPenyakit(listPenyakit, penyakitID)
    printAsciiNgobatin()
    output("Pasien memiliki penyakit ",
listPenyakit.items[penyakitIdx].nama_penyakit)
    output("Obat yang harus diberikan:")

    i ← 1
    while (i < MAX_OBAT_PER_PENYAKIT) do
        obatID ←
mapObatPenyakit.items[penyakitID].value[i].obat_id
        if (obatID = 0) then
            i ← MAX_OBAT_PER_PENYAKIT { berhenti eksplisit }
        else
            obatIdx ← cariIdxObat(listObat, obatID)
            output(i, ". ", listObat.items[obatIdx].nama_obat)
            config.inventoryPasien.contents[pasienID][i] ←
listObat.items[obatIdx].obat_id
            i ← i + 1
        { i = MAX_OBAT_PER_PENYAKIT atau obatID = 0 }

    return

```

### 13. F13 - Aku boleh pulang ga, dok 😊 ?

```
function cekPerutDenganUrutan(perut : Stack, penyakitID : integer, mapObatPenyakit : MapObatPenyakit) -> integer
{ Mengecek apakah obat yang diminum pasien sesuai dengan urutannya. Mengembalikan 0 jika sesuai, i jika tidak (i > 0) dengan i adalah urutan pertama yang salah. }

procedure pulangDok(input loginID : integer,
                     input/output config : Config,
                     input/output listUser : ListDinUser,
                     input listObat : ListObat,
                     input listPenyakit : ListPenyakit,
                     input mapObatPenyakit : MapObatPenyakit)

{ I.S.: loginID adalah ID pasien yang sedang login }
{ F.S.: Jika valid, pasien dikeluarkan dari antrian dan data pasien di-reset;
      Jika tidak valid, akan muncul penjelasan kenapa tidak bisa pulang }
```

#### KAMUS LOKAL

```
userIdx, penyakitId, penyakitIdx : integer
banyakObat, i : integer
isiPerut : array [1..MAX_OBAT_PER_PENYAKIT] of integer
temp : pointer to Node
urutanSalah : integer
posisiPasien : array [0..1] of integer
obatId, obatIdx : integer
namaObat : string
```

#### ALGORITMA

```
userIdx ← cariIdxUser(listUser, loginID)
penyakitId ← searchPenyakitIDByName(listPenyakit,
listUser.buffer[userIdx].riwayat_penyakit)

if listUser.buffer[userIdx].suhu_tubuh = 0 then
    output("Kamu belum checkup!")
    return

else if listUser.buffer[userIdx].riwayat_penyakit = "" then
    output("Kamu belum menerima diagnosis apapun dari dokter,
jangan buru-buru pulang!")
    return

else if isMatriksRowEmpty(config.inventoryPasien, loginID)
and (config.perutPasien[loginID].head = NIL) then
    output("Kamu belum menerima obat apapun dari dokter!")
    return

else if not isMatriksRowEmpty(config.inventoryPasien,
loginID) then
```

```

    output("Masih ada obat yang belum kamu habiskan, minum
semuanya dulu yukk!")
    return

    output("Dokter sedang memeriksa keadaanmu...")
    urutanSalah ←
cekPerutDenganUrutan(config.perutPasien[loginID], penyakitId,
mapObatPenyakit)

    if isMatriksRowEmpty(config.inventoryPasien, loginID) and
(urutanSalah ≠ 0) then
        output("Maaf, tapi kamu masih belum bisa pulang!")
        banyakObat ← config.perutPasien[loginID].length
        temp ← config.perutPasien[loginID].head
        i ← banyakObat
        while temp ≠ NIL and i > 0 do
            isiPerut[i] ← temp.data
            i ← i - 1
            temp ← temp.next
        { i = 0 atau temp = NIL }

        output("Urutan peminuman obat yang diharapkan:")
        i ← 1
        while i ≤ banyakObat do
            obatId ←
mapObatPenyakit.items[penyakitId].value[i].obat_id
            obatIdx ← cariIdxObat(listObat, obatId)
            namaObat ← listObat.items[obatIdx].nama_obat
            if i = 1 then
                output(namaObat)
            else
                output(" -> ", namaObat)
            i ← i + 1
        { i = banyakObat + 1 }

        output("Urutan obat yang kamu minum:")
        i ← 1
        while i ≤ banyakObat do
            obatId ← isiPerut[i]
            obatIdx ← cariIdxObat(listObat, obatId)
            namaObat ← listObat.items[obatIdx].nama_obat
            if i < urutanSalah then
                if i = 1 then output(namaObat)
                else output(" -> ", namaObat)
            else
                if i = 1 then output("****", namaObat, "****") {
penanda salah }
                else output(" -> ", "****", namaObat, "****")
            i ← i + 1
        { i = banyakObat + 1 }

        output("Silahkan kunjungi dokter untuk meminta penawar
yang sesuai!")

```

```

return

posisiPasien[0] ← -1
posisiPasien[1] ← -1

i ← 0
while i < config.denah.rows do
    j ← 0
    while j < config.denah.cols do
        if config.denah.contents[i][j].antrian.front.data =
loginID then
            posisiPasien[0] ← i
            posisiPasien[1] ← j
            j ← config.denah.cols { break inner loop }
            j ← j + 1
        { j = config.denah.cols }
        if posisiPasien[0] = i then
            i ← config.denah.rows { break outer loop }
        else
            i ← i + 1
    { i = config.denah.rows }

dequeue(config.denah.contents[posisiPasien[0]][posisiPasien[1]].a
ntrian)
resetUserData(listUser.buffer[userIdx])
banyakObat ← config.perutPasien[loginID].length
i ← 0
while i < banyakObat do
    pop(config.perutPasien[loginID])
    i ← i + 1
{ i = banyakObat }

output("Selamat! Kamu sudah dinyatakan sembuh oleh dokter.
Silahkan pulang dan semoga sehat selalu!")
idxDokter ← cariIdxUser(listUser,
config.denah.contents[posisiPasien[0]][posisiPasien[1]].dokterID)
listUser.buffer[idxDokter].aura ←
listUser.buffer[idxDokter].aura + 1

```

#### 14. F14 - Daftar Checkup

```

function cari_user (input UserData: ListDinUser, input ID:
integer) → string
{ Mencari username berdasarkan ID user }
{ I.S. UserData dan ID terdefinisi }
{ F.S. Mengembalikan username jika ditemukan, "-" jika tidak }

procedure daftarCheckUp (input/output UserData: ListDinUser,
input/output rumahsakit: Config,
input loginId: integer)
{ Mendaftarkan user untuk checkup dengan dokter yang tersedia }
{ I.S. UserData, rumahsakit, dan loginId terdefinisi }

```

```

{ F.S. User terdaftar dalam antrian check-up jika memenuhi syarat
}

KAMUS LOKAL
    i, j, baris, kolom, nomor, choice: integer
    current: Node
    suhu, berat, saturasi: real
    sistolik, diastolik, detak, gula, tinggi, kolestrol,
    trombosit: integer
    valid: boolean
    choice_holder: array [1..50, 0..1] of integer
    antrian_total, current_antrian_luar: integer
    nama, kodeRuang: string

ALGORITMA
    i traversal [0..rumahsakit.denah.rows-1]
        j traversal [0..rumahsakit.denah.cols-1]
            current ←
            rumahsakit.denah.contents[i][j].antrian.front
                while (current ≠ NULL) do
                    if (current.data = loginId) then
                        output("Anda sudah terdaftar
dalam antrian check-up!")
                        output("Silakan selesaikan
check-up yang sudah terdaftar terlebih dahulu.")
                        current ← current.next

output("Silakan masukkan data check-up Anda:")

repeat
    output("Suhu Tubuh (Celcius): ")
        input(suhu)
        valid ← (suhu > 0)
        if not valid then
            output("Suhu tubuh harus berupa angka positif!")
until valid
UserData.buffer[loginId].suhu_tubuh ← suhu

repeat
    output("Tekanan Darah (sistol diastol, contoh: 120 80): ")
    input(sistolik, diastolik)
    valid ← (sistolik > 0) AND (diastolik > 0)
    if not valid then
        output("Tekanan darah harus berupa dua angka
positif!")
until valid
UserData.buffer[loginId].tekanan_darah_sistolik ← sistolik
UserData.buffer[loginId].tekanan_darah_diastolik ← diastolik

repeat
    output("Detak Jantung (bpm): ")
    input(detak)
    valid ← (detak > 0)
    if not valid then
        output("Detak jantung harus berupa angka positif!")

```

```

until valid
UserData.buffer[loginId].detak_jantung ← detak

repeat
    output("Saturasi Oksigen (%): ")
    input(saturasi)
    valid ← (saturasi > 0)
    if not valid then
        output("Saturasi oksigen harus berupa angka
positif!")
until valid
UserData.buffer[loginId].saturasi_oksigen ← saturasi

repeat
    output("Kadar Gula Darah (mg/dL): ")
    input(gula)
    valid ← (gula > 0)
    if not valid then
        output("Kadar gula darah harus berupa angka
positif!")
until valid
UserData.buffer[loginId].kadar_gula_darah ← gula

repeat
    output("Berat Badan (kg): ")
    input(berat)
    valid ← (berat > 0)
    if not valid then
        output("Berat badan harus berupa angka positif!")
until valid
UserData.buffer[loginId].berat_badan ← berat

repeat
    output("Tinggi Badan (cm): ")
    input(tinggi)
    valid ← (tinggi > 0)
    if not valid then
        output("Tinggi badan harus berupa angka positif!")
until valid
UserData.buffer[loginId].tinggi_badan ← tinggi

repeat
    output("Kadar Kolesterol (mg/dL): ")
    input(kolesterol)
    valid ← (kolesterol > 0)
    if not valid then
        output("Kadar kolesterol harus berupa angka
positif!")
until valid
UserData.buffer[loginId].kadar_kolesterol ← kolesterol

repeat
    output("Trombosit (ribu/µL): ")

```

```

    input(trombosit)
    valid ← (trombosit > 0)
    if not valid then
        output("Trombosit harus berupa angka positif!")
until valid
UserData.buffer[loginId].trombosit ← trombosit

nomor ← 1
output("Berikut adalah daftar dokter yang tersedia:")
baris traversal [0..rumahsakit.denah.rows-1]
    kolom traversal [0..rumahsakit.denah.cols-1]
        if (rumahsakit.denah.contents[baris][kolom].dokterID
= 0) then
            continue
        antrian_total ←
            rumahsakit.denah.contents[baris][kolom].antrian.coun
            ter
        current_antrian_luar ← antrian_total -
            rumahsakit.kapasitasRuangan
        if (current_antrian_luar = 4) then
            continue
        nama ← cari_user(UserData,
            rumahsakit.denah.contents[baris][kolom].dokterID)
        kodeRuang ←
            rumahsakit.denah.contents[baris][kolom].kodeRuang
        if (current_antrian_luar > 0) then
            output(nomor, ". Dr. ", nama, " - Spesialisasi
            Umum - Ruangan ", kodeRuang, " (Antrian: ",
            current_antrian_luar, ")")
        else
            output(nomor, ". Dr. ", nama, " - Spesialisasi
            Umum - Ruangan ", kodeRuang, " (Antrian: 0)")
        choice_holder[nomor][0] ← baris
        choice_holder[nomor][1] ← kolom
        nomor ← nomor + 1
    output("Pilih dokter (contoh input: 1) : ")
    input(choice)
    enqueue(rumahsakit.denah.contents[choice_holder[choice][0]][choic
    e_holder[choice][1]].antrian, loginId)
    printAsciiDaftarCheckup()
    output("Pendaftaran check-up berhasil!")
    output("Anda terdaftar pada antrian ", cari_user(UserData,
        rumahsakit.denah.contents[choice_holder[choice][0]][choice_holder
        [choice][1]].dokterID), " di ruangan ",
        rumahsakit.denah.contents[choice_holder[choice][0]][choice_holder
        [choice][1]].kodeRuang, ".")
    current_antrian_luar ←
        rumahsakit.denah.contents[choice_holder[choice][0]][choice_holder
        [choice][1]].antrian.counter - rumahsakit.kapasitasRuangan
    if (current_antrian_luar > 0) then
        output("Posisi antrian anda: ", current_antrian_luar)
    else

```

```

        output("Karena tidak ada antrian di luar ruangan, silahkan
langsung masuk saja.")
if
    (rumahsakit.denah.contents[choice_holder[choice][0]][choice_holder
    [choice][1]].antrian.counter = 1) then
        output("Dokter akan memeriksamu sekarang.")
else
        output("Masih ada pasien dalam ruangan yang harus dilayani
dulu oleh dokter. Anda giliran ke-",
rumahsakit.denah.contents[choice_holder[choice][0]][choice_holder
[choice][1]].antrian.counter, ".")

```

### 15. F15 - Antrian Saya

```

function cari_user (input UserData: ListDinUser, input ID:
integer) → string
{ Mencari username berdasarkan ID user }
{ I.S. UserData dan ID terdefinisi }
{ F.S. Mengembalikan username jika ditemukan, "--" jika
tidak }

procedure antrianSaya(input/output UserData:
ListDinUser, input/output rumahsakit: Config, input
loginId: integer)
{ Menampilkan status antrian check-up user yang sedang
login }
{ I.S. UserData, rumahsakit, dan loginId terdefinisi }
{ F.S. Menampilkan status antrian user atau pesan jika
belum terdaftar }

KAMUS LOKAL
i, j: integer
    current: Node
    posisi_antrian: integer

ALGORITMA
i traversal [0..rumahsakit.denah.rows-1]
    j traversal [0..rumahsakit.denah.cols-1]
        current ←
        rumahsakit.denah.contents[i][j].antrian.front
        while (current ≠ NULL) do
            if (current.data = loginId) then
                if (rumahsakit.denah.contents[i][j].antrian.counter ≤ 3)
                then
                    output("Anda sedang
berada di dalam ruangan dokter!")
                else
                    output("Status
antrian anda:")
                output("Dokter:
Dr.", cari_user(UserData,
rumahsakit.denah.contents[i][j].dokterID))

```

```

                output ("Ruangan:",
rumahsakit.denah.contents[i][j].kodeRuangan)
posisi_antrian ←
rumahsakit.denah.contents[i][j].antrian.counter -
rumahsakit.kapasitasRuangan
                output ("Posisi
antrian:", posisi_antrian, "dari 4")
                current ← current.next
output ("Anda belum terdaftar dalam antrian check-up!")
output ("Silakan daftar terlebih dahulu dengan command
DAFTAR_CHECKUP.")

```

#### 16. F16 - Minum Obat + B05 - Dead or Alive

```

procedure MinumObat(input/output loginID : integer,
                     input/output loginState : integer,
                     input/output config : Config,
                     input/output listUser : ListDinUser,
                     input listObat : ListObat,
                     input listPenyakit : ListPenyakit,
                     input mapObatPenyakit : MapObatPenyakit)

{ I.S. Pasien dalam keadaan login dan memiliki inventory atau
tidak }
{ F.S. Pasien memilih obat untuk diminum, sistem mengecek urutan,
nyawa berkurang jika salah.
    Jika nyawa habis, pasien dikeluarkan dari sistem. }

```

#### KAMUS LOKAL

```

userIdx, i, j, nomorObat, obatPilihan, obatID, obatIdx :
integer
penyakitId, urutanObat : integer
obatAvailable : array [1..MAX_OBAT_PER_PENYAKIT] of integer
posisiPasien : array[0..1] of integer

```

#### ALGORITMA

```

userIdx ← cariIdxUser(listUser, loginID)

if isMatriksRowEmpty(config.inventoryPasien, loginID) then
    output ("Kamu tidak memiliki obat!")
    return

output ("===== DAFTAR OBAT =====")
nomorObat ← 1

i ← 0
while i < MAX_OBAT_PER_PENYAKIT do
    obatID ← config.inventoryPasien.contents[loginID][i]
    if obatID ≠ 0 then
        obatIdx ← cariIdxObat(listObat, obatID)
        output(nomorObat, ". ",
listObat.items[obatIdx].nama_obat)
        obatAvailable[nomorObat] ← obatID

```

```

        nomorObat ← nomorObat + 1
        i ← i + 1
    { i = MAX_OBAT_PER PENYAKIT }

    output("Pilih obat untuk diminum: ")
    input(obatPilihan)

    while obatPilihan < 1 or obatPilihan ≥ nomorObat do
        output("Pilihan nomor tidak tersedia!")
        output("Pilih obat untuk diminum: ")
        input(obatPilihan)
    { obatPilihan >= 1 and obatPilihan < nomorObat }

    obatIdx ← cariIdxObat(listObat, obatAvailable[obatPilihan])

    if config.perutPasien[loginID].head = NIL then
        config.jumlahPerutPasien ← config.jumlahPerutPasien + 1

        output("GLEKGLEKGLEK... ", listObat.items[obatIdx].nama_obat,
" berhasil diminum!!!!")
        push(config.perutPasien[loginID], obatAvailable[obatPilihan])

        i ← 0
        while i < MAX_OBAT_PER PENYAKIT do
            if config.inventoryPasien.contents[loginID][i] =
obatAvailable[obatPilihan] then
                j ← i
                while j < MAX_OBAT_PER PENYAKIT - 1 do
                    if config.inventoryPasien.contents[loginID][j] =
0 then
                        config.inventoryPasien.contents[loginID][j+1]
← 0
                        j ← MAX_OBAT_PER PENYAKIT
                    else
                        config.inventoryPasien.contents[loginID][j] ←
config.inventoryPasien.contents[loginID][j+1]
                        j ← j + 1
                    { j = MAX_OBAT_PER PENYAKIT }
                    i ← MAX_OBAT_PER PENYAKIT
                else
                    i ← i + 1
    { i = MAX_OBAT_PER PENYAKIT }

    penyakitId ← searchPenyakitIDByName(listPenyakit,
listUser.buffer[userIdx].riwayat_penyakit)
    urutanObat ← config.perutPasien[loginID].length

    if peekStack(config.perutPasien[loginID]) ≠
mapObatPenyakit.items[penyakitId].value[urutanObat].obat_id then
        listUser.buffer[userIdx].nyawa ←
listUser.buffer[userIdx].nyawa - 1
        output("Kamu salah minum obat! Nyawamu berkurang menjadi
", listUser.buffer[userIdx].nyawa)

```

```

        i ← 0
        while i < listUser.buffer[userIdx].nyawa do
            output("❤")
            i ← i + 1
        i ← 0
        while i < 3 - listUser.buffer[userIdx].nyawa do
            output("💔")
            i ← i + 1
        output("Segera kontak dokter untuk minum penawar!")

if listUser.buffer[userIdx].nyawa = 0 then
    posisiPasien ← <-1, -1>
    i ← 0
    while i < config.denah.rows do
        j ← 0
        while j < config.denah.cols do
            if config.denah.contents[i][j].antrian.front.data
= loginID then
                posisiPasien ← <i, j>
                j ← config.denah.cols
                j ← j + 1
                { j = config.denah.cols }
                if posisiPasien[0] = i then
                    i ← config.denah.rows
                else
                    i ← i + 1
                { i = config.denah.rows }

dequeue(config.denah.contents[posisiPasien[0]][posisiPasien[1]].a
ntrian)
    deleteUser(listUser, userIdx)

    i ← 0
    while i < MAX_OBAT_PER_PENYAKIT do
        insertMatrixByIndex(config.inventoryPasien, loginID,
i, 0)
        i ← i + 1
    { i = MAX_OBAT_PER_PENYAKIT }

config.jumlahPemilikObat ← config.jumlahPemilikObat - 1

    i ← 0
    while i < config.perutPasien[loginID].length do
        pop(config.perutPasien[loginID])
        i ← i + 1
    { i = banyakObat }

config.jumlahPerutPasien ← config.jumlahPerutPasien - 1
printAsciiDed()
    output("Mohon maaf, karena kamu telah meminum obat yang
salah sebanyak tiga kali, rumah sakit enggan untuk memberimu
penawar")

```

```

    output ("Kamu dinyatakan ded ")

    logout(loginState, loginID)
    return

    if config.inventoryPasien.contents[loginID][0] = 0 then
        config.jumlahPemilikObat ← config.jumlahPemilikObat - 1

```

### 17. F17 - Minum Penawar

```

procedure MinumPenawar(input loginID : integer,
                      input/output config : Config,
                      input listUser : ListDinUser,
                      input listObat : ListObat,
                      input listPenyakit : ListPenyakit,
                      input mapObatPenyakit : MapObatPenyakit)

{ I.S.: perutPasien[loginID] mungkin kosong atau berisi obat }
{ F.S.: Obat terakhir yang diminum dikeluarkan dari perut dan
dimasukkan kembali ke inventory }

```

#### KAMUS LOKAL

```

obatID, obatIdx : integer
i : integer

```

#### ALGORITMA

```

if config.perutPasien[loginID].head = NIL then
    output("Perut kosong!! Belum ada obat yang dimakan.")
    return

    obatID ← pop(config.perutPasien[loginID])
    obatIdx ← cariIdxObat(listObat, obatID)

    output("Uwekkk!!! ", listObat.items[obatIdx].nama_obat, "
keluar dan kembali ke inventory.")

    i ← 1
    while i < MAX_OBAT_PER_PENYAKIT do
        if config.inventoryPasien.contents[loginID][i] = 0 then
            config.inventoryPasien.contents[loginID][i] ← obatID
            return
        i ← i + 1
    { i = MAX_OBAT_PER_PENYAKIT (sudah dimasukkan ke inventory )
}

```

### 18. F18 - Exit

```

procedure exitProgram(input/output run: integer,
                      input/output UserData: ListDinUser,
                      input/output listObat: ListObat,
                      input/output listPenyakit: ListPenyakit,
                      input/output mapObatPenyakit:
MapObatPenyakit,
                      input/output rumahsakit: Config)

```

**KAMUS LOKAL**

```
input: array [0..99] of char
konfirmasi: char
```

**ALGORITMA**

```
while (getchar() != '\n') do
    repeat
        output("Apakah Anda mau melakukan penyimpanan file yang
sudah diubah? (y/n) ")
        input(input)
        output()

        input[panjangString(input) - 1] <- '\0'

        if (panjangString(input) = 1) and ((input = 'y') or
(input = 'n')) then
            konfirmasi <- input
            break
        else
            output("Input tidak valid! Masukkan hanya satu
karakter: 'y' atau 'n.'")
        until false

        if (konfirmasi = 'y') then
            output("Data akan disimpan.")
            save(UserData, listObat, listPenyakit, mapObatPenyakit,
rumahsakit)
        else
            output("Data tidak disimpan.")

        printLogo()
        output("THANK YOU FOR USING OUR SERVICE !!!")
    run <- 0
```

**19. B02 - Denah dinamis**

```
function checkRuangKosong (rumahSakit : Config, barisNew :
integer, kolomNew : integer) → integer
{ I.S. rumahsakit, barisNew, dan kolomNew terdefinisi. barisNew
atau kolomNew lebih kecil dari ukuran awal }
{ F.S. Melakukan pengecekan dari barisNew dan kolomNew sampai ke
baris dan kolom lama }

procedure ubahDenah(input/output rumahsakit : Config)
{ I.S. rumahsakit terdefinisi }
{ F.S. Mengambil inputan user berupa 2 integer dan mengganti
konfigurasi ruangan berdasarkan inputan user jika ruangan yang
akan dihilangkan kosong }
```

**KAMUS LOKAL**

```
barisNew : integer
kolomNew : integer
```

**ALGORITMA**

```

input(barisNew, kolomNew)
if (barisNew < rumahSakit.denah.rows or
     kolomNew < rumahSakit.denah.cols) then
     if (checkRuanganKosong(rumahSakit, barisNew, \
         kolomNew) then
         output("Denah rumah sakit berhasil diubah
menjadi +
         barisNew + " baris dan " + kolomNew + "
kolom")
         rumahsakit.denah.rows ← barisNew
         rumahSakit.denah.cols ← kolomNew
     else
         output("Ruangan tidak kosong, pindahkan dokter
terlebih dahulu")

else
    output("Denah rumah sakit berhasil diubah menjadi +
barisNew + " baris dan " + kolomNew + " kolom")
    rumahsakit.denah.rows ← barisNew
    rumahSakit.denah.cols ← kolomNew

procedure pindahDokter(input/output userData : ListDinUser,
                        input/output rumahsakit : Config)
{ I.S. userData dan rumahsakit terdefinisi }
{ F.S. Mengambil inputan user berubah 2 integer dan memindahkan
isi ruangan awal ke ruangan tujuan jika ruangan tujuan kosong }

KAMUS LOKAL
kodeRuanganNew : array[0..10] of char
kodeRuanganNow : array[0..10] of char
barisNow, kolomNow, barisNew, kolomNew: integer

```

#### **ALGORITMA**

```

iterate
    input(kodeRuanganNow, kodeRuanganNew)
stop Length(kodeRuanganNow)= 2 and length(kodeRuanganNew)
= 2
    output("Input tidak valid! Format harus 1 huruf
kapital
            dan 1 angka (contoh A1)")

    barisNow ← ASCII(kodeRuanganNow[0])
    kolomNow ← ASCII(kodeRuanganNow[1])
    barisNew ← ASCII(kodeRuanganNow[0])
    kolomNew ← ASCII(kodeRuanganNow[1])

    if(rumahsakit.denah.contents[barisNow] [kolomNow].dokterID
== 0)then
        output("Tidak ada dokter di ruangan awal")

    if (barisNow < 0 OR kolomNow < 0
        OR barisNow >= rumahSakit.denah.rows
        OR kolomNow >= rumahSakit.denah.cols)
        then

```

```

output("Ruangan awal tidak ditemukan")

if (barisNow < 0 OR kolomNow < 0 OR barisNow >=
      rumahSakit.denah.rows OR kolomNow >=
      rumahSakit.denah.cols)
then
output("Ruangan akhir tidak ditemukan")

if rumahSakit.denah.contents[barisNow][kolomNow].dokterID
≠ 0
then
  rumahSakit.denah.contents[barisNew][kolomNew] ←
  rumahSakit.denah.contents[barisNow][kolomNow]

  rumahSakit.denah.contents[barisNow][kolomNow].
  dokterID ← 0

  idDokter
  ←rumahSakit.denah.contents[barisNew][kolomNew].dokte
  rID

  userData.buffer[cariIdxUser(userData,
  idDokter)].ruang ← kodeRuangNew
  output("Berhasil memindahkan dokter ",

else
  output("Ruangan awal tidak memiliki dokter")

```

**20. B06 - Mainin Antrian (Skip Antrian dan Cancel Antrian)**

```

procedure SkipAntrian(input loginID : integer,
                      input/output config : Config,
                      input listUser : ListDinUser)
{ Memindahkan pasien ke antrian paling depan }
{ I.S. Semua argumen terdefinisi, loginID adalah ID pasien }
{ F.S. Posisi pasien dalam antrian menjadi terdepan jika pasien
terdapat pada antrian;
  selain itu, tidak dilakukan apapun }

```

#### KAMUS LOKAL

```

inAntrian : boolean
posisi : array[0..1] of integer
i, j, tempId : integer
temp : pointer to Node
tempQueue : pointer to Queue
namaDokter : string

```

#### ALGORITMA

```

inAntrian ← false
posisi ← <-1, -1>

i ← 0
while i < config.denah.rows and not inAntrian do
  j ← 0
  while j < config.denah.cols and not inAntrian do

```

```

        if config.denah.contents[i][j].dokterID ≠ 0 then
            temp ← config.denah.contents[i][j].antrian.front
            while temp ≠ NIL and not inAntrian do
                if temp.data = loginID then
                    inAntrian ← true
                    posisi ← <i, j>
                    temp ← temp.next
                    { temp = NIL or ditemukan }
                    j ← j + 1
                    { j = config.denah.cols or ditemukan }
                    i ← i + 1
                    { i = config.denah.rows or ditemukan }

                if not inAntrian then
                    output("Skip antrian gagal! Anda tidak sedang terdaftar
dalam antrian manapun!")
                    return

                else if
peekQueue(config.denah.contents[posisi[0]][posisi[1]].antrian) =
loginID then
                    output("Anda sudah berada di posisi paling depan antrian!
Tidak bisa skip lagi.")
                    return

            tempQueue ← CreateQueue()

            while
config.denah.contents[posisi[0]][posisi[1]].antrian.front ≠ NIL
do
            tempId ←
dequeue(config.denah.contents[posisi[0]][posisi[1]].antrian)
            enqueue(tempQueue, tempId)
            { antrian utama kosong }

            enqueue(config.denah.contents[posisi[0]][posisi[1]].antrian,
loginID)

            while tempQueue.front ≠ NIL do
                tempId ← dequeue(tempQueue)
                if tempId ≠ loginID then

enqueue(config.denah.contents[posisi[0]][posisi[1]].antrian,
tempId)
            { tempQueue kosong }

            namaDokter ← cariUsername(listUser,
config.denah.contents[posisi[0]][posisi[1]].dokterID)
            output("Anda berhasil maju ke depan antrian Dr. ",
namaDokter,
            " di ruangan ",
config.denah.contents[posisi[0]][posisi[1]].kodeRuang, "!")
            output("Posisi antrian Anda sekarang: 1")

```

```

procedure CancelAntrian(input loginID : integer,
                      input/output config : Config,
                      input/output listUser : ListDinUser)
{ Menghapus pasien dari antrian }
{ I.S. Semua argumen terdefinisi, loginID adalah ID pasien }
{ F.S. Pasien hilang dari antrian yang ditempatinya dan datanya
direset jika pasien terdapat pada antrian; selain itu, tidak
dilakukan apapun }

```

#### **KAMUS LOKAL**

```

inAntrian : boolean
posisi : array[0..1] of integer
i, j, tempId, userIdx, banyakObat : integer
temp : pointer to Node
tempQueue : pointer to Queue
namaDokter : string

```

#### **ALGORITMA**

```

inAntrian ← false
posisi ← <-1, -1>

i ← 0
while i < config.denah.rows and not inAntrian do
    j ← 0
    while j < config.denah.cols and not inAntrian do
        if config.denah.contents[i][j].dokterID ≠ 0 then
            temp ← config.denah.contents[i][j].antrian.front
            while temp ≠ NIL and not inAntrian do
                if temp.data = loginID then
                    inAntrian ← true
                    posisi ← <i, j>
                    temp ← temp.next
                { temp = NIL or ditemukan }
                j ← j + 1
            i ← i + 1
        { i = config.denah.rows or ditemukan }

        if not inAntrian then
            output("Pembatalan antrian gagal! Anda tidak sedang
terdaftar dalam antrian manapun!")
            return

tempQueue ← CreateQueue()

while
config.denah.contents[posisi[0]][posisi[1]].antrian.front ≠ NIL
do
    tempId ←
dequeue(config.denah.contents[posisi[0]][posisi[1]].antrian)
    enqueue(tempQueue, tempId)
{ antrian utama kosong }

```

```

while tempQueue.front ≠ NIL do
    tempId ← dequeue(tempQueue)
    if tempId ≠ loginID then

enqueue(config.denah.contents[posisi[0]][posisi[1]].antrian,
tempId)
{ tempQueue kosong }

userIdx ← cariIdxUser(listUser, loginID)
resetUserData(listUser.buffer[userIdx])

i ← 0
while i < config.inventoryPasien.cols do
    config.inventoryPasien.contents[loginID][i] ← 0
    i ← i + 1
{ semua obat dihapus dari inventory }

config.jumlahPemilikObat ← config.jumlahPemilikObat - 1

banyakObat ← config.perutPasien[loginID].length
i ← 0
while i < banyakObat do
    pop(config.perutPasien[loginID])
    i ← i + 1
{ semua obat dikeluarkan dari perut }

config.jumlahPerutPasien ← config.jumlahPerutPasien - 1

namaDokter ← cariUsername(listUser,
config.denah.contents[posisi[0]][posisi[1]].dokterID)
output("Anda berhasil keluar dari antrian Dr. ", namaDokter,
" di ruangan ",
config.denah.contents[posisi[0]][posisi[1]].kodeRuangan, ".")

```

# Dokumentasi

## F01 - Login

- Deskripsi Fungsi ini digunakan untuk otentikasi pengguna agar dapat mengakses sistem. Pengguna memasukkan username dan password untuk memverifikasi identitas mereka.
- Testing
  - Login dengan username dan password yang benar.

```
15. EXIT          : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGOUT
Sampai Jumpa!

>>> HELP
=====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. SAVE       : Menyimpan kondisi rumah sakit
4. EXIT       : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass33
Halo kebin! Ada keluhan apa?

>>> 
```

Gambar 9.1a Dokumentasi F01 - Login - Login Sukses

- Login dengan username yang salah.

```

>>> HELP
=====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN    : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER : Membuat akun baru
3. SAVE     : Menyimpan kondisi rumah sakit
4. EXIT      : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass33
Halo kebin! Ada keluhan apa?

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): kebinn
Password (max 20 characters): pass33
Tidak ada user di database dengan username kebinn!

>>>

```

Gambar 9.1b Dokumentasi F01 - Login - Username Salah

- Login dengan password yang salah.

```

4. EXIT      : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass33
Halo kebin! Ada keluhan apa?

>>> LOGOUT
Logout gagal!
Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass333
Password yang dimasukkan salah!

>>>

```

Gambar 9.1c Dokumentasi F01 - Login - Password Salah

- Login dari berbagai peran pengguna (manager, dokter, pasien) untuk memastikan diarahkan ke menu yang sesuai.

```

Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass333
Password yang dimasukkan salah!

>>> LOGIN
Username (max 20 characters): neronimo
Password (max 20 characters): pass1010
Halo Dokter neronimo

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): zero
Password (max 20 characters): pass77
Halo Manager zero

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): kebin
Password (max 20 characters): pass33
Halo kebin! Ada keluhan apa?

>>> █

```

Gambar 9.1d Dokumentasi F01 - Login - Login Berbagai Peran

## F02 - Register Pasien

- Deskripsi Fungsi ini memungkinkan pendaftaran pengguna baru dengan peran pasien ke dalam sistem. Memerlukan username unik untuk setiap pendaftaran.
- Testing
  - Registrasi pasien baru dengan username unik.

```

3. ANTRIAN      : Melihat status antrian
4. LIHAT_DENAH  : Membuka denah rumah sakit
5. LIHAT_RUANGAN : Melihat isi dari ruangan pada rumah sakit
6. PULANGDOK   : Berkonsultasi kembali ke dokter untuk menanyakan apakah kondisi sudah baik dan boleh pulang
7. MINUM_OBAT   : Melihat daftar obat dan memilih obat yang akan diminum
8. MINUM_PENAWAR : Meminum penawar jika salah minum obat dan mengeluarkan obat terakhir yang diminum
9. SAVE         : Menyimpan kondisi rumah sakit
10. EXIT        : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGOUT
Sampai Jumpa!

>>> REGISTER
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Pasien egalt berhasil ditambahkan!

>>> REGISTER
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Registrasi gagal! User dengan nama egalt sudah terdaftar.

>>> █

```

Gambar 9.2a Dokumentasi F02 - Register Pasien - Registrasi Sukses

- Registrasi pasien baru dengan username yang sudah ada.

```

3. ANTRIAN      : Melihat status antrian
4. LIHAT_DENAH : Membuka denah rumah sakit
5. LIHAT_RUANGAN : Melihat isi dari ruangan pada rumah sakit
6. PULANGDOK   : Berkonsultasi kembali ke dokter untuk menanyakan apakah kondisi sudah baik dan boleh pulang
7. MINUM_OBAT    : Melihat daftar obat dan memilih obat yang akan diminum
8. MINUM_PENAWAR : Meminum penawar jika salah minum obat dan mengeluarkan obat terakhir yang diminum
9. SAVE         : Menyimpan kondisi rumah sakit
10. EXIT        : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGOUT
Sampai Jumpa!

>>> REGISTER
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Pasien egalt berhasil ditambahkan!

>>> REGISTER
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Registrasi gagal! User dengan nama egalt sudah terdaftar.

>>> █

```

Gambar 9.2b Dokumentasi F02 - Register Pasien - Username Sudah Ada

### F03 - Logout

- Deskripsi Fungsi ini mengakhiri sesi pengguna yang sedang aktif, mengeluarkan mereka dari sistem.
- Testing
  - Logout dari akun yang sedang aktif.

```

Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Registrasi gagal! User dengan nama egalt sudah terdaftar.

>>> ANTRIAN
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LOGIN
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Halo egalt! Ada keluhan apa?

>>> LIHAT_DENAH
     1   2   3
     +---+---+---+
A | A1  | A2  | A3  |
     +---+---+---+
B | B1  | B2  | B3  |
     +---+---+---+

>>> LOGOUT
Sampai Jumpa!

>>> LIHAT_DENAH
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>>

```

Gambar 9.3a Dokumentasi F03 - Logout - Logout Sukses

- Mencoba mengakses fitur yang memerlukan login setelah logout.

```

Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Registrasi gagal! User dengan nama egalt sudah terdaftar.

>>> ANTRIAN
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LOGIN
Username (max 20 characters): egalt
Password (max 20 characters): 12345678
Halo egalt! Ada keluhan apa?

>>> LIHAT_DENAH
    1   2   3
+---+---+---+
A | A1 | A2 | A3 |
+---+---+---+
B | B1 | B2 | B3 |
+---+---+---+

>>> LOGOUT
Sampai Jumpa!

>>> LIHAT_DENAH
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>>

```

Gambar 9.3b Dokumentasi F03 - Logout - Akses Setelah Logout

#### F04 - Lupa Password

- Deskripsi Fungsi ini membantu pengguna yang lupa password mereka untuk mengatur ulang password melalui proses verifikasi identitas, biasanya melibatkan kode unik.
- Testing
  - Meminta pemulihan password dengan username yang terdaftar.

```

=====
SELAMAT DATANG
=====

>>> HELP
=====
HELP =====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. LUPA_PASSWORD : Mengubah password pengguna dengan menggunakan kode unik
4. SAVE       : Menyimpan kondisi rumah sakit
5. EXIT       : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Password Baru: pass123

>>> █

```

Gambar 9.4a Dokumentasi F04 - Lupa Password - Username Terdaftar

- Meminta pemulihan password dengan username yang tidak terdaftar.

```

>>> HELP
=====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. LUPA_PASSWORD : Mengubah password pengguna dengan menggunakan kode unik
4. SAVE       : Menyimpan kondisi rumah sakit
5. EXIT       : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Password Baru: pass123

>>> LUPA_PASSWORD
Username (max 20 characters): tes
KodeUnik: tes

Tidak ada user di database dengan username tes!

>>>

```

Gambar 9.4b Dokumentasi F04 - Lupa Password - Username Tidak Terdaftar

- Memasukkan kode verifikasi yang benar.

```

>>> HELP
=====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. LUPA_PASSWORD : Mengubah password pengguna dengan menggunakan kode unik
4. SAVE       : Menyimpan kondisi rumah sakit
5. EXIT       : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Password Baru: pass123

>>> LUPA_PASSWORD
Username (max 20 characters): tes
KodeUnik: tes

Tidak ada user di database dengan username tes!

>>>

```

Gambar 9.4c Dokumentasi F04 - Lupa Password - Kode Verifikasi Benar

- Memasukkan kode verifikasi yang salah.

```
 Password Baru: pass123

>>> LUPA_PASSWORD
Username (max 20 characters): tes
KodeUnik: tes

Tidak ada user di database dengan username tes!

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebi2n

Tidak ada user di database dengan username kebin!

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Tidak ada user di database dengan username kebin!

>>> LUPA_PASSWORD
Username (max 20 characters): gro
KodeUnik: gr2o

Kode unik salah!

>>> 
```

Gambar 9.4d Dokumentasi F04 - Lupa Password - Kode Verifikasi Salah

- Membuat password baru dengan format yang valid.

```
Tidak ada user di database dengan username tes!

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebi2n

Tidak ada user di database dengan username kebin!

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Tidak ada user di database dengan username kebin!

>>> LUPA_PASSWORD
Username (max 20 characters): gro
KodeUnik: gr2o

Kode unik salah!

>>> LUPA_PASSWORD
Username (max 20 characters): gro
KodeUnik: gro

Password Baru: gro123

>>> 
```

Gambar 9.4e Dokumentasi F04 - Lupa Password - Buat Password Baru Valid

- Validasi input username dan kode unik.

```

>>> LUPA_PASSWORD
Username (max 20 characters): kebin
KodeUnik: kebin

Tidak ada user di database dengan username kebin!

>>> LUPA_PASSWORD
Username (max 20 characters): gro
KodeUnik: gr2o

Kode unik salah!

>>> LUPA_PASSWORD
Username (max 20 characters): gro
KodeUnik: gro

Password Baru: gro123

>>> LOGIN
Username (max 20 characters): gro
Password (max 20 characters): gro123

>>> LOGIN
Username (max 20 characters): gro
Password (max 20 characters): pass22
Password yang dimasukkan salah!

>>> 

```

Gambar 9.4f Dokumentasi F04 - Lupa Password - Validasi Input

#### F05 - Menu & Help

- Deskripsi Fungsi ini berfungsi untuk menampilkan daftar perintah atau menu yang relevan dengan peran pengguna saat ini, serta menyediakan informasi bantuan.
- Testing
  - Menampilkan menu dan perintah yang tersedia untuk role manager

```

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): zero
Password (max 20 characters): pass77
Halo Manager zero

>>> HELP
=====
Halo Manager zero. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

1. LOGOUT : Keluar dari akun yang sedang digunakan
2. TAMBAH_DOKTER : Mendaftarkan dokter baru ke sistem
3. Lihat_Denah : Melihat denah rumah sakit
4. Lihat_Ruangan : Melihat isi dari ruangan pada rumah sakit
5. Lihat_User : Melihat data seluruh pengguna
6. Lihat_Pasien : Melihat data seluruh pasien
7. Lihat_Dokter : Melihat data seluruh dokter
8. Cari_User : Mencari pengguna tertentu berdasarkan ID atau Nama
9. Cari_Pasien : Mencari pasien tertentu berdasarkan ID, Nama, atau Penyakit
10. Cari_Dokter : Mencari dokter tertentu berdasarkan ID atau Nama
11. Lihat_Semua_Antri : Melihat rincian seluruh ruangan saat ini
12. TAMBAH_DOKTER : Menambah dokter baru dengan username dan password yang ditentukan
13. ASSIGN_DOKTER : Melakukan assign ruangan ke dokter tertentu yang belum memiliki ruangan
14. UBAH_DENAH : Mengganti ukuran ruangan rumah sakit
15. PINDAH_DOK : Memindahkan ruangan dokter ke ruangan baru beserta dengan antriannya
16. SAVE : Menyimpan kondisi rumah sakit
17. EXIT : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>>

```

Gambar 9.5a Dokumentasi F05 - Menu & Help - Manager

- Menampilkan menu dan perintah untuk pengguna yang belum login.

```
ifi1210-tubes-2025-k01-k/src on [?] via C v14.2.0-gcc
> ./main load

[REDACTED LOGO]

=====
SELAMAT DATANG
=====

>>> HELP
=====
Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. LUPA_PASSWORD : Mengubah password pengguna dengan menggunakan kode unik
4. SAVE       : Menyimpan kondisi rumah sakit
5. EXIT       : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> █
```

Gambar 9.5b Dokumentasi F05 - Menu & Help - Menu Pengguna Belum Login

- Memastikan menu yang ditampilkan sesuai dengan peran pengguna (dokter, pasien).

```
1. LOGIN      : Masuk ke dalam akun yang sudah terdaftar
2. REGISTER   : Membuat akun baru
3. LUPA_PASSWORD : Mengubah password pengguna dengan menggunakan kode unik
4. SAVE       : Menyimpan kondisi rumah sakit
5. EXIT       : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGOUT
Logout gagal!
Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout

>>> LOGIN
Username (max 20 characters): neronimo
Password (max 20 characters): pass1010
Halo Dokter neronimo

>>> HELP
=====
Halo Dokter neronimo. Kamu memanggil HELP. Berikut command yang tersedia:

1. LOGOUT      : Keluar dari akun
2. DIAGNOSIS   : Melakukan diagnosis pasien
3. NGOBATIN    : Meresepkan obat
4. LIHAT_DENAH  : Membuka denah rumah sakit
5. LIHAT_RUANGAN : Melihat isi dari ruangan pada rumah sakit
6. SAVE        : Menyimpan kondisi rumah sakit
7. EXIT        : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> █
```

```

17. EXIT : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): gro
Password (max 20 characters): pass22
Halo gro! Ada keluhan apa?

>>> HELP
===== HELP =====
Selamat datang, gro. Berikut command yang tersedia:

1. LOGOUT : Keluar dari akun
2. DAFTAR_CHECKUP : Mendaftarkan diri untuk pemeriksaan
3. ANTRIAN : Melihat status antrian
4. LIHAT_DENAH : Membuka denah rumah sakit
5. LIHAT_RUANGAN : Melihat isi dari ruangan pada rumah sakit
6. PULANGDOK : Berkonsultasi kembali ke dokter untuk menanyakan apakah kondisi sudah baik dan boleh pulang
7. MINUM_OBAT : Melihat daftar obat dan memilih obat yang akan diminum
8. MINUM_PENAWAR : Meminum penawar jika salah minum obat dan mengetuarkan obat terakhir yang diminum
9. SKIP_ANTRIAN : Langsung maju ke posisi terdepan
10. CANCEL_ANTRIAN : Keluar sepenuhnya dari ruangan
11. SAVE : Menyimpan kondisi rumah sakit
12. EXIT : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> 

```

Gambar 9.5c Dokumentasi F05 - Menu & Help (Sesuai Peran)

#### F06 - Lihat Denah

- Deskripsi Fungsi ini menampilkan visualisasi denah rumah sakit dan memungkinkan pengguna untuk melihat detail data dari ruangan tertentu.
- Testing
  - Menampilkan denah rumah sakit.

```

1. LOGOUT : Keluar dari akun
2. DIAGNOSIS : Melakukan diagnosis pasien
3. NGOBATIN : Meresepkan obat
4. LIHAT_DENAH : Membuka denah rumah sakit
5. LIHAT_RUANGAN : Melihat isi dari ruangan pada rumah sakit
6. SAVE : Menyimpan kondisi rumah sakit
7. EXIT : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> LIHAT DENAH
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LIHAT_DENAH
     1   2   3
     +---+---+---+
A | A1 | A2 | A3 |
     +---+---+---+
B | B1 | B2 | B3 |
     +---+---+---+
>>>

```

Gambar 9.6a Dokumentasi F06 - Lihat Denah - Tampil Denah

- Menampilkan data ruangan dengan kode ruangan yang valid (fungsi Lihat\_Ruangan).

```

>>> LIHAT DENAH
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LIHAT_DENAH
    1   2   3
+---+---+---+
A | A1 | A2 | A3 |
+---+---+---+
B | B1 | B2 | B3 |
+---+---+---+

>>> LIHAT_RUANGAN
Ruangan: A1

--- Detail Ruangan A1 ---
Kapasitas : 3
Dokter : neronimo
Pasien di dalam ruangan:
1. gro
2. kebin
3. stewart
-----
>>>

```

Gambar 9.6b Dokumentasi F06 - Lihat Denah - Lihat Ruangan Valid

- Menampilkan data ruangan dengan kode ruangan yang tidak valid (fungsi Lihat\_Ruangan).

```

>>> PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LIHAT_DENAH
    1   2   3
+---+---+---+
A | A1 | A2 | A3 |
+---+---+---+
B | B1 | B2 | B3 |
+---+---+---+

>>> LIHAT_RUANGAN
Ruangan: A1

--- Detail Ruangan A1 ---
Kapasitas : 3
Dokter : neronimo
Pasien di dalam ruangan:
1. gro
2. kebin
3. stewart
-----
>>> LIHAT_RUANGAN
Ruangan: B4
Ruangan tidak ditemukan!

>>>

```

Gambar 9.6c Dokumentasi F06 - Lihat Denah - Lihat Ruangan Tidak Valid

#### F07 - Lihat User

- Deskripsi Fungsi ini menampilkan daftar seluruh pengguna yang terdaftar dalam sistem, dengan opsi untuk menyortir daftar tersebut.
- Testing
  - Menampilkan daftar seluruh user.

```

>>> LIHAT_USER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan : 1

Urutan sort?
1. ASC (A-Z)
2. DES (Z-A)
>>> Pilihan : 1
Menampilkan semua pengguna dengan ID terurut ascending...
-----
ID | Nama           | Role    | Penyakit
---|---|---|---
1  | stewart       | pasien  | -
2  | gro            | pasien  | COVID-19
3  | kebin          | pasien  | -
4  | pop             | pasien  | Diabetes Mellitus
5  | opor            | pasien  | -
6  | nikeb           | pasien  | -
7  | tuart           | pasien  | -
8  | minonette      | pasien  | -
9  | tobo            | pasien  | -
10 | neronimo       | dokter  | -
10 | ropik           | pasien  | -
11 | ciciko          | dokter  | -
12 | cacako          | dokter  | -
13 | kroket           | dokter  | -
14 | junior           | pasien  | -
15 | risol            | dokter  | -

```

Gambar 9.7a Dokumentasi F07 - Lihat User - Tampil Semua User

- Menampilkan daftar user dengan opsi sort (misalnya, berdasarkan ID, nama, peran).

```

>>> LIHAT_USER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan : 1

Urutan sort?
1. ASC (A-Z)
2. DES (Z-A)
>>> Pilihan : 1
Menampilkan semua pengguna dengan ID terurut ascending...
-----
ID | Nama           | Role    | Penyakit
---|---|---|---
1  | stewart       | pasien  | -
2  | gro            | pasien  | COVID-19
3  | kebin          | pasien  | -
4  | pop             | pasien  | Diabetes Mellitus
5  | opor            | pasien  | -
6  | nikeb           | pasien  | -
7  | tuart           | pasien  | -
8  | minonette      | pasien  | -
9  | tobo            | pasien  | -
10 | neronimo       | dokter  | -
10 | ropik           | pasien  | -
11 | ciciko          | dokter  | -
12 | cacako          | dokter  | -
13 | kroket           | dokter  | -
14 | junior           | pasien  | -
15 | risol            | dokter  | -

```

Gambar 9.7b Dokumentasi F07 - Lihat User - Tampil User Sort

#### F08 - Cari User

- Deskripsi Fungsi ini memungkinkan pencarian pengguna berdasarkan kriteria tertentu seperti ID, nama, atau penyakit yang dimiliki (khusus untuk pasien).
- Testing
  - Mencari user berdasarkan ID yang valid.

```

Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan : 2.
>>> Masukkan nama user: Tidak ditemukan user: dengan nama .!

>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan : 2
>>> Masukkan nama user: kebin
-----
ID | Nama           | Role   | Penyakit
-----
3 | kebin          | pasien | -
>>>
>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan : 1
>>> Masukkan nomor ID user: 5
-----
ID | Nama           | Role   | Penyakit
-----
5 | opor           | pasien | -
>>>

```

Gambar 9.8a Dokumentasi F08 - Cari User - Cari User ID Valid

- Mencari user berdasarkan nama yang valid.

```

11. LIHAT_SEMUA_ANTRIAN : Melihat rincian seluruh ruangan saat ini
12. TAMBAH_DOKTER      : Menambah dokter baru dengan username dan password yang ditentukan
13. ASSIGN_DOKTER       : Melakukan assign ruangan ke dokter tertentu yang belum memiliki ruangan
14. SAVE                : Menyimpan kondisi rumah sakit
15. EXIT                : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan : 2.
>>> Masukkan nama user: Tidak ditemukan user: dengan nama .!

>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan : 2
>>> Masukkan nama user: kebin
-----
ID | Nama           | Role   | Penyakit
-----
3 | kebin          | pasien | -
>>>

```

Gambar 9.8b Dokumentasi F08 - Cari User - Cari User Nama Valid

#### F09 - Lihat Antrian

- Deskripsi Fungsi ini menampilkan daftar nama-nama pengguna yang sedang dalam antrian untuk ruangan-ruangan yang memiliki dokter.
- Testing
  - Menampilkan antrian untuk ruangan yang memiliki dokter.

```

>>> LIHAT_SEMUA_ANTRIAN
      1   2   3
+---+---+---+
A | A1 | A2 | A3 |
+---+---+---+
B | B1 | B2 | B3 |
+---+---+---+
===== A1 =====
Kapasitas : 3
Dokter : neronimo
Pasien di dalam ruangan:
1. greg
2. kebin
3. stewart
Pasien di antrian:
1. tobokan
2. popokan
-----
===== A2 =====
Kapasitas : 3
Dokter : ciciko
Pasien di dalam ruangan:
1. pop
2. opor
Pasien di antrian:
Tidak ada pasien di antrian luar ruangan
-----
===== A3 =====
Kapasitas : 3
Dokter : cacako
Pasien di dalam ruangan:
1. junior
2. nikeb
Pasien di antrian:
Tidak ada pasien di antrian luar ruangan
-----
===== B1 =====
Kapasitas : 3
Dokter : kroket
Pasien di dalam ruangan:

```

Gambar 9.9a Dokumentasi F09 - Lihat Antrian - Antrian Ruangan Dokter

#### F10 - Tambah Dokter & Assign Dokter

- Deskripsi Fungsi ini memiliki dua bagian mendaftarkan dokter baru ke sistem dan menempatkan dokter tersebut ke ruangan tertentu di rumah sakit.
- Testing
  - Mendaftarkan dokter baru ke sistem dengan data yang valid.

```

5. LIHAT_USER           : Melihat data seluruh pengguna
6. LIHAT_PASIEN         : Melihat data seluruh pasien
7. LIHAT_DOKTER         : Melihat data seluruh dokter
8. CARI_USER            : Mencari pengguna tertentu berdasarkan ID atau Nama
9. CARI_PASIEN          : Mencari pasien tertentu berdasarkan ID, Nama, atau Penyakit
10. CARI_DOKTER          : Mencari dokter tertentu berdasarkan ID atau Nama
11. LIHAT_SEMUA_ANTRIAN : Melihat rincian seluruh ruangan saat ini
12. TAMBAH_DOKTER        : Menambah dokter baru dengan username dan password yang ditentukan
13. ASSIGN_DOKTER        : Melakukan assign ruangan ke dokter tertentu yang belum memiliki ruangan
14. SAVE                 : Menyimpan kondisi rumah sakit
15. EXIT                 : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 123456
Dokter almighty berhasil ditambahkan!

>>>

```

Gambar 9.10a Dokumentasi F10 - Tambah Dokter & Assign Dokter - Tambah Dokter Valid

- Mendaftarkan dokter baru dengan username yang sudah ada.

```

akit
10. CARI_DOKTER      : Mencari dokter tertentu berdasarkan ID atau Nama
11. LIHAT_SEMUA_ANTRIAN : Melihat rincian seluruh ruangan saat ini
12. TAMBAH_DOKTER      : Menambah dokter baru dengan username dan password yang
ditentukan
13. ASSIGN_DOKTER       : Melakukan assign ruangan ke dokter tertentu yang belum
memiliki ruangan
14. SAVE                 : Menyimpan kondisi rumah sakit
15. EXIT                  : Keluar dari program

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 123456
Dokter almighty berhasil ditambahkan!

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 9999
Penambahan dokter gagal! Sudah ada user bernama almighty!

>>>

```

Gambar 9.10b Dokumentasi F10 - Tambah Dokter & Assign Dokter -  
Tambah Dokter Username Ada

- Menempatkan dokter ke ruangan dengan kode ruangan yang valid (Assign Dokter).

```

13. ASSIGN_DOKTER      : Melakukan assign ruangan ke dokter tertentu yang belum
memiliki ruangan
14. SAVE                : Menyimpan kondisi rumah sakit
15. EXIT                : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 123456
Dokter almighty berhasil ditambahkan!

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 9999
Penambahan dokter gagal! Sudah ada user bernama almighty!

>>> ASSIGN_DOKTER
Username (max 20 characters): almighty
Ruang: B2
Dokter almighty berhasil diassign ke ruangan B2

>>>

```

Gambar 9.10c Dokumentasi F10 - Tambah Dokter & Assign Dokter -  
Assign Dokter Valid

- Menempatkan dokter ke ruangan dengan kode ruangan yang tidak valid (Assign Dokter).

```

Username (max 20 characters): almighty
Password (max 20 characters): 123456
Dokter almighty berhasil ditambahkan!

>>> TAMBAH_DOKTER
Username (max 20 characters): almighty
Password (max 20 characters): 9999
Penambahan dokter gagal! Sudah ada user bernama almighty!

>>> ASSIGN_DOKTER
Username (max 20 characters): almighty
Ruang: B2
Dokter almighty berhasil diassign ke ruangan B2

>>> TAMBAH_DOKTER
Username (max 20 characters): raven
Password (max 20 characters): 9999
Dokter raven berhasil ditambahkan!

>>> ASSIGN_DOKTER
Username (max 20 characters): raven
Ruang: B4
Ruangan tidak ditemukan!

>>>

```

Gambar 9.10d Dokumentasi F10 - Tambah Dokter & Assign Dokter -  
Assign Dokter Tidak Valid

## F11 - Diagnosis

- Deskripsi Fungsi ini melakukan proses diagnosis terhadap pasien berdasarkan riwayat penyakit atau kondisi medis yang mereka miliki.

- Testing
  - Mendiagnosis pasien berdasarkan riwayat penyakit atau kondisi pasien yang valid.

```

>>> DIAGNOSIS
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
 
>>> █

```

Gambar 9.11a Dokumentasi F11 - Diagnosis - Diagnosis Valid

- Mendiagnosis pasien dengan riwayat penyakit atau kondisi pasien yang tidak valid atau tidak lengkap.
- Gambar 9.11.2 Dokumentasi F11 - Diagnosis - Diagnosis Tidak Valid/Lengkap
- Hasil diagnosis yang berbeda berdasarkan input yang berbeda.
- Gambar 9.11.3 Dokumentasi F11 - Diagnosis - Hasil Diagnosis Berbeda

## F12 - Ngobatin

- **Deskripsi** Fungsi ini memberikan resep obat kepada pasien berdasarkan hasil diagnosis yang telah dilakukan sebelumnya.
- **Testing**
- Memberikan resep obat berdasarkan diagnosa yang telah dilakukan.



- Berkonsultasi ulang dengan dokter sebelum pasien dipulangkan.

```

9. SAVE      : Menyimpan kondisi rumah sakit
10. EXIT     : Keluar dari program
Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

>>> PULANGDOK
Kamu belum menerima diagnosis apapun dari dokter, jangan buru-buru pulang!

>>> LOGOUT
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): gro
Password (max 20 characters): pass22
Halo gro! Ada keluhan apa?

>>> PULANGDOK
Masih ada obat yang belum kamu habiskan, minum semuanya dulu yukk!

>>> MINUM_OBAT
===== DAFTAR OBAT =====
1. Remdesivir
Pilih obat untuk diminum: 1
GLEKGLEKGLEK... Remdesivir berhasil diminum!!!

>>> PULANGDOK
Dokter sedang memeriksa keadaanmu...

Selamat! Kamu sudah dinyatakan sembuh oleh dokter. Silahkan pulang dan semoga sehat selalu!

>>>

```

Gambar 9.13a Dokumentasi F13 - Pulangdok - Konsultasi Sebelum Pulang

#### F14 - Daftar Check-Up

- **Deskripsi** Fungsi ini digunakan untuk mendaftarkan pasien ke pemeriksaan medis (check-up) dengan memasukkan data kondisi tubuh mereka.
- **Testing**
- Mendaftarkan diri ke pemeriksaan medis dengan data kondisi tubuh yang valid.

```

>>> DAFTAR_CHECKUP
Silakan masukkan data check-up Anda:
Suhu Tubuh (Celcius): 36
Tekanan Darah (sistol diastol, contoh: 120 80): 119 75
Detak Jantung (bpm): 65
Saturasi Oksigen (%): 95
Kadar Gula Darah (mg/dL): 66
Berat Badan (kg): 66
Tinggi Badan (cm): 180
Kadar Kolesterol (mg/dL): 135
Trombosit (ribu/µL): 8
Berikut adalah daftar dokter yang tersedia:
1. Dr. ropik - Spesialisasi Umum - Ruangan A1 (Antrian: 1)- Aura 1
2. Dr. ciciko - Spesialisasi Umum - Ruangan A2 (Antrian: 0)- Aura 2
3. Dr. cacako - Spesialisasi Umum - Ruangan A3 (Antrian: 0)- Aura 100
4. Dr. kroket - Spesialisasi Umum - Ruangan B1 (Antrian: 0)- Aura 1
5. Dr. almighty - Spesialisasi Umum - Ruangan B2 (Antrian: 0)- Aura 0
6. Dr. risol - Spesialisasi Umum - Ruangan B3 (Antrian: 0)- Aura 0
Pilih dokter (contoh input: 1) : 3
:= = +*****+*****#**+
:= = + ... +
:= -.# ..... .+ .
:= =.#
:= = ... .... +
:= = * .. .:... .,* :
:= =.#
:= = .. .:=-.-..,* :
:= =.#
:= = .. %#=..,*@*:...,* :
:= =.#
:= = #:+...:#+. .@:...,* :
- + * :+...=+ - .....+ .----:-
- .+-+.. .%#+.. ,=,...,*-
- .*#%+..%-=+.*+::.-.*#####
- .+:#-::+.. #@*...-.**=...+...#
- .#%. ,--,-+,- =-:#####
#*+%#=--# : =---, *--*- -+;.....
.0%+%@*--%. , +====%:- -*-=%@%####%
.0%+%@*--%. , +====%:- -*-=%@%####%
.0%+%@*--%. , +====%:- -*-=%@%####%
.0%+%@*--%. , +====%:- -*-=%@%####%

```

Gambar 9.14a Dokumentasi F14 - Daftar Check-Up - Daftar Check-Up Valid

- Mendaftarkan diri dengan data kondisi tubuh yang tidak lengkap atau format yang salah.

```

Sampai Jumpa:
>>> LOGIN
Username (max 20 characters): tobo
Password (max 20 characters): pass99
Halo tobo! Ada keluhan apa?

>>> DAFTAR_CHECKUP
Silakan masukkan data check-up Anda:
Suhu Tubuh (Celcius): 99
Tekanan Darah (sistol diastol, contoh: 120 80): 999
999
Detak Jantung (bpm): 999
Saturasi Oksigen (%): 999
Kadar Gula Darah (mg/dL): 9999
Berat Badan (kg): 999
Tinggi Badan (cm): 999
Kadar Kolesterol (mg/dL): 999
Trombosit (ribu/µL): 999
Berikut adalah daftar dokter yang tersedia:
1. Dr. ropik - Spesialisasi Umum - Ruangan A1 (Antrian: 1)- Aura 1
2. Dr. ciciko - Spesialisasi Umum - Ruangan A2 (Antrian: 0)- Aura 2
3. Dr. cacako - Spesialisasi Umum - Ruangan A3 (Antrian: 0)- Aura 100
4. Dr. kroket - Spesialisasi Umum - Ruangan B1 (Antrian: 0)- Aura 1
5. Dr. almighty - Spesialisasi Umum - Ruangan B2 (Antrian: 0)- Aura 0
6. Dr. risol - Spesialisasi Umum - Ruangan B3 (Antrian: 0)- Aura 0
Pilih dokter (contoh input: 1) : 5
:= = +*****+*****#**%
:= = + ... +
:= -.# ..... .+ .
:= =.#
:= = ... .... +
:= = * .. .:... .,* :
:= =.#
:= = .. .:=-.-..,* :
:= =.#
:= = .. %#=..,*@*:...,* :
:= =.#
:= = #:+...:#+. .@:...,* :
- + * :+...=+ - .....+ .----:-

```

Gambar 9.14b Dokumentasi F14 - Daftar Check-Up - Daftar Check-Up Tidak Valid/Lengkap

## F15 - Antrian Saya!

- Deskripsi** Fungsi ini memungkinkan pasien yang sedang login untuk melihat status antrian mereka.
- Testing**
- Melihat status antrian pasien yang sedang login.

```

Masih ada pasien dalam ruangan yang harus dilayani dutu oleh dokter. Anda giliran ke-3.

>>> ANTRIAN
Anda belum terdaftar dalam antrian check-up!
Silakan daftar terlebih dahulu dengan command DAFTAR_CHECKUP.

>>> LOGOUT
PLEASE ENTER A VALID COMMAND!
TYPE "HELP" TO SHOW VALID COMMANDS!

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): gro
Password (max 20 characters): pass22
Halo gro! Ada keluhan apa?

>>> ANTRIAN
Anda belum terdaftar dalam antrian check-up!
Silakan daftar terlebih dahulu dengan command DAFTAR_CHECKUP.

>>> LOGOUT
Sampai Jumpa!

>>> LOGIN
Username (max 20 characters): popokan
Password (max 20 characters): passpopokan
Halo popokan! Ada keluhan apa?

>>> ANTRIAN
Status antrian anda:
Dokter: Dr. ropik
Ruang: A1
Posisi antrian: 1 dari 4

>>> █

```

Gambar 9.15a Dokumentasi F15 - Antrian Saya - Status Antrian Pasien

#### F16 - Minum Obat + B05 - Dead or Alive

- **Deskripsi** Fungsi ini memungkinkan pasien melihat dan memilih obat untuk diminum, serta memeriksa kondisi kesehatan mereka setelah minum obat.
- **Testing**
- Melihat daftar obat yang dimiliki pasien.

Gambar 9.16a Dokumentasi F16 - Minum Obat - Daftar Obat Pasien

- Memilih obat dari daftar obat yang dimiliki.

Gambar 9.16b Dokumentasi F16 - Minum Obat - Pilih Obat

- Meminimum obat yang valid.

Gambar 9.16c Dokumentasi F16 - Minum Obat - Minum Obat Valid

- Memeriksa kondisi kesehatan pasien setelah minum obat (Dead or Alive).

Gambar 9.16d Dokumentasi F16 - Minum Obat - Cek Kondisi Kesehatan

- Hasil pemeriksaan kondisi kesehatan yang berbeda tergantung obat yang diminum dan kondisi pasien.

Gambar 9.16e Dokumentasi F16 - Minum Obat - Hasil Kondisi Kesehatan Berbeda

#### F17 - Minum Penawar

- **Deskripsi** Fungsi ini memungkinkan pasien meminum penawar untuk membatalkan konsumsi obat terakhir jika terjadi salah minum.
- **Testing**
- Meminimum penawar setelah salah minum obat terakhir.

Gambar 9.17a Dokumentasi F17 - Minum Penawar - Minum Penawar Salah Obat

- Meminum penawar saat pasien tidak salah minum obat terakhir.

Gambar 9.17b Dokumentasi F17 - Minum Penawar - Minum Penawar Tidak Salah Obat

- Efek penawar terhadap kondisi kesehatan pasien.

Gambar 9.17c Dokumentasi F17 - Minum Penawar - Efek Penawar

F18 - Exit

- **Deskripsi** Fungsi ini berfungsi untuk menyimpan data kondisi sistem sebelum program berakhir.
- **Testing**
- Menyimpan data sebelum keluar dari program.

Gambar 9.18a Dokumentasi F18 - Exit - Simpan Data Sebelum Keluar

- Keluar dari program tanpa menyimpan data (jika ada opsi).

Gambar 9.18b Dokumentasi F18 - Exit - Keluar Tanpa Simpan

- Proses penyimpanan data yang berhasil.

Gambar 9.18c Dokumentasi F18 - Exit - Proses Simpan Sukses

D03 - Load

- **Deskripsi** Fungsi ini memuat data rumah sakit (user, obat, penyakit, peta obat-penyakit, dan konfigurasi) dari file eksternal ke dalam program.
- **Testing**

- Memuat data dari folder yang valid.

```

>>> LOGIN
Username (max 20 characters): zerro
Password (max 20 characters): pass77
Password yang dimasukkan salah!

>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) y
Data akan disimpan.
Masukkan nama folder: TES
Data berhasil disimpan di ../data/TES

RUMAH SAKIT JAWA
RUMAH SAKIT JAWA

SELAMAT DATANG
SELAMAT DATANG

THANK YOU FOR USING OUR SERVICE !!!
if1210-tubes-2025-k01-k/src on ✘ main [?] via C v14.2.0-gcc took 1m19s
> ./main TES

RUMAH SAKIT JAWA
RUMAH SAKIT JAWA

SELAMAT DATANG
SELAMAT DATANG

THANK YOU FOR USING OUR SERVICE !!!
if1210-tubes-2025-k01-k/src on ✘ main [?] via C v14.2.0-gcc took 1m19s
> ./main TES

RUMAH SAKIT JAWA
RUMAH SAKIT JAWA

SELAMAT DATANG
SELAMAT DATANG

THANK YOU FOR USING OUR SERVICE !!!
if1210-tubes-2025-k01-k/src on ✘ main [?] via C v14.2.0-gcc took 1m19s
> ./main TES
  
```

Gambar 9.19a Dokumentasi D03 - Load - Memuat Data Valid

- Memuat data dari folder yang tidak valid.

```

RUMAH SAKIT JAWA
RUMAH SAKIT JAWA

SELAMAT DATANG
SELAMAT DATANG

>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) n
Data tidak disimpan.

RUMAH SAKIT JAWA
RUMAH SAKIT JAWA

SELAMAT DATANG
SELAMAT DATANG

THANK YOU FOR USING OUR SERVICE !!!
if1210-tubes-2025-k01-k/src on ✘ main [?] via C v14.2.0-gcc took 1m33s
> ./main tes
FOLDER TIDAK DITEMUKAN! PASTIKAN FOLDER ADA DAN BERISI DATA YANG VALID!

if1210-tubes-2025-k01-k/src on ✘ main [?] via C v14.2.0-gcc
>
  
```

Gambar 9.19b Dokumentasi D03 - Load - Memuat Data Tidak Valid

#### D04 - Save

- **Deskripsi** Fungsi ini menyimpan data rumah sakit (user, obat, penyakit, peta obat-penyakit, dan konfigurasi) dari program ke dalam file eksternal.
- **Testing**

- Menyimpan data ke folder yang valid.

Gambar 9.20a Dokumentasi D04 - Save - Menyimpan Data Valid

- Menyimpan data ke folder yang tidak valid.

Gambar 9.20b Dokumentasi D04 - Save - Menyimpan Data Tidak Valid

- Menyimpan data dengan izin folder terbatas.

Gambar 9.20c Dokumentasi D04 - Save - Izin Folder Terbatas

#### B02 - Denah Dinamis

- **Deskripsi** Fungsi ini memungkinkan perubahan ukuran denah rumah sakit dan pemindahan dokter antar ruangan.
- **Testing**
  - Mengubah ukuran denah dengan input yang valid.

Gambar 9.21a Dokumentasi B02 - Denah Dinamis - Ubah Ukuran Valid

- Mengubah ukuran denah dengan input yang tidak valid.

Gambar 9.21b Dokumentasi B02 - Denah Dinamis - Ubah Ukuran Tidak Valid

- Memindahkan dokter ke ruangan yang kosong.

Gambar 9.21c Dokumentasi B02 - Denah Dinamis - Pindah Dokter Valid

- Memindahkan dokter ke ruangan yang sudah terisi.

Gambar 9.21d Dokumentasi B02 - Denah Dinamis - Pindah Dokter Tidak Valid

#### B03 - Aura!

- **Deskripsi** Fungsi ini mengurutkan daftar pengguna berdasarkan nilai aura mereka.
- **Testing**
  - Mengurutkan daftar pengguna dengan data yang ada.

Gambar 9.22a Dokumentasi B03 - Aura! - Urutkan Data

- Mengurutkan daftar pengguna dengan data kosong.

Gambar 9.22b Dokumentasi B03 - Aura! - Urutkan Data Kosong

B05 - Dead or Alive

- **Deskripsi** Fungsi ini memeriksa status nyawa pengguna dan mengeluarkan pengguna dari sistem jika nyawa habis.
- **Testing**
  - Memeriksa pengguna dengan nyawa lebih dari 0.

Gambar 9.23a Dokumentasi B05 - Dead or Alive - Nyawa Lebih Dari 0

- Memeriksa pengguna dengan nyawa 0.

Gambar 9.23b Dokumentasi B05 - Dead or Alive - Nyawa 0

B06 - Mainin Antrian

- **Deskripsi** Fungsi ini memanipulasi antrian pasien, misalnya memanggil pasien ke ruangan dokter.
- **Testing**
  - Memanggil pasien dari antrian yang tidak kosong.

Gambar 9.24a Dokumentasi B06 - Mainin Antrian - Panggil Pasien Valid

- Memanggil pasien dari antrian yang kosong.

Gambar 9.24b Dokumentasi B06 - Mainin Antrian - Panggil Pasien Kosong

# Lampiran

**Form MoM Asistensi Tugas Besar  
IF1210/Algoritma dan Pemrograman 1  
Sem. 2 2024/2025**

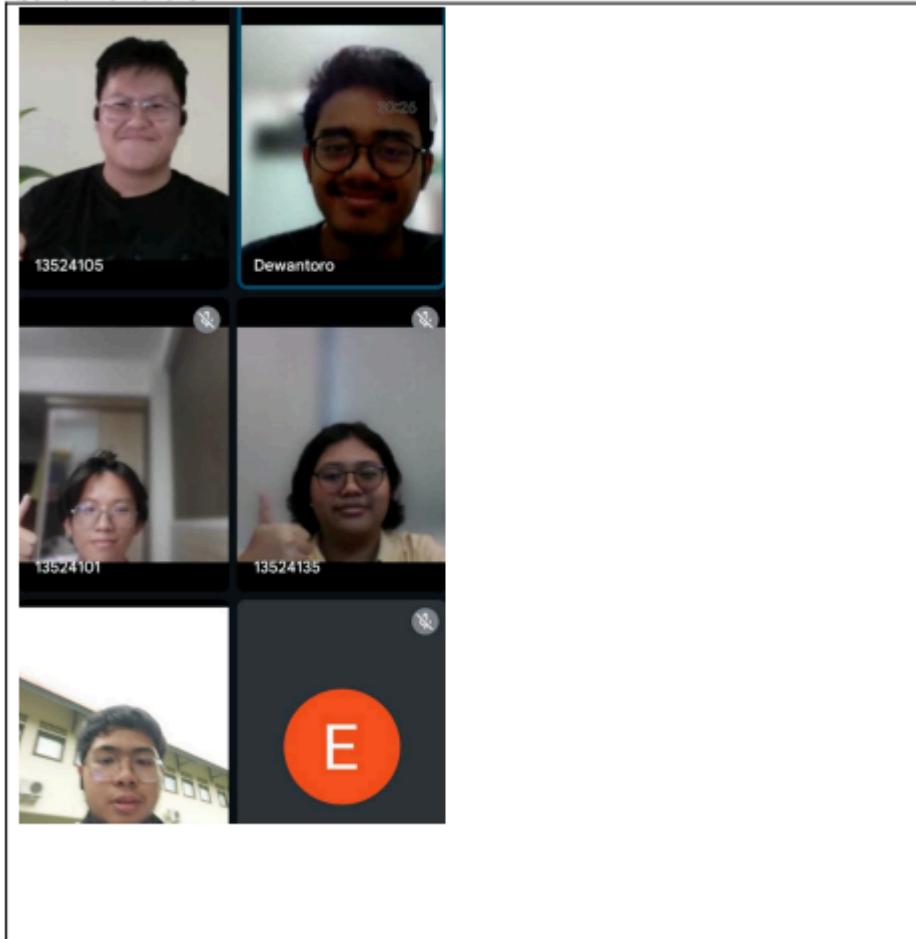
Nomor Asistensi	:	1
No. Kelompok/Kelas	:	K/K01
Tanggal asistensi	:	6 Mei 2025
Anggota kelompok		
NIM / Nama (Hanya yang Hadir)		
1	13524003 / Faiq Azzam Nafidz	
2	13524101 / Philipp Hamara	
3	13524105 / Nicholas Luis Chandra	
4	13524115 / Ega Luthfi Rais	
5	13524135 / Varistha Devi	
6		
Asisten pembimbing		
NIM / Nama		
13522011 / Dewantoro Triatmojo		

**Catatan Asistensi:**

Rangkuman Diskusi
-Klarifikasi beberapa spesifikasi program yang ambigu seperti akses dari F04 - Lupa Password, kondisi dan keberadaan dari file-file data seperti user.csv saat program pertama dijalankan -Saran mengenai penggunaan WSL untuk memastikan program dapat berjalan -Saran mengenai penggunaan ADT-ADT tambahan -Parsing file-file data csv secara general dengan penggunaan library <stdarg.h>
Tindak Lanjut
- Pembuatan ADT untuk setiap file .csv - Pembuatan fungsi yang mengambil data dari file .csv (fungsi parsing secara general), untuk dimasukkan ke list of tipe data (jika user.csv, berarti list of User, jika obat.csv berarti list of Obat, dst) - Pembuatan fungsi initialize_program() yang isinya ada fungsi yang mengambilkan data dari file - Pembuatan ADT-ADT tambahan (pasien, dokter, dan sebagainya bila diperlukan) - Pembuatan fungsi yang dapat mengambil/merujuk (disarankan pakai pointer) data yang ada di data csv ke ADT tambahan
Dokumentasi

Gambar 10.1a Hasil Pindai Form Asistensi 1 (Halaman Pertama)

**Form MoM Asistensi Tugas Besar**  
**IF1210/Algoritma dan Pemrograman 1**  
**Sem. 2 2024/2025**



Gambar 10.1b Hasil Pindai Form Asistensi 1 (Halaman Kedua)

**Form MoM Asistensi Tugas Besar**  
**IF1210/Algoritma dan Pemrograman 1**  
**Sem. 2 2024/2025**

Nomor Asistensi : 2  
No. Kelompok/Kelas : K/K01  
Tanggal asistensi : 23 Mei 2025

Anggota kelompok	NIM / Nama (Hanya yang Hadir)
1	13524101 / Philipp Hamara
2	13524105 / Nicholas Luis Chandra
3	13524115 / Ega Luthfi Rais
4	13524135 / Varistha Devi
5	
6	

Asisten pembimbing	NIM / Nama
	13522011 / Dewantoro Triatmojo

**Catatan Asistensi:**

Rangkuman Diskusi
-Klarifikasi mengenai isi file user.csv dan tentang kemungkinan keacakannya -Klarifikasi mengenai mungkin spasi di dalam nama username -Save data boleh disimpan terurut apabila data awal acak -Diagnosis dan ngobatin dokter hanya kepada yang di dalam ruangan -Penerimaan input menggunakan scanf() tidak masalah
Tindak Lanjut
- Revisi penerimaan input ADT List userdata, obat, penyakit dan disortir terlebih dahulu - Pembuatan fungsi cari{User/Obat/Penyakit}Idx yang mengembalikan indeks dimana id user/obat/penyakit berada di dalam list - Revisi config setelah spesifikasi direvisi
Dokumentasi

Gambar 10.2 Hasil Pindai Form Asistensi 2

Link Repository Github: <https://github.com/orgs/Labpro-22/teams/k01-k>