

RUMAH SAKIT PARA NIMONS

LAPORAN TUGAS BESAR



Kelompok M

Anggota Kelompok:

Jose Luis Fernando Saragi	NIM 13224013
Muhammad Zaki Azzamy Syauqi	NIM 13224045
Luis Matthew Sembiring	NIM 13224053
Muhammad Aqeel Ghani	NIM 13224071
Nahidl Denhaq Syaha	NIM 13224089
Gabrielle Garcia Hardanta P	NIM 18324015

IF1210 – ALGORITMA DAN PEMROGRAMAN – K05

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA - REKAYASA

INSTITUT TEKNOLOGI BANDUNG

APRIL 2025

PERNYATAAN KELOMPOK

"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."

30 Mei 2025

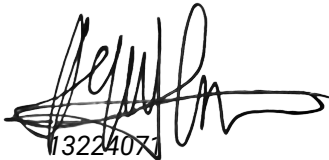


13224045

Muhammad Zaki Azzamy Syauqi

"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."

30 Mei 2025



13224071

Muhammad Aqeel Ghani

"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."

30 Mei 2025



13224053

Luis Matthew Sembiring

"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."

30 Mei 2025

18324015

Gabrielle Garcia Hardanta Prakoso

"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."

30 Mei 2025

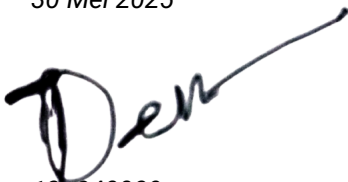


13224013

Jose Luis Fernando Saragi

"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."

30 Mei 2025



132240089

Nahidl Denhaq Syaha

DAFTAR ISI

PERNYATAAN KELOMPOK.....	2
DAFTAR ISI.....	4
DAFTAR TABEL.....	7
DAFTAR GAMBAR.....	8
DESKRIPSI PERSOALAN.....	9
RENCANA IMPLEMENTASI ADT.....	9
DAFTAR PEMBAGIAN KERJA ANGGOTA KELOMPOK.....	15
HASIL RANCANGAN, IMPLEMENTASI, DAN TESTING SETIAP PRIMITIF.....	16
DESAIN DEKOMPOSISI ALGORITMIK DAN FUNSIONAL PROGRAM.....	17
DESAIN COMMAND SETIAP PRIMITIF.....	17
1. F01.....	17
2. F02.....	18
3. F03.....	18
4. F04.....	19
5. F05.....	19
6. F06.....	21
7. F07.....	22
8. F08.....	23
9. F09.....	24
10. F10.....	25
11. F11.....	25
12. F12.....	26
13. F13.....	26
14. F14.....	27
15. F15.....	29
16. F16.....	29
17. F17.....	30
18. F18 (D03, D04).....	30
19. B02.....	31
DESAIN KAMUS DATA.....	32
1. F00 (ADT dan Struct).....	32
2. F01.....	41
3. F02.....	41
4. F03.....	42
5. F04.....	42
6. F05.....	42
7. F06.....	43
8. F07.....	43
9. F08.....	44
10. F09.....	45
11. F10.....	45
12. F11.....	45

13. F12.....	46
14. F13.....	46
15. F14.....	46
16. F15.....	46
17. F16.....	47
18. F17.....	47
19. F18.....	48
DESAIN DEKOMPOSISI ALGORITMIK DAN FUNGSIONAL PROGRAM.....	48
NOTASI ALGORITMIK SETIAP FUNGSI DAN PROSEDUR UTAMA.....	48
1. F01.....	48
2. F02.....	49
3. F03.....	49
4. F04.....	50
5. F05.....	51
6. F06.....	56
7. F07.....	58
8. F08.....	62
9. F09.....	67
10. F10.....	68
11. F11.....	69
12. F12.....	70
13. F13.....	71
14. F14.....	73
15. F15.....	75
16. F16.....	76
17. F17.....	77
18. F18.....	78
19. D03.....	79
22. B02.....	85
SCREENSHOT FITUR.....	87
1. F01.....	87
2. F02.....	91
3. F03.....	91
4. F04.....	91
5. F05.....	92
6. F06.....	93
7. F07.....	95
8. F08.....	108
9. F09.....	111
10. F10.....	112
11. F11.....	113
12. F12.....	114
13. F13.....	114
14. F14.....	115

15. F15.....	116
16. F16.....	117
17. F17.....	118
18. F18.....	118
19. D03.....	118
20. D04.....	118
21. B02.....	120
22. B06.....	121
LAMPIRAN.....	123

DAFTAR TABEL

Tabel 1 – Rencana Implementasi ADT.....	4
Tabel 2 – Daftar Pembagian Kerja Anggota Kelompok.....	4
Tabel 3 – Hasil Rancangan, Implementasi, dan Testing Setiap Primitif.....	4

DAFTAR GAMBAR

Gambar 1 – Rencana Implementasi ADT.....	14
Gambar 2 - Desain Dekomposisi Algoritmik dan Fungsional Program.....	17

DESKRIPSI PERSOALAN

Gro kini hidup normal bersama keluarganya, namun rumahnya selalu kacau karena ulah para Nimon. Dr. Neroifa, ilmuwan Gro, merasa jenuh dengan rutinitas dan melihat para Nimon sering mengalami insiden medis kecil yang ditangani seadanya. Ia pun berinisiatif membangun Rumah Sakit Nimon yang canggih.

Masalah muncul saat Dr. Neroifa sadar ia tidak mampu membuat sistem manajemen untuk rumah sakit tersebut. Pengelolaan pasien, dokter, antrian, stok obat, dan data medis menjadi kacau. Ia membutuhkan bantuan untuk merancang jantung digital rumah sakit ini.

Di sinilah peran kami, mahasiswa IF1210. Tugas kami adalah mengubah kekacauan ini menjadi sistem manajemen rumah sakit yang terorganisir, memastikan setiap Nimon mendapat perawatan, dan membantu Dr. Neroifa mewujudkan visinya.

Kami akan membangun sistem manajemen Rumah Sakit Nimon. Proyek ini dimulai dengan F00 - Rencana Implementasi. Sistem akan memiliki fitur dasar seperti F01 - Login, F02 - Register Pasien (dengan validasi keunikan username menggunakan Set), F03 - Logout, F04 - Lupa Password (validasi dengan Run-Length Encoding), dan F05 - Menu & Help.

Kami akan mengimplementasikan F06 - Denah Rumah Sakit dan F09 - Lihat Antrian, yang detailnya akan disesuaikan. Untuk manajemen pengguna dan data, kami akan membuat F07 - Lihat User, F08 - Cari User (menggunakan binary search untuk ID dan sequential search untuk nama), dan F10 - Tambah Dokter beserta fungsi assign ruangan. Dokter dapat melakukan F11 - Diagnosis otomatis berdasarkan data pasien dan file penyakit.csv, serta F12 - Ngobatin sesuai urutan dari file obat_penyakit.csv.

Pasien akan memiliki fitur F13 - Aku boleh pulang ga, dok?, F14 - Daftar Check-Up, F15 - Antrian Saya!, F16 - Minum Obat (disimpan dalam struktur data stack "perut"), dan F17 - Minum Penawar. Program akan ditutup dengan F18 - Exit, yang juga akan mencakup mekanisme penyimpanan dan pemuatan data jika diperlukan oleh spesifikasi.

RENCANA IMPLEMENTASI ADT

Untuk menyelesaikan persoalan tersebut, kami membuat beberapa ADT yang akan digunakan dalam algoritma sehingga terbuatnya suatu program yang utuh. ADT-ADT tersebut dapat dilihat di Tabel 1.

Implementasi ADT	FITUR	Deskripsi Implementasi	Alasan Implementasi
------------------	-------	------------------------	---------------------

ADT List Dinamik Eksplisit Rata Kiri	<p>Secara langsung digunakan : F01 (login), F02 (register pasien), F07 (lihat user), F08 (cari user), F10 (tambah dokter), D03 (load), D04 (save).</p> <p>Secara tidak langsung digunakan (elemen dalam list digunakan /diganti) Seluruh fungsi lainnya.</p>	<p>Digunakan sebagai database user-user (pasien, dokter, manager). ADT ini memiliki komponen integer <i>size</i> dan kapasitas, serta <i>pointer</i> ke struct GenericData (<i>array</i> dari GenericData).</p>	<p>List dinamik paling mudah dan efisien untuk penambahan elemen dan ekspansi kapasitas list, dengan kompleksitas waktu $O(1)$. <i>Size</i> array dapat bertambah berkali-kali hingga teoritis tak hingga akibat dari F02 (register pasien) dan F10 (tambah pasien)</p> <p>List dinamik memudahkan iterasi (dengan index) sehingga algoritma searching seperti <i>binary search</i> dengan kompleksitas waktu $O(\log n)$ dapat dilakukan (setelah diurutkan) untuk F07 (lihat user) dan F08 (cari user). Jika sudah diketahui indeks elemen yang diinginkan, maka $O(1)$ dalam mendapatkan elemennya. Kemudahan iterasi juga memudahkan proses load dan save. Tambah user selalu di akhir karena id harus unik dan mendapatkan id (id terakhir + 1), sehingga insertion tidak perlu dipertimbangkan. Penghapusan elemen dilakukan sekali saja dalam program (B05 <i>Dead or Alive?!)</i>, Karena array harus rata-kiri, harus melakukan elemen shift ke rata kiri, sehingga ketika harus delete suatu elemen, kompleksitas waktu adalah $O(n)$. Namun karena delete element hanya dilakukan sekali dalam jalannya program (B05 <i>Dead or Alive?!)</i>, maka waktu tersebut tidak signifikan.</p>
ADT Map	F12 (ngobatin) , F14 (daftar check-up)	<p>Matching antara penyakit sebagai key dan obat-obat sebagai value, serta dokter id sebagai key dan antrian sebagai value. ADT Map tidak dibuat secara eksplisit, namun digunakan secara implisit dalam pembuatan struct</p>	<p>Kemudahan mendapatkan informasi value berdasarkan key. ADT Map tidak dibuat secara eksplisit dikarenakan kesederhanaannya, sehingga kami memilih untuk tidak perlu membuat ADT-nya secara langsung dan fungsi implementasinya, sehingga sebenarnya ADT Map tidak bersifat ADT secara definisi yang memiliki fungsi primitif. Hal ini dikarenakan kesederhanaannya dan</p>

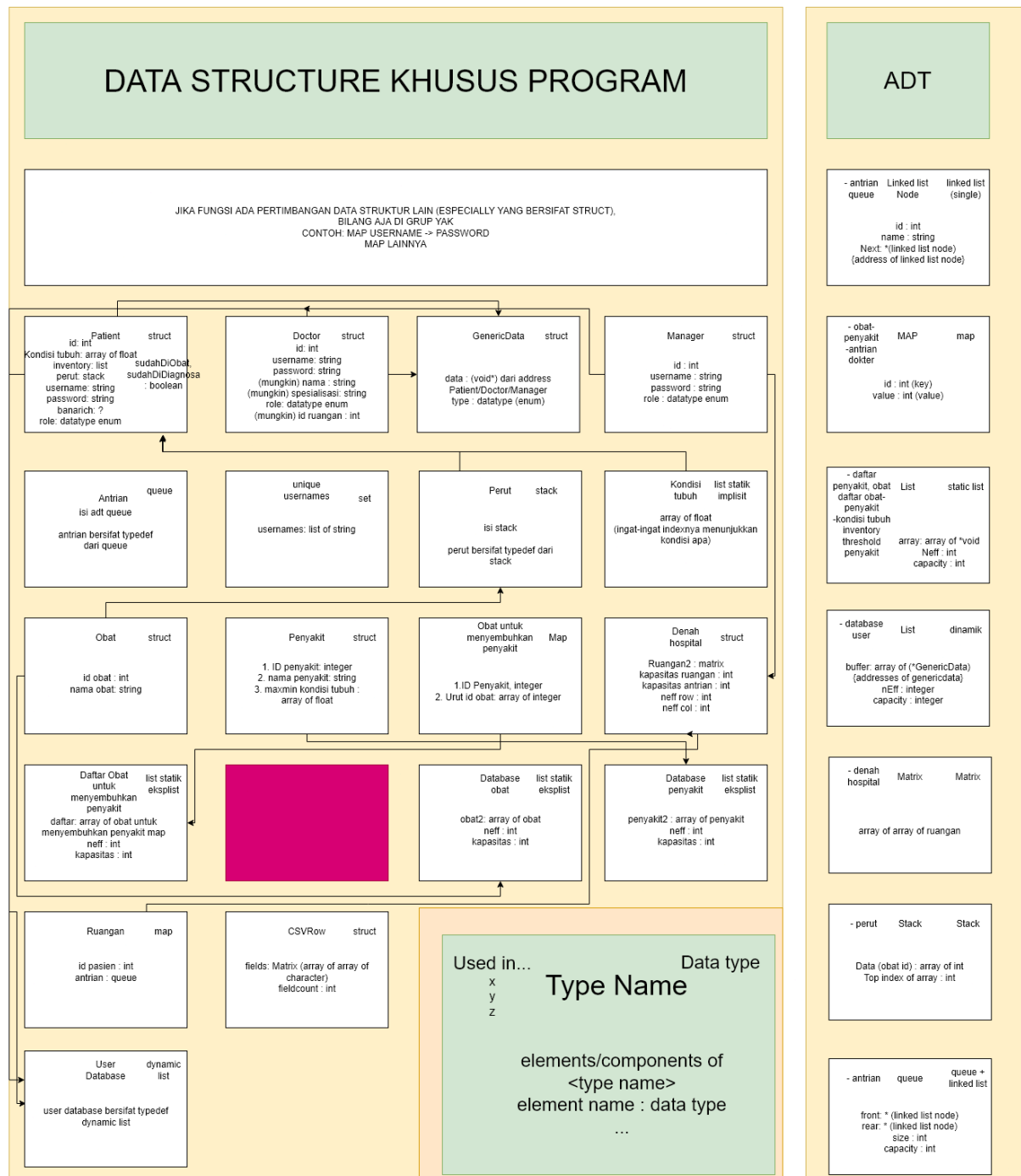
		relasi obat-penyakit dan dokter-antrian.	pertimbangan waktu.
ADT Stack	F16 (minum obat), F17 (minum penawar)	Digunakan untuk perut pasien. Stack tidak dibuat secara generik namun dibuat khusus untuk perut. Maka dari itu, komponen dari stack adalah <i>array</i> dari obat dan integer indeks top. <i>Array</i> menggunakan list statik implisit rata kiri, dengan makro UNDEF_INT_DATA sebagai MARK.	Stack dibuat khusus perut karena tidak ada bagian lain selain perut yang menggunakan stack. Minum obat dan minum penawar (mengeluarkan obat terakhir diminum) menggunakan konsep <i>last in first out</i> (LIFO), oleh karena itu digunakan stack.
ADT Queue dengan Linked List (list berkaitan)	F06 (denah rumah sakit), F09 (lihat antrian), F11 (diagnosis), F12 (ngobatin), F13 (aku boleh pulang ga dok), F14 (daftar check-up), F15 (antrian saya), D01 (denah rumah sakit), D02 (lihat antrian), D03 (load), D04 (save), B02 (denah	Digunakan untuk antrian ruangan dan luar ruangan. Queue tidak dibuat secara generik namun dibuat khusus untuk antrian tersebut. Maka dari itu, komponen dari queue adalah <i>pointer</i> ke awal dan akhir list berkaitan, integer <i>size</i> dan kapasitas. List berkaitan dibuat secara linear.	Queue dibuat khusus untuk antrian, karena tidak ada bagian lain selain antrian yang menggunakan antrian. Antrian menggunakan konsep <i>first in first out</i> (FIFO), oleh karena itu digunakan queue. Queue menggunakan list berkaitan untuk memudahkan penambahan (<i>enqueue</i>) atau pengurangan (<i>dequeue</i>), karena list berkaitan tidak mengalokasi tempat untuk list secara eksplisit, tetapi secara acak, sehingga penambahan dan pengurangan dapat dilakukan dengan cepat ($O(1)$). Selain itu, list berkaitan memudahkan penggantian elemen dalam list berkaitan, sehingga pertukaran pasien dalam antrian pada B06 (mainin antrian) dapat dilakukan dengan cepat ($O(1)$).

	dinamis), B06 (mainin antrian).		
ADT Matrix	<p>Secara langsung digunakan : F06 (denah rumah sakit), F09 (lihat antrian), F10 (tambah dokter), F14 (daftar check-up), F15 (antrian saya), D03 (load), D04 (save)</p> <p>Secara tidak langsung digunakan (elemen dalam matrix digunakan /diganti): F11 (diagnosis), F12 (ngobatin), F13 (aku boleh pulang ga, dok), F16 (minum obat), F17 (minum penawar)</p>	<p>Digunakan untuk denah rumah sakit. Matrix tidak dibuat secara eksplisit, namun digunakan secara implisit dalam pembuatan struct <code>DataTypeDenah</code>. Matrix terbentuk sebuah <i>array</i> dari <i>array</i> dari ruangan, dengan <i>array</i> berbentuk list statik rata kiri atas dan eksplisit.</p>	<p>Matrix dibuat khusus untuk denah, karena tidak ada bagian lain selain denah yang menggunakan matrix. Denah terbuat seperti sebuah <i>grid</i>, sehingga matrix cocok untuk denah. Matrix menggunakan list statik rata kiri eksplisit karena kapasitas denah bersifat statik (26 x 26 karena hanya terdapat 26 huruf dalam alfabet). Matrix tidak dibuat secara eksplisit karena alasan tersebut dan juga karena menggunakannya cukup sederhana tanpa harus membuat fungsi implementasi, sehingga sebenarnya Matrix dalam program kami tidak ADT secara definisi yang memiliki fungsi primitif. Fungsi implementasi tidak digunakan karena kesederhanaan tersebut dan pertimbangan waktu.</p>

ADT List Statik Eksplisit Rata Kiri	Seluruh fungsi	Digunakan untuk database yang bersifat statik, yaitu database obat, database penyakit, dan database obat-penyakit. ADT ini tidak dibuat secara eksplisit, namun digunakan konsepnya.	Database-database yang statik (tidak berubah ukuran) tidak perlu memperhatikan kapasitas dari sebuah <i>array</i> karena list statik dapat dideklarasikan dengan kapasitas yang dianggap pasti lebih besar daripada <i>size</i> database. Selain itu, list statik eksplisit rata kiri digunakan karena mudahnya pembuatan sebuah list statik eksplisit rata kiri dan dapat digunakan tanpa fungsi primitif-primitifnya (sehingga tidak membuat sebuah ADT secara definisi yang memiliki fungsi primitif dan dideklarasikan) agar tidak memakan waktu.
ADT List Statik Implisit Rata Kiri	Seluruh fungsi	Digunakan untuk struct yang memiliki sebuah <i>array</i> di dalamnya, dimana <i>array</i> tersebut perlu menunjukkan perbedaan antara nilai <i>valid</i> dan <i>invalid</i> , dan <i>array</i> tersebut bertujuan untuk hal yang sederhana. Contohnya untuk kondisi tubuh pasien dan <i>threshold</i> penyakit. ADT ini tidak dibuat secara eksplisit, namun digunakan konsepnya.	Kemudahan membuat list statik implisit rata kiri, dan <i>array-array</i> yang bertujuan untuk hal sederhana hanya perlu komponen <i>array</i> dalam suatu struct. Tidak digunakan eksplisit agar struct tidak mempunyai komponen terlalu banyak. ADT dan fungsi primitifnya tidak dibuat secara eksplisit, sehingga tidak menjadi sebuah ADT secara definisi dengan fungsi primitifnya, karena pertimbangan kesederhanaan dan waktu.
ADT List Statik Secara Umum	Seluruh fungsi	Digunakan untuk keperluan fungsi masing-masing jika membutuhkan <i>array</i> dalam algoritma fungsi. ADT ini tidak dibuat secara eksplisit, namun digunakan	Kemudahan membuat list statik dengan cepat untuk keperluan fungsi masing-masing yang memiliki tujuan dan tipe data elemen yang beragam. Oleh karena itu, tidak dibuat sebuah ADT secara eksplisit dengan fungsi implementasinya, juga karena pertimbangan waktu.

		konsepnya.	
--	--	------------	--

Tabel 1 – Rencana Implementasi ADT



Gambar 1 – Rencana Implementasi ADT

DAFTAR PEMBAGIAN KERJA ANGGOTA KELOMPOK

Fitur	Implementasi	NIM Desainer	NIM Coder	NIM Tester
F00, D03 (Read CSV), D04 (Write CSV), F18, diagram ADT	Merencanakan seluruh ADT dan struct yang digunakan kecuali ADT yang berkaitan dengan denah dan ruangan. Untuk ADT yang memerlukan, juga membuat fungsi implementasinya. Fungsi load dan save untuk file-file csv.	13224045	13224045	13224045
F01, F02, F03, F04, F10	Fungsi Login, Register, Logout, Lupa Password, Tambah Dokter	13224053	13224053 13224071	13224071
F05, F06, F09, D01, D02, D03 (Read Config), D04(Write Config), B02, B06, BXX, <i>flowchart</i> program	Fungsi Menu dan Help, Fungsi Lihat Denah, Fungsi Lihat Antrian, Denah Dinamis, Mainin Antrian, Kreatifitas berupa ASCII ART dan easter egg (hehehehe)	13224071	13224071	13224071
F07, F08, F14	Fungsi Lihat User, Fungsi Cari User, dan Fungsi Daftar Check-up	13224089	13224089	13224089 13224071
F15, F16, F17	Lihat antrian(queue), minum obat(queue dan stack), minum penawar(stack dan queue)	13224013	13224013	13224071 (f15) 13224045 (f16 & f17)
F11,F12,F13	Fungsi Diagnosa, Fungsi Ngobatin, Fungsi Pulang Dok	18324015	18324015 13224045	18324015 13224045

Tabel 2 – Daftar Pembagian Kerja Anggota Kelompok

HASIL RANCANGAN, IMPLEMENTASI, DAN TESTING SETIAP PRIMITIF

Fitur	Desain	Implementasi	Testing
F01- Login	V	V	V
F02 - Register	V	V	V
F03 - Logout	V	V	V
F04 - Lupa Password	V	V	V
F05 - Menu & Help	V	V	V
F06 - Denah RS	V	V	V
F07 - Lihat User	V	V	V
F08 - Cari User	V	V	V
F09 - Lihat Antrian	V	V	V
F10 - Tambah Dokter	V	V	V
F11 - Diagnosis	V	V	V
F12 - Ngobatin	V	V	V
F13 - Aku boleh pulang ga dok 🙄?	V	V	V
F14 - Daftar Check-up	V	V	V
F15 - Antrian Saya	V	V	V
F16 - Minum Obat	V	V	V
F17 - Minum Penawar	V	V	V
F18 - Exit	V	V	V
D01 - Denah	V	V	V

SELAMAT DATANG PASIEN gro !
<pre># Login sebagai Dokter >>> LOGIN Masukkan username: neronimo Masukkan password: pass1010 SELAMAT DATANG DOKTER neronimo!</pre>
<pre># Login sebagai Manager >>> LOGIN Masukkan username: zeru Masukkan password: pass77 SELAMAT DATANG PASIEN zeru!</pre>
<pre># Login dengan username atau password salah >>> LOGIN Masukkan username: neronimo Masukkan password: pass31 USERNAME ATAU PASSWORD SALAH. SILAHKAN COBA LAGI.</pre>
<pre># Login saat user sudah login >>> LOGIN ANDA SUDAH LOGIN. MOHON LOGOUT TERLEBIH DAHULU UNTUK MELAKUKAN LOGIN KEMBALI.</pre>

2. F02

<pre># Registrasi >>> REGISTER Username: tungsahur Password: pass100 Pasien tungsahur berhasil ditambahkan!</pre>
<pre># Registrasi akun jika sudah tersedia >>> REGISTER Username: zeru Password: zerucakep7 Registrasi gagal! Pasien dengan nama zeru sudah terdaftar.</pre>

3. F03

<pre># Logout dari akun >>> LOGOUT # Balik ke mainmenu belum login Sampai jumpa</pre>
<pre># Logout saat berada di mainmenu belum login >>> LOGOUT Logout gagal!</pre>

Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout.

4. F04

User pasien/dokter/manager ada dan kode unik benar

>>> **LUPA_PASSWORD**

Masukkan username: **memoryyy**

Kode Unik: **me2mor3y**

Halo memoryyy, silakan daftarkan ulang password anda!

Masukkan password baru: **pass1234**

Password berhasil diperbarui.

User pasien/dokter/manager ada tetapi kode unik salah

>>> **LUPA_PASSWORD**

Masukkan username: **memoryyy**

Kode Unik: **me5mor3y**

Kode unik salah!

User pasien/dokter/manager ada tetapi kode unik salah

>>> **LUPA_PASSWORD**

Masukkan username: **mamaAkuMasukTV**

Kode Unik: **mamaAkuMasukTV**

Username tidak ditemukan.

5. F05

User pasien

>>> **HELP**

Terimakasih telah memanggil fungsi Help

Berikut merupakan fungsi-fungsi yang dapat anda gunakan

1) **HELP** : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya

2) **LIHAT_DENAH** : Memunculkan denah rumah sakit

3) **LIHAT_RUANGAN XX** : Memunculkan detail ruangan XX (XX: kode ruangan)

4) **PULANGDOK** : Bertanya ke dokter apakah kamu sudah boleh pulang

5) **DAFTAR_CHECKUP** : Mendaftarkan check-up dengan dokter

6) **ANTRIAN** : Menunjukan status antrian pasien

7) **MINUM_OBAT** : Meminum obat yang berada di inventory

8) **PENAWAR** : Meminum penawar untuk memuntahkan obat yang berada di perut

9) **LOGOUT** : Keluar dari akun yang sedang digunakan

10) **EXIT** : Keluar dari program

Footnote:

1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

2) Jangan lupa untuk memasukkan input yang valid

User dokter

>>> **HELP**

Terimakasih telah memanggil fungsi Help

Berikut merupakan fungsi-fungsi yang dapat anda gunakan

- 1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya
- 2) LIHAT_DENAH : Memunculkan denah rumah sakit
- 3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode ruangan)
- 4) DIAGNOSIS : Mendiagnosis pasien yang berada di depan antrian
- 5) NGOBATIN : Mengobati pasien yang berada di depan antrian
- 6) LOGOUT : Keluar dari akun yang sedang digunakan
- 7) EXIT : Keluar dari program

Footnote:

- 1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
- 2) Jangan lupa untuk memasukkan input yang valid

User manager

>>> HELP

Terimakasih telah memanggil fungsi Help

Berikut merupakan fungsi-fungsi yang dapat anda gunakan

- 1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya
- 2) LIHAT_DENAH : Memunculkan denah rumah sakit
- 3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode ruangan)
- 4) LIHAT_USER : Melihat data seluruh pengguna
- 5) LIHAT_PASIEN : Melihat data seluruh pasien
- 6) LIHAT_DOKTER : Melihat data seluruh dokter
- 7) CARI_USER : Mencari data pengguna secara spesifik berdasarkan ID atau Nama
- 8) CARI_PASIEN : Mencari data pengguna secara spesifik berdasarkan ID, Nama, atau Penyakit
- 9) CARI_DOKTER : Mencari data pengguna secara spesifik berdasarkan ID, atau Nama
- 10) LIHAT_SEMUA_ANTRIAN : Melihat rincian di seluruh ruangan saat ini
- 11) TAMBAH_DOKTER : Menambahkan dokter baru
- 12) ASSIGN_DOKTER : Melakukan assign ruangan ke dokter tertentu yang belum memiliki ruangan
- 13) PINDAH_DOKTER XX YY: Melakukan pemindahan dokter dari ruangan XX ke YY (XX, YY: kode ruangan)
- 14) UBAH_DENAH X Y : Mengubah ukuran denah menjadi Y X (Y: Jumlah barisan, X : Jumlah kolom)
- 15) LOGOUT : Keluar dari akun yang sedang digunakan
- 16) EXIT : Keluar dari program

Footnote:

- 1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
- 2) Jangan lupa untuk memasukkan input yang valid

User belum login

>>> HELP

Terimakasih telah memanggil fungsi Help

Berikut merupakan fungsi-fungsi yang dapat anda gunakan

- 1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya

- 2) LOGIN : Masuki suatu akun
- 3) LUPA_PASSWORD : Mengganti atau memperbarui password akun
- 4) REGISTER : Membuat akun baru
- 5) EXIT : Keluar dari program

Footnote:

- 1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
- 2) Jangan lupa untuk memasukkan input yang valid

6. F06

```
# Fungsi Lihat Denah
```

```
>>> LIHAT_DENAH
```

```

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

```
# Kasus 1: ruangan terdapat dokter dan pasien
```

```
>>> LIHAT_RUANGAN A1
```

```
--- Detail Ruangan A1 ---
```

```
Kapasitas : 3
```

```
Dokter : neronimo
```

```
Pasien di dalam ruangan :
```

1. gro
2. kebin
3. stewart

```
-----
```

```
# Kasus 2: ruangan terdapat dokter dan tidak ada pasien
```

```
>>> LIHAT_RUANGAN B3
```

```
--- Detail Ruangan B3 ---
```

```
Kapasitas : 3
```

```
Dokter : risol
```

```
Pasien di dalam ruangan :
```

```
Tidak ada pasien di dalam ruangan saat ini.
```

```
-----
```

```
# Kasus 3: ruangan tidak terdapat dokter.
```

```
>>> LIHAT_RUANGAN B2
```

```
-- Detail Ruangan B2 ---
```

```
Kapasitas : 3
```

```
Dokter : -
```

```
Pasien di dalam ruangan :
```

```
Tidak ada pasien di dalam ruangan saat ini.
```

```
-----
```

```
# Kasus 3: tidak ada ruangan dengan kode ruangan tersebut
```

```
>>> LIHAT_RUANGAN C9
```

Tidak ada ruangan dengan kode ruangan C9

7. F07

```
# Kasus 1: Melihat data user (bisa berarti dokter atau pasien)
>>> LIHAT_USER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

Urutan Sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data seluruh user berdasarkan ID terurut ascending...

ID	Nama	Role	Penyakit
1	stewart	Pasien	(Belum diperiksa dokter)
2	gro	Pasien	COVID-19
3	kebin	Pasien	(Belum diperiksa dokter)
6	nikeb	Pasien	(Belum diperiksa dokter)
11	ciciko	Dokter	-
12	cacako	Dokter	-
15	risol	Dokter	-
100	pendatang	Pasien	(Belum diperiksa dokter)

```
# Kasus 2: Spesifik melihat data pasien
```

```
>>> LIHAT_PASIEN
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2
```

Menampilkan data pasien berdasarkan nama terurut descending...

ID	Nama	Penyakit
1	stewart	(Belum diperiksa dokter)
100	pendatang	(Belum diperiksa dokter)
6	nikeb	(Belum diperiksa dokter)
3	kebin	(Belum diperiksa dokter)
2	gro	COVID-19

```
# Kasus 3: Spesifik melihat data dokter
```

```
>>> LIHAT_DOKTER
```

```

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

Urutan sort?
1. ID
2. Nama
>>> Pilihan: 1

Menampilkan data dokter berdasarkan nama terurut ascending...
+-----+-----+
| ID | Nama |
+-----+-----+
| 12 | cacako |
| 11 | ciciko |
| 15 | risol |
+-----+-----+

```

8. F08

```

# Kasus 1: Mencari data user (bisa berarti dokter atau pasien)
>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

>>> Masukkan nomor ID user: 6

Menampilkan pengguna dengan nomor ID 6...
+-----+-----+-----+-----+
| ID | Nama | Role | Penyakit |
+-----+-----+-----+-----+
| 6 | nikeb | Pasien | (Belum diperiksa dokter) |
+-----+-----+-----+-----+

```

```

# Kasus 2: User yang dicari tidak ditemukan
>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2

>>> Masukkan nama user: nahid

Tidak ditemukan pengguna dengan nama nahid!

```

```

# Kasus 3: Mencari data pasien
>>> CARI_PASIEN
Cari berdasarkan?
1. ID
2. Nama
3. Penyakit

```

```
>>> Pilihan: 3
```

```
>>> Masukkan nama penyakit: COVID-19
```

Menampilkan pasien dengan penyakit COVID-19!

ID	Nama	Penyakit
2	gro	COVID-19

```
# Kasus 4: Mencari data dokter
```

```
>>> CARI_DOKTER
```

Cari berdasarkan?

1. ID

2. Nama

```
>>> Pilihan: 2
```

```
Masukkan nama dokter: ciciko
```

Menampilkan dokter dengan nama ciciko!

ID	Nama
11	ciciko

9. F09

```
# Fungsi Lihat Semua Antrian
```

```
>>> LIHAT_SEMUA_ANTRIAN
```

	1	2	3
A	A1	A2	A3
B	B1	B2	B3

```
===== A1 =====
```

Kapasitas : 3

Dokter : neronimo

Pasien di dalam ruangan :

1. gro
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 saat ini.

===== A3 =====
Kapasitas : 3
Dokter    : cacako
Pasien di dalam ruangan :
    1. nikeb
Pasien di antrian:
    Tidak ada pasien di antrian saat ini.

===== B1 =====
Kapasitas : 3
Dokter    : kroket
Pasien di dalam ruangan :
    1. minonette
    2. tuart
Pasien di antrian:
    Tidak ada pasien di antrian saat ini.

===== B3 =====
Kapasitas : 3
Dokter    : risol
Pasien di dalam ruangan :
    Tidak ada pasien di dalam ruangan saat ini.
Pasien di antrian:
    Tidak ada pasien di antrian saat ini.

```

10. F10

```

# Tambah Dokter hanya bisa dilakukan oleh manager
# Tambah Dokter/ Register Dokter
>>> TAMBAH_DOKTER
Username: fluor
Password: gurtyo33
Dokter fluor berhasil ditambahkan!

# Tambah Dokter jika user sudah tersedia
>>> TAMBAH_DOKTER
Username: ciciko
Password: pass1111
Sudah ada dokter bernama ciciko!

```

11. F11

```

# KASUS 1: Dokter memiliki pasien yang perlu diperiksa dan cocok dengan
penyakit di database
>>> DIAGNOSIS

Budi terdiagnosa penyakit Influenza!

# KASUS 2: Antrian pasien sudah kosong dan tidak ada pasien yang perlu
diperiksa
>>> DIAGNOSIS

```

Tidak ada pasien untuk diperiksa!
<pre># KASUS 3: Pasien tidak terjangkit penyakit apapun setelah diperiksa. Jika terdapat kasus ini maka pasien diperbolehkan pulang. >>> DIAGNOSIS</pre> <p>Budi tidak terdiagnosis penyakit apapun!</p>
<pre># KASUS 4: Pasien sudah pernah diperiksa sebelumnya >>> DIAGNOSIS</pre> <p>Budi Telah Didiagnosa</p>
<pre># KASUS 5: Data pasien kosong (NULL) >>> DIAGNOSIS</pre> <p>Pasien tidak ditemukan</p>

12. F12

<pre># KASUS 1: Pasien belum didiagnosis, tidak boleh diobati >>> NGOBATIN</pre> <p>Pasien belum menerima diagnosis!</p>
<pre># KASUS 2: Pasien sudah pernah diobati sebelumnya >>> NGOBATIN</pre> <p>Pasien sudah diobatin!</p>
<pre># KASUS 3: Pasien telah didiagnosis, dan daftar obat untuk penyakit ditemukan >>> NGOBATIN</pre> <p>Dokter sedang mengobati pasien Budi Pasien memiliki penyakit Influenza Obat yang harus diberikan: 1. Oseltamivir 2. Vitamin C</p>
<pre># KASUS 4: Penyakit tidak ditemukan dalam database >>> NGOBATIN</pre> <p>Penyakit tidak ditemukan dalam database!</p>
<pre># KASUS 5: Tidak ada daftar obat untuk penyakit yang dimaksud >>> NGOBATIN</pre> <p>Tidak ada daftar obat untuk penyakit ini.</p>
<pre># KASUS 6: Antrian kosong, tidak ada pasien untuk diobati >>> NGOBATIN</pre> <p>Tidak ada pasien untuk diobatin!</p>

13. F13

<pre># KASUS 1: Pasien belum menerima diagnosis >>> PULANG_DOK</pre> <p>Kamu belum menerima diagnosis apapun dari dokter, jangan buru-buru pulang!</p>
<pre># KASUS 2: Pasien sudah diagnosis tapi belum menerima obat >>> PULANG_DOK</pre> <p>Kamu belum menerima obat dari dokter, jangan buru buru pulang!</p>
<pre># KASUS 3: Pasien belum mengonsumsi semua obat dari dokter >>> PULANG_DOK</pre> <p>Dokter sedang memeriksa keadaanmu... Masih ada obat yang belum kamu habiskan, minum semuanya dulu yukk!</p>
<pre># KASUS 4: Pasien sudah konsumsi semua obat, tapi urutannya salah >>> PULANG_DOK</pre> <p>Dokter sedang memeriksa keadaanmu... Maaf, tapi kamu masih belum bisa pulang! Urutan peminuman obat yang diharapkan: Paracetamol -> Amoxicillin</p> <p>Urutan obat yang kamu minum: Amoxicillin -> Paracetamol</p> <p>Silahkan kunjungi dokter untuk meminta penawar yang sesuai!</p>
<pre># KASUS 5: Pasien sudah diagnosis, dapat obat, minum obat dengan urutan benar >>> PULANG_DOK</pre> <p>Dokter sedang memeriksa keadaanmu...</p> <p>Selamat! Kamu sudah dinyatakan sembuh oleh dokter. Silahkan pulang dan semoga sehat selalu!</p>
<pre># KASUS 6: Data pasien tidak ditemukan (globalCurrentPatient == NULL) >>> PULANG_DOK</pre> <p>Pasien tidak ditemukan dalam database!</p>
<pre># KASUS 7: Nama penyakit tidak ditemukan dalam database penyakit >>> PULANG_DOK</pre> <p>Dokter sedang memeriksa keadaanmu... Penyakit tidak ditemukan dalam database!</p>
<pre># KASUS 8: Penyakit ada, tapi tidak ada data obat yang cocok di database OP >>> PULANG_DOK</pre> <p>Dokter sedang memeriksa keadaanmu... Data obat tidak ditemukan untuk penyakit ini!</p>

Kasus 1: Pasien berhasil mendaftar check-up

>>> **DAFTAR_CHECKUP**

Suhu tubuh (celecius): 35

Tekanan darah (sistol/diastol, contoh 120 80): 100 60

Detak jantung (bpm): 100

Saturasi oksigen: 95.5

Kadar gula darah (mg/dL): 80

Berat badan (kg): 67.5

Tinggi badan (cm): 173

Kadar kolestrol (mg/dL): 200

Trombosit (ribu/ μ L): 165000

Berikut adalah daftar dokter yang tersedia:

1. Dr. neronimo - Spesialisasi (-) - Ruangan A1 (Antrian: 2 orang)
2. Dr. ciciko - Spesialisai (-) - Ruangan A2 (Ruangan belum penuh)
3. Dr. cacako - Spesialisai (-) - Ruangan A3 (Ruangan belum penuh)
4. Dr. kroket - Spesialisai (-) - Ruangan B1 (Ruangan belum penuh)
5. Dr. risol - Spesialisai (-) - Ruangan B3 (Ruangan belum penuh)

Pilih dokter (1 - 5): 2

Pendaftaran check-up berhasil!

Anda terdaftar pada antrian Dr. ciciko di ruangan A2

Anda dapat langsung masuk ke dalam ruangan

spesialisasi masih '(-)' karena pada database belum terisi

Kasus 2: Pasien sudah terdaftar dalam antrian

>>> **DAFTAR_CHECKUP**

Anda sudah terdaftar dalam antrian check-up

Silahkan selesaikan check-up yang sudah terdaftar terlebih dahulu

Kasus 3: Input tidak valid

>>> **DAFTAR_CHECKUP**

Suhu tubuh (celecius): -1

Suhu tubuh harus berupa angka positif!

Suhu tubuh (celecius): 35.2

Tekanan darah (sistol/diastol, contoh 120 80): -1 -1

Tekanan darah harus berupa angka positif!

Tekanan darah (sistol/diastol, contoh 120 80): 60 100

Tekanan sistolik harus lebih besar dibanding diastolik!

Tekanan darah (sistol/diastol, contoh 120 80): 100 60

Detak jantung (bpm): -1

Detak jantung harus berupa angka positif!

Detak jantung (bpm): 100

Saturasi oksigen: 101

Saturasi oksigen harus dalam rentang 1 - 100!

Saturasi oksigen: 95.5

Kadar gula darah (mg/dL): -1

Kadar gula darah harus berupa angka positif!

Kadar gula darah (mg/dL): 80

Berat badan (kg): -1

Berat badan harus berupa angka positif!

Berat badan (kg): 67.5

Tinggi badan (cm): -1

Tinggi badan harus berupa angka positif!

Tinggi badan (cm): 173

```
Kadar kolesterol (mg/dL): -1
Kadar kolesterol harus berupa angka positif!
Kadar kolesterol (mg/dL): 200
Trombosit (ribu/ $\mu$ L): -1
Trombosit harus berupa angka positif!
Trombosit (ribu/ $\mu$ L): 165000
...(Selanjutnya sesuai dengan tampilan pada kasus 1)
```

15. F15

```
# KASUS 1: Pasien sudah terdaftar dalam antrian
>>> ANTRIAN
```

```
Status antrian Anda:
Dokter: Dr. Budi
Ruangan: A1
Posisi antrian: 3 dari 4
```

```
# KASUS 2: Pasien belum terdaftar dalam antrian
>>> ANTRIAN
```

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

```
# KASUS 3: Pasien sudah berada di ruangan
>>> ANTRIAN
```

```
Anda sedang berada di dalam ruangan dokter!
```

16. F16

```
>>> MINUM_OBAT
```

```
# Kasus 1: Apabila belum melakukan diagnosis (jika belum diagnosis otomatis belum diobatin)
Anda belum melakukan diagnosis. Lakukan diagnosis dengan perintah DIAGNOSIS.
```

```
# Kasus 2: Apabila sudah diagnosis tetapi belum diobatin
Anda belum meminta obat dari dokter. Minta obat dengan command NGOBATIN.
```

```
# Kasus 3: Sudah diagnosis dan sudah diobatin
#Kasus 3.1: Semua obat sudah diminum tetapi masih menggunakan command ini
Inventorynya udah kosong. Minum penawar dengan command PENAWAR jika perlu
mengulang minum obat.
```

```
#Kasus 3.2: Obat masih ada
===== DAFTAR OBAT =====
1. Paracetamol
2. Amoxicillin
3. Panadol
```

```
# Kasus 3.2.1: Apabila obat yang dipilih tidak ada
>>> Pilih obat untuk diminum: 4
```

```
Pilihan nomor tidak tersedia!
```

```
# Kasus 3.2.2: Apabila obat yang dipilih ada
Pilih obat untuk diminum: 1

GLEKGLEKGLEK... Paracetamol berhasil diminum!!!
```

```
# Tampilan daftar obat setelah paracetamol diminum

>>> MINUM_OBAT

===== DAFTAR OBAT =====
1. Amoxicillin
2. Panadol

# dan seterusnya
```

17. F17

```
# Kasus 1: Obat terakhir yang diminum adalah Paracetamol
>>> PENAWAR
Uwekkk!!! Paracetamol keluar dan kembali ke inventory
```

```
# Kasus 2: Belum ada obat yang diminum
>>> PENAWAR
Perut kosong, belum ada obat yang diminum.
```

18. F18 (D03, D04)

```
# Kasus 1: Input tidak valid
>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) a
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) y
```

```
# Kasus 2: Input y, folder sudah ada (overwrite isi folder)
>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) y
Masukkan nama folder (contoh: data/hari_ini): data/31-05-2025
SAVED USER DATABASE!
SAVED OBAT DATABASE!
SAVED PENYAKIT DATABASE!
SAVED OBAT PENYAKIT DATABASE!
Data berhasil disimpan di folder data/31-05-2025!
Sampai jumpa, Niemons!
```

```
# Kasus 3: Input y, folder tidak ada (buat isi folder)
>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n): y
Masukkan nama folder (contoh: data/hari_ini): data/32-05-2025
Membuat folder data/32-05-2025 ...
SAVED USER DATABASE!
SAVED OBAT DATABASE!
SAVED PENYAKIT DATABASE!
SAVED OBAT PENYAKIT DATABASE!
Data berhasil disimpan di folder data/32-05-2025!
Sampai jumpa, Niemons!
```

```
# Kasus 4: Input n
>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n): n
```

Sampai jumpa, Niemons!

19. B02

```
# Denah awal berukuran 2x3
```

>>> LIHAT_DENAH

	1	2	3
A	A1	A2	A3
B	B1	B2	B3

```
# Manajer ingin mengubah ukuran denah
```

>>> UBAH_DENAH 26 26

Denah rumah sakit berhasil diubah menjadi 26 baris dan 26 kolom.

>>> LIHAT_DENAH

	1	2	3	4	5	6		8	9	10	11	12	13	14	15	16	17	18	19	20	21		23	24	25	26
A	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26
B	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20	B21	B22	B23	B24	B25	B26
C	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	C26
D	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26
E	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15	E16	E17	E18	E19	E20	E21	E22	E23	E24	E25	E26
F	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25	F26
G	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15	G16	G17	G18	G19	G20	G21	G22	G23	G24	G25	G26
H	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13	H14	H15	H16	H17	H18	H19	H20	H21	H22	H23	H24	H25	H26
I	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12	I13	I14	I15	I16	I17	I18	I19	I20	I21	I22	I23	I24	I25	I26
J	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12	J13	J14	J15	J16	J17	J18	J19	J20	J21	J22	J23	J24	J25	J26
K	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15	K16	K17	K18	K19	K20	K21	K22	K23	K24	K25	K26
L	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15	L16	L17	L18	L19	L20	L21	L22	L23	L24	L25	L26
M	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20	M21	M22	M23	M24	M25	M26
N	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13	N14	N15	N16	N17	N18	N19	N20	N21	N22	N23	N24	N25	N26
O	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26
P	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26
Q	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26
R	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
S	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22	S23	S24	S25	S26
T	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25	T26
U	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26
V	V1	V2	V3	V4	V5	V6																				

```
# Manager mencoba mengecilkan denah ke 1x1, tapi masih terdapat dokter di
A2 dan A3
```

>>> UBAH DENAH 1 1

Tidak dapat mengubah ukuran denah. Ruangan A2 masih ditempati oleh Dr. ciciko. Silakan pindahkan dokter terlebih dahulu.

>>> LIHAT DENAH

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
A	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26
B	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20	B21	B22	B23	B24	B25	B26
C	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	C26
D	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26
E	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15	E16	E17	E18	E19	E20	E21	E22	E23	E24	E25	E26
F	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25	F26
G	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15	G16	G17	G18	G19	G20	G21	G22	G23	G24	G25	G26
H	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13	H14	H15	H16	H17	H18	H19	H20	H21	H22	H23	H24	H25	H26
I	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12	I13	I14	I15	I16	I17	I18	I19	I20	I21	I22	I23	I24	I25	I26

J	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12	J13	J14	J15	J16	J17	J18	J19	J20	J21	J22	J23	J24	J25	J26
K	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15	K16	K17	K18	K19	K20	K21	K22	K23	K24	K25	K26
L	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15	L16	L17	L18	L19	L20	L21	L22	L23	L24	L25	L26
M	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20	M21	M22	M23	M24	M25	M26
N	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13	N14	N15	N16	N17	N18	N19	N20	N21	N22	N23	N24	N25	N26
O	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26
P	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26
Q	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26
R	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
S	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22	S23	S24	S25	S26
T	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25	T26
U	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26
V	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26
W	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15	W16	W17	W18	W19	W20	W21	W22	W23	W24	W25	W26
X	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18	X19	X20	X21	X22	X23	X24	X25	X26
Y	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12	Y13	Y14	Y15	Y16	Y17	Y18	Y19	Y20	Y21	Y22	Y23	Y24	Y25	Y26
Z	Z1	Z2	Z3	Z4	Z5	Z6	Z7	Z8	Z9	Z10	Z11	Z12	Z13	Z14	Z15	Z16	Z17	Z18	Z19	Z20	Z21	Z22	Z23	Z24	Z25	Z26

Kasus pindah dokter berhasil
>>> **PINDAH_DOKTER A2 C4**
Dr. ciciko berhasil dipindahkan dari ruangan A2 ke ruangan C4.

Kasus ruangan tujuan sudah ditempati
>>> **PINDAH_DOKTER A3 C4**
Pemindahan gagal. Ruangan C4 Sudah ditempati.

Kasus ruangan asal kosong
>>> **PINDAH_DOKTER B4 A1**
Pemindahan gagal. Ruangan B4 Kosong.

20. B06

KASUS 1: Pasien (misal: popokan) sedang dalam antrian Dr. neronimo (posisi 3 dari 4) dan ingin skip # Dokter: Dr. Budi # Ruangan: B2 # Posisi antrian: 3 dari 4 >>> SKIP_ANTRIAN Anda berhasil maju ke depan antrian Dr. neronimo di ruangan B2! Posisi antrian Anda sekarang: 1 # Dokter: Dr. neronimo # Ruangan: B2 # Posisi antrian: 1 dari 4 (Pasien yang tadinya di posisi 1 dan 2 bergeser ke 2 dan 3)
KASUS 2: Pasien (misal: popokan) mencoba skip padahal sudah di depan # Dokter: Dr. neronimo # Ruangan: B2 # Posisi antrian: 1 dari 4 >>> SKIP_ANTRIAN

Anda sudah berada di posisi paling depan antrian! Tidak bisa skip lagi.

KASUS 3: Pasien (misal: gro) mencoba skip padahal sudah berda di dalam ruangan
Kondisi Awal: Pasien gro tidak terdaftar di antrian manapun

>>> SKIP_ANTRIAN

Anda sudah berada di dalam ruangan B2 bersama Dr. neronimo! Tidak bisa skip antrian lagi.

KASUS 4: Pasien (misal: tebin) mencoba skip padahal tidak sedang dalam antrian
Kondisi Awal: Pasien tebin tidak terdaftar di antrian manapun

>>> SKIP_ANTRIAN

Skip antrian gagal! Anda tidak sedang terdaftar dalam antrian manapun!

>>> CANCEL_ANTRIAN

Anda berhasil keluar dari antrian Dr. Budi di ruangan A1.

>>> CANCEL_ANTRIAN

Anda sudah berada di dalam ruangan B2 bersama Dr. neronimo! Tidak bisa cancel antrian.

>>> CANCEL_ANTRIAN

Cancel antrian gagal! Anda tidak sedang terdaftar dalam antrian manapun!

DESAIN KAMUS DATA

1. F00 (ADT dan Struct)

KAMUS STRUCTS DALAM HOSPITAL (USER DAN DATABASE)

{ Modul ADT Sederhana }
{ Berisi kumpulan tipe data dasar dan primitif untuk sistem rumah sakit }
{ Terdiri dari Pasien, Dokter, Manager, Obat, Penyakit, database-database mereka, dan entitas lainnya }

{ KONSTANTA }

const OBAT_MAX_SIZE : integer = 100
const IDX_UNDEF : integer = -1
const STR_MAX_SIZE : integer = 50
const PENYAKIT_MAX_SIZE : integer = 100
const OBAT_PENYAKIT_MAX_SIZE : integer = 10
const UNDEF_INT_DATA : integer = -99
const KONDISI_TUBUH_SIZE : integer = 10
const INVENTORY_SIZE : integer = 10
const THRESHOLD_SIZE : integer = 20

{ TIPE ENUMERASI }

type DataType : (

DATA_TYPE_UNKNOWN, { indikator tidak diketahui }
DATA_TYPE_PATIENT, { indikator data patient }
DATA_TYPE_DOCTOR, { indikator data doctor }

```

DATA_TYPE_MANAGER { indikator data manager }
)

{ TIPE BENTUKAN }
type_Obat : <
    id : integer,
    name : string[STR_MAX_SIZE]
>

type_Stack : <
    obat : array[0..OBAT_MAX_SIZE-1] of Obat,
    top : integer
>

type_Patient : <
    id : integer,
    username : string[STR_MAX_SIZE],
    password : string[STR_MAX_SIZE],
    type : DataType,
    riwayatPenyakit : string[STR_MAX_SIZE],
    perut : Stack,
    inventory : array[0..INVENTORY_SIZE-1] of integer,
    kondisiTubuh : array[0..KONDISI_TUBUH_SIZE-1] of real,
    sudahDiDiagnosis : boolean,
    sudahDiObatin : boolean
>

type_Doctor : <
    id : integer,
    username : string[STR_MAX_SIZE],
    password : string[STR_MAX_SIZE],
    type : DataType,
    name : string[STR_MAX_SIZE],
    spesialisasi : string[STR_MAX_SIZE]
>

type_Manager : <
    id : integer,
    username : string[STR_MAX_SIZE],
    password : string[STR_MAX_SIZE],
    type : DataType
>

type_Penyakit : <
    id : integer,
    name : string[STR_MAX_SIZE],
    threshold : array[0..THRESHOLD_SIZE-1] of real
>

type_ObatDatabase : <
    contents : array[0..OBAT_MAX_SIZE-1] of Obat,
    nEff : integer
>

type_PenyakitDatabase : <
    contents : array[0..PENYAKIT_MAX_SIZE-1] of Penyakit,
    nEff : integer
>

```

```

type_ObatPenyakit : <
    idPenyakit : integer,
    idObat : array[0..OBAT_PENYAKIT_MAX_SIZE-1] of integer,
    nEff : integer
>

type_ObatPenyakitDatabase : <
    contents : array[0..PENYAKIT_MAX_SIZE-1] of ObatPenyakit,
    nEff : integer
>

```

KAMUS STRUCTS DALAM HOSPITAL (DENAH)

```

{ KONSTANTA }
const MAX_CAPACITY : integer = 100

{ TIPE BENTUKAN }
type DataTypeRuangan : <
    idDokter : integer,           { id dokter }
    idAntrian : Queue           { Antrian Pasien }
>

type DataTypeDenah : <
    nRow : integer,              { Jumlah barisan }
    nColumn : integer,          { Jumlah kolom }
    kapasitasRuangan : integer, { Kapasitas Maksimal Pasien dalam
ruangan }
    Ruangan : array [0..25] of array[0..25] of DataTypeRuangan, {
Matriks denah }
    kapasitasAntrian : integer { Kapasitas Maksimal antrian }
>

type Point : <
    row : integer,              { Posisi baris }
    column : integer,          { Posisi kolom }
    antrian : integer { Nomor antrian }
>

```

KAMUS ADT LIST DINAMIK

```

{ Modul Dynamic List Eksplisit Rata Kiri untuk Database Logik User }
{ Berisi implementasi list dinamis untuk menyimpan data anggota rumah
sakit }
{ (Pasien, Dokter, Manager) dengan alokasi memori dinamis }

{ KONSTANTA }
const IDX_MIN : integer = 0      { Indeks minimum list }
const IDX_UNDEF : integer = -1  { Indeks tak terdefinisi }

{ TIPE BENTUKAN }
type GenericData : <
    data : ↑void,                { pointer ke data generik }
    type : DataType            { jenis data yang ditunjuk }
>
{ GenericData adalah tipe bentukan yang dapat memuat data dengan type
apapun }
{ (Patient, Doctor, etc) karena menggunakan sebuah pointer void }

```

```

    { DataType menunjukkan type yang ada dalam pointer data }
    { Contoh: }
    { variabel GD bertipe GenericData menampung data dari sebuah pasien
dengan type Patient, maka: }
    { GD.data adalah ↑Patient, GD.type bernilai DATA_TYPE_PATIENT }

    type_ElType : ↑GenericData      { type elemen yang digunakan dalam
list }
    type_IdxType : integer        { type index yang digunakan dalam
list }

    type_ListDin : <
        buffer : ↑array[0..capacity-1] of ElType, { memori tempat
penyimpan elemen }
        nEff : integer, { banyaknya elemen
efektif, ≥0 }
        capacity : integer { ukuran maksimum
elemen }
    >
    { Indeks yang digunakan [0..capacity-1] }

    { FUNGSI PRIMITIF }
    { ***** KONSTRUKTOR ***** }
    procedure createLD(input/output l : ↑ListDin, input_capacity :
integer)
    { I.S. l sembarang, capacity > 0 }
    { F.S. Terbentuk list dinamis l kosong dengan kapasitas capacity }

    function createGD(input data : ↑void, input_type : DataType) →
↑GenericData
    { Membuat GenericData untuk data dalam heap }
    { Jika gagal alokasi: output "GAGAL REALOKASI MEMORI" }

    { ***** DEALOKATOR ***** }
    procedure deallocateLD(input/output l : ↑ListDin)
    { I.S. l terdefinisi, mungkin kosong }
    { F.S. Semua elemen difree, CAPACITY(l)=0, NEFF(l)=0 }

    procedure deallocateGD(input/output gd : ↑GenericData)
    { I.S. gd terdefinisi }
    { F.S. Data dalam gd di-free, type=DATA_TYPE_UNKNOWN, data=↑NULL }

    { ***** SELEKTOR (TAMBAHAN) ***** }
    function lengthLD(input l : ↑ListDin) → integer
    { Mengembalikan banyaknya elemen efektif list }

    function getDataTypeGD(input gd : ↑GenericData) → DataType
    { Mengembalikan type dari gd }

    function getGDbyIdx(input l : ↑ListDin, input_idx : integer) →
↑GenericData
    { Mengembalikan address GenericData pada index idx }
    { Jika idx tidak valid: output "INVALID IDX", return ↑NULL }

    function getPatientInGD(input gd : ↑GenericData) → ↑Patient
    { Mengembalikan address Patient dalam gd }
    { Jika type bukan Patient: output "BUKAN PASIEN", return ↑NULL }

```

```

function getDoctorInGD(input gd : ↑GenericData) → ↑Doctor
{ Mengembalikan address Doctor dalam gd }
{ Jika type bukan Doctor: output "BUKAN DOKTER", return ↑NULL }

function getManagerInGD(input gd : ↑GenericData) → ↑Manager
{ Mengembalikan address Manager dalam gd }
{ Jika type bukan Manager: output "BUKAN MANAGER", return ↑NULL }

{ *** Selektor INDEKS *** }
function getLDFirstIdx(input l : ↑ListDin) → integer
{ Mengembalikan index pertama (0) }

function getLDLastIdx(input l : ↑ListDin) → integer
{ Mengembalikan index elemen efektif terakhir (NEFF(l)-1) }

{ ***** Test Indeks ***** }
function isLDIdxValid(input l : ↑ListDin, input i : integer) →
boolean
{ Mengembalikan true jika i valid untuk kapasitas list ( $0 \leq i <$ 
capacity) }

function isLDIdxEff(input l : ↑ListDin, input i : integer) → boolean
{ Mengembalikan true jika i terdefinisi untuk list ( $0 \leq i <$  NEFF(l))
}

{ ***** TEST KOSONG/PENUH ***** }
function isLDEmpty(input l : ↑ListDin) → boolean
{ Mengembalikan true jika list kosong (NEFF(l) = 0) }

function isLDFull(input l : ↑ListDin) → boolean
{ Mengembalikan true jika list penuh (NEFF(l) = capacity) }

{ ***** OPERASI TAMBAH/HAPUS ***** }
procedure insertLastLD(input/output l : ↑ListDin, input_val : ElType)
{ I.S. l terdefinisi, mungkin penuh }
{ F.S. Jika tidak penuh: val menjadi elemen terakhir }
{ Jika penuh: output "LIST PENUH" }

procedure deleteLastLD(input/output l : ↑ListDin)
{ I.S. l terdefinisi, mungkin kosong }
{ F.S. Jika tidak kosong: elemen terakhir dihapus }
{ Jika kosong: output "LIST KOSONG" }

{ ***** OPERASI UBAH UKURAN ***** }
procedure expandLD(input/output l : ↑ListDin, input_num : integer)
{ I.S. List terdefinisi }
{ F.S. Ukuran list bertambah sebanyak num }
{ Jika gagal: output "GAGAL REALOKASI MEMORI" }

procedure shrinkLD(input/output l : ↑ListDin, input_num : integer)
{ I.S. l terdefinisi }
{ F.S. Ukuran list berkurang sebanyak num, jika valid }
{ Jika invalid num: output "INVALID NUM" }
{ Jika gagal: output "GAGAL REALOKASI MEMORI" }

procedure compressLD(input/output l : ↑ListDin)
{ I.S. l terdefinisi, mungkin kosong }
{ F.S. Jika tidak kosong: nEff = capacity }

```

```
{ Jika kosong: output "LIST KOSONG" }
{ Jika gagal: output "GAGAL REALOKASI MEMORI" }
```

KAMUS ADT SET

```
{ Modul Set (Array Dinamis Unik Rata Kiri Eksplisit) }
{ Berisi definisi dan primitif untuk ADT Set dengan elemen unik (tidak duplikat) }
```

```
{ Digunakan untuk menyimpan username pengguna secara unik }
```

```
type Set : <
    buffer : ↑array[0..capacity-1] of string,
    nEff : integer,
    capacity : integer
>
{ Definisi:
    - Set kosong: nEff = 0
    - Elemen pertama: buffer[0]
    - Elemen terakhir: buffer[nEff-1]
    - Indeks valid: [0..capacity-1] }

{ ***** KONSTRUKTOR ***** }
procedure createSet(input/output s : ↑Set, input_capacity : integer)
{ I.S. s sembarang, capacity > 0 }
{ F.S. Set kosong s terbentuk dengan kapasitas = capacity }

{ ***** DEALOKATOR ***** }
procedure freeSet(input/output s : ↑Set)
{ I.S. s terdefinisi, mungkin kosong }
{ F.S. Semua elemen di-free, buffer=↑NULL, capacity=0, nEff=0 }

{ ***** FUNGSIONAL ***** }
procedure addToSet(input/output s : ↑Set, input_val : string)
{ I.S. s dan val terdefinisi, mungkin penuh }
{ F.S. Jika tidak penuh dan val unik: val ditambahkan, nEff++ }
{ Jika penuh: output "SET PENUH" }
{ Jika duplikat: output "VAL SUDAH ADA" }

procedure removeFromSet(input/output s : ↑Set, input_val : string)
{ I.S. s terdefinisi, mungkin kosong }
{ F.S. Jika val ada: val dihapus, elemen dirata kiri, nEff-- }

function idxIsValInSet(input s : ↑Set, input_val : string) → integer
{ Mengembalikan indeks val jika ada, -1 jika tidak }

{ ***** TEST KOSONG/PENUH ***** }
function isSetEmpty(input s : ↑Set) → boolean
{ Mengembalikan true jika nEff = 0 }

function isSetFull(input s : ↑Set) → boolean
{ Mengembalikan true jika nEff = capacity }

{ ***** OPERASI UKURAN ***** }
procedure expandSet(input/output s : ↑Set, input_num : integer)
{ I.S. s terdefinisi }
{ F.S. Kapasitas bertambah num }
{ Jika gagal: output "GAGAL REALOKASI MEMORI" }

procedure shrinkSet(input/output s : ↑Set, input_num : integer)
```

```

{ I.S. s terdefinisi }
{ F.S. Kapasitas berkurang num (jika valid) }
{ Jika invalid: output "INVALID NUM" }

```

```

procedure compressSet(input/output s : ↑Set)
{ I.S. s terdefinisi }
{ F.S. Kapasitas = nEff (jika tidak kosong) }
{ Jika kosong: output "SET KOSONG" }

```

KAMUS ADT LINKED LIST

```

{ Modul Linked List Dinamik dengan Data Generik }
{ Berisi definisi dan primitif untuk linked list dinamis dengan data generik }
{ Setiap elemen berupa node yang dialokasikan dinamis dan menyimpan pointer ke data generik }

```

```

type LinkedListNode : <
  id : integer,
  name : string,
  next : ↑LinkedListNode
>

```

```

{ Definisi:

```

```

- Node: Elemen penyimpan data (id, nama) dan pointer ke node berikutnya

```

```

- Head: Node pertama (l.head)
- Tail: Node terakhir (next = ↑NULL)
- Size: Banyak node (l.size) }

```

```

type LinkedList : <
  head : ↑LinkedListNode,
  size : integer
>

```

```

{ Definisi:

```

```

- List kosong: size = 0, head = ↑NULL }

```

```

{ ***** KONSTRUKTOR ***** }

```

```

procedure createLL(input/output l : ↑LinkedList)
{ I.S. l sembarang }
{ F.S. List l kosong (size=0, head=↑NULL) }

```

```

function createLLNode(input id : integer, input name : string) →
↑LinkedListNode

```

```

{ Mengembalikan address node baru dengan id dan name }

```

```

procedure putNodeInLastLL(input/output l : ↑LinkedList, input
NodeAddress : ↑LinkedListNode)
{ I.S. l terdefinisi, mungkin kosong }
{ F.S. NodeAddress menjadi tail baru }

```

```

procedure putNodeInIdxLL(input/output l : ↑LinkedList, input
NodeAddress : ↑LinkedListNode, input idx : integer)
{ I.S. l terdefinisi, size ≥ 1 }
{ F.S. NodeAddress disisipkan di posisi idx }
{ Jika idx invalid: output "POSISI TIDAK VALID" }

```

```

{ ***** SELEKTOR ***** }

```

```

function getLLNodeById(input l : ↑LinkedList, input id : integer) →
↑LinkedListNode

```

```

    { Mengembalikan address node dengan id tertentu }

    function getLLNodeByIdx(input l : ↑LinkedList, input_idx : integer)
→ ↑LinkedListNode
    { Mengembalikan address node pada posisi idx }

    function getLLNodeByName(input l : ↑LinkedList, input_name : string)
→ ↑LinkedListNode
    { Mengembalikan address node dengan nama tertentu }

    function getPatientFromNode(input node : ↑LinkedListNode) → ↑Patient
    { Mengembalikan address Patient dari globalUserDatabase berdasarkan
id di node }

    { ***** TEST KOSONG ***** }
    function isLLEmpty(input l : LinkedList) → boolean
    { Mengembalikan true jika l kosong (size=0) }

    { ***** DEALOKATOR ***** }
    procedure freeLL(input/output l : ↑LinkedList)
    { I.S. l terdefinisi }
    { F.S. Semua node di-free, head=↑NULL, size=0 }

    procedure freeNodeLL(input/output l : ↑LinkedList, input_node :
↑LinkedListNode)
    { I.S. l terdefinisi, size ≥ 1 }
    { F.S. Node dihapus dan di-free }

```

KAMUS ADT QUEUE

```

{ Modul ADT Queue (Linked List-based) }
{ Berisi definisi dan primitif untuk Queue berbasis linked list }
{ Mengimplementasikan konsep FIFO (First-In-First-Out) }

type Queue : <
    front : ↑LinkedListNode, { Node paling depan }
    rear : ↑LinkedListNode, { Node paling belakang (next=↑NULL) }
    size : integer, { Jumlah elemen saat ini }
    capacity : integer { Kapasitas maksimum }
>
{ Definisi:
- Queue kosong: front = rear = ↑NULL, size = 0
- Queue penuh: size = capacity
- Satu elemen: front = rear = address node
- Elemen pertama: front→data (jika tidak kosong)
- Elemen terakhir: rear→data (jika tidak kosong) }

    { ***** KONSTRUKTOR ***** }
    procedure createQueue(input/output q : ↑Queue, input_capacity :
integer)
    { I.S. q sembarang }
    { F.S. Queue q kosong dengan kapasitas tertentu }

    procedure enqueue(input/output q : ↑Queue, input_node :
↑LinkedListNode)
    { I.S. q terdefinisi, mungkin kosong }
    { F.S. Jika tidak penuh: node ditambahkan di rear, size++ }
    { Jika penuh: output "ANTRIAN PENUH" }

```



```

{ ***** DEALOKATOR ***** }
procedure deQueue(input/output q : ↑Queue)
{ I.S. q terdefinisi }
{ F.S. Jika tidak kosong: front dihapus, front = front→next }
{ Jika kosong: output "ANTRIAN KOSONG" }

procedure freeQueue(input/output q : ↑Queue)
{ I.S. q terdefinisi, mungkin berisi node }
{ F.S. Semua node di-free, front=rear=↑NULL, size=0 }

{ ***** TEST KOSONG/PENUH ***** }
function isQueueEmpty(input q : ↑Queue) → boolean
{ Mengembalikan true jika q kosong (size=0) }

function isQueueFull(input q : ↑Queue) → boolean
{ Mengembalikan true jika q penuh (size=capacity) }

```

KAMUS ADT STACK

```

{ Modul ADT Stack (Array-based) }
{ Berisi definisi dan primitif untuk Stack berbasis array statik }
{ Mengimplementasikan konsep LIFO (Last-In-First-Out) }

const OBAT_MAX_SIZE : integer = 100 { Kapasitas maksimum stack }
const IDX_UNDEF : integer = -1 { Indeks tak terdefinisi }

type Stack : <
    obat : array[0..OBAT_MAX_SIZE-1] of Obat,
    top : integer
>
{ Definisi:
    - Stack kosong: top = IDX_UNDEF
    - Stack penuh: top = OBAT_MAX_SIZE - 1
    - Satu elemen: top = 0
    - Elemen teratas: obat[top] }

{ ***** SELEKTOR ***** }
{ TOPIDX(s) → s.top }
{ TOPELMT(s) → s.obat[s.top] }

{ ***** KONSTRUKTOR ***** }
procedure createStack(input/output s : ↑Stack)
{ I.S. s sembarang }
{ F.S. Stack s kosong (top = IDX_UNDEF) }

procedure pushStack(input/output s : ↑Stack, input o : Obat)
{ I.S. s terdefinisi, mungkin kosong }
{ F.S. Jika tidak penuh: o dimasukkan ke top, top++ }
{ Jika penuh: output "STACK PENUH" }

{ ***** DEALOKATOR/SELEKTOR ***** }
function popStack(input/output s : ↑Stack) → Obat
{ I.S. s terdefinisi }
{ F.S. Jika tidak kosong: mengembalikan obat di top, top-- }
{ Jika kosong: output "STACK KOSONG", return obat sentinel }

procedure deleteStack(input/output s : ↑Stack)
{ I.S. s terdefinisi }
{ F.S. s kosong (top = IDX_UNDEF) }

```

```

{ ***** SELEKTOR TAMBAHAN ***** }
function getTopIdx(input s : ↑Stack) → integer
{ Mengembalikan indeks top }

function getTopElmt(input s : ↑Stack) → Obat
{ Mengembalikan elemen top }
{ Jika kosong: output "STACK KOSONG", return obat sentinel }

function getElmtByIdxStack(input s : ↑Stack, input_idx : integer) →
Obat
{ Mengembalikan elemen pada indeks idx }
{ Jika kosong: output "STACK KOSONG" }
{ Jika idx invalid: output "IDX DI LUAR RANGE", return obat sentinel
}

function stackSize(input s : ↑Stack) → integer
{ Mengembalikan ukuran stack (top + 1) }

{ ***** TEST KOSONG/PENUH ***** }
function isStackEmpty(input s : ↑Stack) → boolean
{ Mengembalikan true jika kosong (top = IDX_UNDEF) }

function isStackFull(input s : ↑Stack) → boolean
{ Mengembalikan true jika penuh (top = OBAT_MAX_SIZE - 1) }

```

2. F01

```

KAMUS GLOBAL
globalUserDatabase: ListDin
globalCurrentUserGD: GenericData*

procedure login()
KAMUS LOKAL:
username, password : string
found : boolean
i : integer
Gd : GenericData*

```

3. F02

```

KAMUS GLOBAL
globalUserDatabase: ListDin
globalUsernames: Set

procedure register()
KAMUS LOKAL:
username, password : string
usernameLower : string
maxId, newId : integer
i : integer
newPatient: Patient*
newGD: GenericData*
emptyInventory : array[0..INVENTORY_SIZE-1] of integer

```

```
emptyKondisiTubuh : array[0..KONDISI_TUBUH_SIZE-1] of float
```

4. F03

```
KAMUS GLOBAL  
globalCurrentUserGD: GenericData*  
globalNotLogin: GenericData  
  
procedure logout()  
KAMUS LOKAL:  
-
```

5. F04

```
KAMUS GLOBAL  
globalUserDatabase: ListDin  
  
procedure RunLengthEncoding()  
KAMUS LOKAL:  
i: integer  
count: integer  
buffer: string  
  
procedure lupaPassword()  
KAMUS LOKAL:  
usernameInput: string  
kodeUnik: string  
encodedStored: string  
found: boolean  
correct: boolean  
i: integer  
user: GenericData*  
userPatient: Patient*  
userDoctor: Doctor*  
userManager: Manager*
```

6. F05

```
KAMUS GLOBAL  
globalCurrentUser: GenericData  
globalCurrentPatient : Patient  
globalCurrentDoctor : Doctor  
globalCurrentManager : Manager  
  
procedure mainMenu ()  
KAMUS LOKAL  
-  
  
procedure menuPasien ()  
KAMUS LOKAL  
input: string
```

```

procedure menuDokter ()
KAMUS LOKAL
input: string

procedure menuManager ()
KAMUS LOKAL
input: string

procedure menuBelumLogin ()
KAMUS LOKAL
input: string

procedure helpPasien ()
KAMUS LOKAL
-

procedure helpDokter ()
KAMUS LOKAL
-

procedure helpManager ()
KAMUS LOKAL
-

procedure helpBelumLogin ()
KAMUS LOKAL
-

```

7. F06

```

KAMUSGLOBAL
globalDenahRumahSakit: DataTypeDenah
globalUserDataBase: ListDin

procedure lihatDenah ()
KAMUSLOKAL
row: integer
column : integer

procedure lihatRuangan (input row, column: integer)
KAMUSLOKAL
nomorRuangan: string
rowRuangan: integer
columnRuangan: integer
idDokter: integer
jumlahPasien: integer

```

8. F07

```

KAMUS GLOBAL :
globalUserDatabase: ListDin

procedure lihatUser(input globalUserDatabase: Listdin)
KAMUS LOKAL :

```

```

urt, srt, tabelEff, idxPrinted, i, j, count: integer
save: GenericData*
p: Patient*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]
urutan: array of char[0..12]
sortin: array of char[0..12]

procedure lihatDokter(input globalUserDatabase: Listdin)
KAMUS LOKAL:
urt, srt, tabelEff, idxPrinted, i, j, count: integer
save: GenericData*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]
urutan: array of char[0..12]
sortin: array of char[0..12]

procedure lihatPasien(input globalUserDatabase: Listdin)
KAMUS LOKAL:
urt, srt, tabelEff, idxPrinted, i, j, count: integer
save: GenericData*
p: Patient*
printed: array of tabel[0..tabelEff - 1]
urutan: array of char[0..12]
sortin: array of char[0..12]

```

9. F08

```

KAMUS GLOBAL:
globalUserDatabase: ListDin

procedure cariUser(input globalUserDatabase: Listdin)
KAMUS LOKAL:
urt, src, eff, tabelEff, idxPrinted, i, j: integer
idx, mid, low, high, cari, count: integer
search: array of char[0..50]
save: GenericData*
p: Patient*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]

procedure cariDokter(input globalUserDatabase: Listdin)
KAMUS LOKAL:
urt, src, eff, tabelEff, idxPrinted, i, j: integer
idx, mid, low, high, cari, count: integer
search: array of char[0..50]
save: GenericData*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]

procedure cariPasien(input globalUserDatabase: Listdin)
KAMUS LOKAL:
eff, src, tabelEff, idxPrinted, count, i, j: integer
idx, mid, low, high, cari, urt, srt: integer
save: GenericData*
p: Patient*

```

```
printed: array of tabel[0..tabelEff - 1]
arr: array of tabel[0..count - 1]
search1: array of char[0..50]
search2: array of char[0..50]
urutan: array of char[0..12]
sortin: array of char[0..12]
```

10. F09

11. F10

```
KAMUS GLOBAL:
globalUserDatabase: ListDin
globalUsernames: Set
globalDenahRumahSakit: Matrix
STR_MAX_SIZE: integer constant

procedure tambahDokter()
KAMUS LOKAL:
usernameNew, passwordNew, usernameLower : string
maxId: integer
newId: integer
newDoctor: Doctor*
newGD: GenericData*
gd: GenericData*
d: Doctor*
p: Patient*
m: Manager*

procedure assignDokter()
KAMUS LOKAL:
username: string
ruangan: string
idDokter: integer
dokterPtr: Doctor*
row, col: integer
dokterSudahDiassign: boolean
rowAssigned, colAssigned: integer
idDokterDiTarget: integer
namaDokterDiRuangan: string
gd: GenericData*
d: Doctor*
```

12. F11

```
procedure diagnosa(input antrianPasien: Queue)
KAMUSLOKAL
pasienNode: LinkedListNode
pasien: Patient
penyakit: Penyakit
cocok: boolean
nilai: float
batasMin: float
```

```
batasMax: float  
i, j: integer
```

13. F12

```
procedure ngobatin(input antrianPasien: Queue)  
KAMUSLOKAL  
pasienNode: LinkedListNode  
pasien: Patient  
penyakit: string  
i: integer  
obat: string
```

14. F13

```
procedure pulangDok(input currentUser: User)  
KAMUSLOKAL  
pasien: Patient  
i: integer  
obatBenar: boolean
```

15. F14

```
procedure daftarCheckup(input globalCurrentUserGD: GenericData)  
KAMUS LOKAL:  
lokasiRuangan: Point  
globalUserDatabase: ListDin  
globalDenahRumahSakit: DataTypeDenah  
count, col, row, idx, idxtemp, idokter, i, j: integer  
rowtemp: array of integer[0..100]  
coltemp: array of integer[0..100]  
pasien: LinkedListNode*  
baris: char  
name: array of array of char[0..100][0..100]  
spes: array of array of char[0..100][0..100]
```

16. F15

```
KAMUS GLOBAL  
globalDenahRumahSakit: DataTypeDenah  
globalCurrentUserGD: Generic Data  
DATA_TYPE_DOCTOR: DataType  
  
function posisiRuanganAntrianPasien(userId: integer) -> Point  
{menghasilkan lokasi ruangan pasien yang bertipe Point}  
KAMUS LOKAL
```

```

lokasiRuangan: Point
{variabel yang akan dihasilkan fungsi}
transversal: LinkedListNode
{variabel untuk membantu mencari lokasi lokasi ruangan saat loop}

procedure antrianSaya()
{I.S.: lokasiRuangan sudah terdefinisi pada fungsi di atas}
{F.S.: Menampilkan posisi antrian pasien}
KAMUS LOKAL:
lokasiRuangan: Point
{variabel yang menunjukkan lokasi ruangan dan posisi antrian}
procedure getAccountName(input id: integer, dataType: DataType)
{untuk menampilkan nama dokter}
function posisiRuanganAntrianPasien(input userId: integer) -> Point
{untuk mencari posisi ruangan dan antrian pasien dan disimpan dalam
variabel lokasiRuangan}

```

17. F16

```

KAMUS GLOBAL
const INVENTORY_SIZE : integer = 10
const UNDEF_INT_DATA : integer = -99
globalCurrentPatient: Patient
globalObatDatabase: ObatDataBase

procedure goToLeft()
{I.S.: array inventory sudah terdefinisi bisa jadi belum rata kiri}
{F.S.: array inventory sudah rata kiri}
KAMUS LOKAL
Idx, i: integer
{idx untuk menandai index temp, i untuk membantu dalam looping}
temp: array[0..INVENTORY_SIZE] of integer
{array sementara yang sudah rata kiri yang isinya akan dipindahkan ke
array inventory}

procedure minumObat()
{I.S.: sudahDiDiagnosis, sudahDiObatin, array inventory, stack perut
sudah terdefinisi}
{F.S.: jika sesuai syarat, obat yang dipilih akan dipindahkan ke perut
dan list obat yang baru sudah rata kiri}
KAMUS LOKAL:
jlhObat, pilihan, i, j: integer
{jlhObat digunakan untuk validasi isi inventory dan obat yang akan
dipilih, pilihan digunakan untuk menyimpan nomor urut obat yang dipilih
dari daftar obat, i dan j digunakan sebagai variabel pembantu dalam loop}
o, iniObat: Obat
{o digunakan untuk mencetak daftar obat, iniObat digunakan untuk push
obat ke perut}
procedure pushStack(input s: Address, o: Obat)
{prosedur untuk memindahkan obat ke perut}
procedure goToLeft()
{prosedur untuk merata-kirikan array inventory setelah minum obat}

```

18. F17

KAMUS GLOBAL

```
const INVENTORY_SIZE : integer = 10  
const UNDEF_INT_DATA : integer = -99  
globalCurrentPatient: Patient
```

```
procedure minumPenawar()
```

{I.S.: stack perut sudah terdefinisi, mungkin kosong, array inventory sudah terdefinisi, mungkin kosong}

KAMUS LOKAL

```
i: integer
```

{i adalah variabel yang digunakan untuk membantu dalam loop}

```
backToInventory: Obat
```

{backToInventory adalah variabel yang digunakan untuk menandai obat yang akan dikembalikan ke perut setelah minum penawar}

```
function popStack(input s: Address) -> Obat
```

{fungsi untuk menghapus obat dari suatu stack dan mengembalikan obat yang dikembalikan tersebut}

```
function isStackEmpty(input s: Address) -> boolean
```

{fungsi untuk mengecek apakah suatu stack kosong(true jika kosong, false jika tidak)}

19. F18**KAMUS LOKAL**

```
input : character
```

20. B02

```
procedure ubahDenah()
```

KAMUS GLOBAL

```
globalDenahRumahSakit: DataTypeDenah
```

KAMUS LOKAL

```
newRow, newColumn: integer
```

```
tempRow: integer
```

```
row, column: integer
```

```
bisaUbahDenah: Point <row:integer, column:integer, antrian: integer>
```

```
tempPoint: Point
```

```
procedure pindahDokter()
```

KAMUS LOKAL

```
ruanganLama, ruanganBaru: string
```

```
rowLama, columnLama, rowBaru, columnBaru: integer
```

```
tempRuanganData: DataTypeRuangan <idDokter: integer, kapasitasRuangan: integer,  
kapasitasAntrian: integer, idAntrian: Queue>
```

21. B06

```
procedure skipAntrian()
```

KAMUS LOKAL

```
lokasiRuangan: Point { Menyimpan informasi baris, kolom, dan status/posisi  
antrian pasien.
```

```
.antrian = -1 jika tidak di antrian, -2 jika di dalam  
ruangan,
```

```
angka positif jika di antrian tunggu (posisi ke-). }
```

```
sizeQueue: integer
```

```
pasienYangSkip, pasienSebelumYangSkip, pasienTerakhirDalamRuangan, tempNode:  
Address { Address ke LinkedListNode }
```

```

i: integer { Variabel iterasi }
currentPatientId: integer { ID pasien saat ini }
roomRow, roomCol: integer { Untuk menyimpan baris dan kolom ruangan dari
lokasiRuangan }
targetQueue: QueueType { Asumsi QueueType adalah tipe dari idAntrian }

procedure cancelAntrian()

KAMUS LOKAL
    lokasiRuangan: Point
    sizeQueue: integer
    pasienYangCancel, pasienSebelumYangCancel: Address { Address ke LinkedListNode
}
    i: integer
    currentPatientId: integer
    roomRow, roomCol: integer
    targetQueue: QueueType

```

DESAIN DEKOMPOSISI ALGORITMIK DAN FUNGSIONAL PROGRAM

// FLOWCHART MAIN

NOTASI ALGORITMIK SETIAP FUNGSI DAN PROSEDUR UTAMA

1. F01

```

procedure login(input globalUserDatabase: ListDin, input/output loggedInUserSet:
Set)
{I.S. globalUserDatabase terdefinisi dan tidak kosong, loggedInUserSet kosong
(belum ada user login)}
{F.S. jika username dan password sesuai maka user ditambahkan ke loggedInUserSet
dan pesan login ditampilkan}

KAMUS_LOKAL
    username, password: array of char[0..50]
    found: boolean
    save: GenericData*
    i: integer
    p: Patient*
    d: Doctor*
    m: Manager*

ALGORITMA
    output("Masukkan username: ")
    input(uname)
    output("Masukkan password: ")
    input(pass)

    found <- false
    i traversal[0..globalUserDatabase.nEff - 1]
        save <- globalUserDatabase.buffer[i]
        if(save.type = DATA_TYPE_PATIENT) then
            p <- (Patient*)(save.data)
            if(stringCompare(p.username, uname) = 0 AND stringCompare(p.password,
pass) = 0) then
                insertSet(&loggedInUserSet, save)
                output("Selamat datang, ", p.nama, ".")
                output("Anda login sebagai Pasien.")
                found <- true
                break
            else if(save.type = DATA_TYPE_DOCTOR) then
                d <- (Doctor*)(save.data)
                if(stringCompare(d.username, uname) = 0 AND stringCompare(d.password,
pass) = 0) then
                    insertSet(&loggedInUserSet, save)

```

```

        output("Selamat datang, ", d.nama, ".")
        output("Anda login sebagai Dokter.")
        found <- true
        break
    else if(save.type = DATA_TYPE_MANAGER) then
        m <- (Manager*)(save.data)
        if(stringCompare(m.username, uname) = 0 AND stringCompare(m.password,
pass) = 0) then
            insertSet(&loggedInUserSet, save)
            output("Selamat datang, ", m.nama, ".")
            output("Anda login sebagai Manajer.")
            found <- true
            break
        if(NOT found) then
            output("Username atau password salah. Silakan coba lagi.")

```

2. F02

```

procedure registerPasien(input/output globalUserDatabase: ListDin, input/output
usernameSet: Set)
{I.S. globalUserDatabase dan usernameSet terdefinisi, usernameSet berisi username
unik dalam lowercase}
{F.S. jika username belum digunakan maka data pasien ditambahkan ke
globalUserDatabase dan usernameSet}

```

KAMUS_LOKAL

```

    newUsername, newPassword, newName: array of char[0..50]
    idBaru: integer
    userCheck: array of char[0..50]
    newPasien: Patient*
    newData: GenericData

```

ALGORITMA

```

    output("Masukkan username: ")
    input(newUsername)
    toLowerString(userCheck, newUsername)

    if(isInSet(usernameSet, userCheck)) then
        output("Username sudah terdaftar. Silakan coba lagi.")
        return

    output("Masukkan password: ")
    input(newPassword)
    output("Masukkan nama: ")
    input(newName)

    idBaru <- globalUserDatabase.nEff + 1
    alokasiPasien(&newPasien, idBaru, newUsername, newPassword, newName)

    newData.type <- DATA_TYPE_PATIENT
    newData.data <- newPasien

    insertLastListDin(&globalUserDatabase, newData)
    insertSet(&usernameSet, userCheck)

    output("Registrasi akun pasien berhasil!")
    output("Silakan login untuk melanjutkan.")

```

3. F03

```

procedure logout(input/output loggedInUserSet: Set)
{I.S. loggedInUserSet berisi satu user yang sedang login}
{F.S. loggedInUserSet kosong (user berhasil logout)}

```

KAMUS_LOKAL

```

    userKeluar: GenericData
    p: Patient*

```

```

d: Doctor*
m: Manager*
namaUser: array of char[0..50]

```

ALGORITMA

```

userKeluar <- getOneFromSet(loggedInUserSet)

if(userKeluar.type = DATA_TYPE_PATIENT) then
  p <- (Patient*)(userKeluar.data)
  namaUser <- p.nama
else if(userKeluar.type = DATA_TYPE_DOCTOR) then
  d <- (Doctor*)(userKeluar.data)
  namaUser <- d.nama
else if(userKeluar.type = DATA_TYPE_MANAGER) then
  m <- (Manager*)(userKeluar.data)
  namaUser <- m.nama

deleteSet(&loggedInUserSet, userKeluar)

output("Sampai jumpa kembali, ", namaUser, ".")
output("Anda berhasil logout dari akun.")

```

4. F04

```

function runLengthEncoding(inputStr: array of char[0..N]) → encodedStr: array of
char[0..M]
{I.S. inputStr terdefinisi, berupa string yang mungkin mengandung karakter
berulang}
{F.S. encodedStr berisi hasil kompresi RLE dalam format karakter + jumlah
kemunculan berturut-turut}

```

KAMUS_LOKAL

```

i, j, count, len: integer
encodedStr: array of char[0..M]
tempCountStr: array of char[0..10]

```

ALGORITMA

```

i <- 0
j <- 0
len <- length(inputStr)

while(i < len) do
  count <- 1

  while(i + 1 < len and inputStr[i] = inputStr[i + 1]) do
    count <- count + 1
    i <- i + 1

  encodedStr[j] <- inputStr[i]
  j <- j + 1

  tempCountStr <- integerToString(count)
  k traversal [0..length(tempCountStr) - 1]
  encodedStr[j] <- tempCountStr[k]
  j <- j + 1

  i <- i + 1

encodedStr[j] <- '\0' // null-terminator jika diperlukan

-> encodedStr

```

```

procedure lupaPassword(input/output globalUserDatabase: ListDin)

```

```

{I.S. globalUserDatabase terdefinisi dan tidak kosong}
{F.S. Jika ditemukan, password ditampilkan berdasarkan username yang cocok dengan
hasil RLE}

```

KAMUS_LOKAL

```

inputUsername, encodedInput, encodedStored: array of char[0..50]
found: boolean

```

```

i: integer
save: GenericData*
p: Patient*
rle: array of char[0..50]

```

ALGORITMA

```

output("Masukkan username: ")
(inputUsername)

encodedInput <- runLengthEncoding(inputUsername)

found <- false

i traversal [0..globalUserDatabase.nEff - 1]
  save <- globalUserDatabase.buffer[i]

  if(save.type = DATA_TYPE_PATIENT) then
    p <- (Patient*)(save.data)
    encodedStored <- runLengthEncoding(p.username)

    if(encodedInput = encodedStored) then
      output("Username ditemukan.")
      output("Password Anda adalah: ", p.password)
      found <- true
      break

if(found = false) then
  output("Username tidak ditemukan atau tidak cocok.")

```

5. F05

```

procedure mainMenu ()
{I.S. Role User (Melalui Global Variable)}
{F.S. Menuju prosedur menu sesuai role user}

```

KAMUS LOKAL

-

ALGORITMA

```

repeat
  depend on (globalCurrentUserGD.type)
    globalCurrentUserGD.type = DATA_TYPE_PATIENT : menuPasien()
    globalCurrentUserGD.type = DATA_TYPE_DOCTOR : menuDokter()
    globalCurrentUserGD.type = DATA_TYPE_MANAGER : menuManager()
    globalCurrentUserGD.type = DATA_TYPE_UNKNOWN : menuBelumLogin()
  until (false)

```

```

procedure menuPasien()
{I.S. Command dari terminal (seperti HELP atau LIHAT DENAH)}
{F.S. Menuju prosedur sesuai command yang diinput atau output "Input tidak valid. Masukkan kembali input yang valid." jika memasukkan input yang invalid}

```

KAMUS LOKAL

```

inputUser: string { Menyimpan input dari pengguna }
keluarMenu: boolean { Flag untuk mengontrol keluarnya dari loop menu }
perintahValidDijalankan: boolean { Flag untuk menandai jika ada perintah valid yang dijalankan }

```

ALGORITMA

```

printMenuPasien()
globalCurrentPatient <- globalCurrentUserGD.data
output("Halo ", globalCurrentUserGD.data.username)
output("Silahkan masukan fungsi yang anda ingin jalankan")
output("Masukan HELP untuk memunculkan list fungsi-fungsi yang valid")
output("")

keluarMenu <- false
repeat
  output(">>> ")
  input(inputUser)
  perintahValidDijalankan <- false

  depend on (inputUser)
    inputUser = "HELP" : helpPasien(); perintahValidDijalankan <-

```

```

true
inputUser = "LIHAT_DENAH"      : lihatDenah(); perintahValidDijalankan <-
true
inputUser = "LIHAT_RUANGAN"    : lihatRuangan(-1, -1);
perintahValidDijalankan
                                <- true
                                :
inputUser = "PULANGDOK"        : pulangDok(getQueueFromPatientId(globalCurrentPatient.id)); perintahValidDijalankan <-
                                true
                                :
inputUser = "ANTRIAN"          : antrianSaya(); perintahValidDijalankan
<-true
inputUser = "DAFTAR_CHECKUP"    : daftarCheckup(globalCurrentUserGD);
                                perintahValidDijalankan <- true
inputUser = "MINUM_OBAT"       : minumObat(); perintahValidDijalankan <-
true
inputUser = "PENAWAR"          : minumPenawar(); perintahValidDijalankan <-
                                true
                                :
inputUser = "LOGIN"            : login(); perintahValidDijalankan <- true
inputUser = "LOGOUT"           : logout()
                                if (globalCurrentUserGD.type !=
                                DATA_TYPE_PATIENT) then
                                keluarMenu <- true
                                perintahValidDijalankan <- true
inputUser = "EXIT"             : exitFromHospital()
                                keluarMenu <- true
                                perintahValidDijalankan <- true

if (not perintahValidDijalankan) then
output("")
output("Input tidak valid. Masukan kembali input yang valid.✗")
output("")
until (keluarMenu)

```

procedure menuDokter()
 {I.S. Command dari terminal (seperti HELP atau LIHAT_DENAH)}
 {F.S. Menuju prosedur sesuai command yang diinput atau output "Input tidak valid. Masukan kembali input yang valid." jika memasukan input yang invalid}

KAMUS LOKAL

```

inputUser: string { Menyimpan input dari pengguna }
keluarMenu: boolean { Flag untuk mengontrol keluarnya dari loop menu }
perintahValidDijalankan: boolean { Flag untuk menandai jika ada perintah valid yang dijalankan }

```

ALGORITMA

```

printMenuDokter()
globalCurrentDoctor <- globalCurrentUserGD.data
output("Halo Dokter ", globalCurrentUserGD.data.username, ". Silahkan masukan fungsi yang anda ingin jalankan.")
output("Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.")
output("")

keluarMenu <- false
repeat
output(">>> ")
input(inputUser)
perintahValidDijalankan <- false

depend_on (inputUser)
inputUser = "HELP"              : helpDokter(); perintahValidDijalankan <-
true
inputUser = "LIHAT_DENAH"      : lihatDenah(); perintahValidDijalankan <-
true
inputUser = "LIHAT_RUANGAN"    : lihatRuangan(-1, -1);
perintahValidDijalankan
                                <- true
                                :
inputUser = "DIAGNOSIS"        : diagnosis(getQueueFromDoctorId(globalCurrentDoctor.id)); perintahValidDijalankan <- true
inputUser = "NGOBATIN"         : ngobatin(getQueueFromDoctorId(globalCurrentDoctor.id)); perintahValidDijalankan <- true
inputUser = "LOGIN"            : login(); perintahValidDijalankan <- true

```

```

inputUser = "LOGOUT"           : logout()
                                if (globalCurrentUserGD.type !=
                                DATA_TYPE_DOCTOR) then
                                    keluarMenu <- true
inputUser = "EXIT"             : exitFromHospital()
                                keluarMenu <- true
                                perintahValidDijalankan <- true

if (not perintahValidDijalankan) then
    output("")
    output("Input tidak valid. Masukan kembali input yang valid.✗")
    output("")
until (keluarMenu)

```

procedure menuManager()
 {I.S. Command dari terminal (seperti HELP atau LIHAT_DENAH)}
 {F.S. Menuju prosedur sesuai command yang diinput atau output "Input tidak valid. Masukan kembali input yang valid." jika memasukan input yang invalid}

KAMUS LOKAL

```

inputUser: string { Menyimpan input dari pengguna }
keluarMenu: boolean { Flag untuk mengontrol keluarnya dari loop menu }
perintahValidDijalankan: boolean { Flag untuk menandai jika ada perintah valid yang dijalankan }

```

ALGORITMA

```

printMenuManager()
globalCurrentManager <- globalCurrentUserGD.data
output("Halo Manager ", globalCurrentUserGD.data.username, ". Silahkan masukan fungsi yang anda ingin jalankan.")
output("Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.")
output("")

keluarMenu <- false
repeat
    output(">>> ")
    input(inputUser)
    perintahValidDijalankan <- false

    depend on (inputUser)
        inputUser = "HELP"           : helpManager();
        perintahValidDijalankan      <- true

        inputUser = "LIHAT_DENAH"    : lihatDenah();
        perintahValidDijalankan      <- true

        inputUser = "LIHAT_USER"     : lihatUser(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "LIHAT_DOKTER"   : lihatDokter(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "LIHAT_PASIEN"   : lihatPasien(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "LIHAT_RUANGAN"  : lihatRuangan(-1, -1);
        perintahValidDijalankan <- true
        inputUser = "CARI_USER"      : cariUser(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "CARI_PASIEN"    : cariPasien(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "CARI_DOKTER"    : cariDokter(globalUserDatabase);
        perintahValidDijalankan <- true
        inputUser = "LIHAT_SEMUA_ANTRIAN" : lihatSemuaAntrian();
        perintahValidDijalankan <- true

        inputUser = "TAMBAH_DOKTER"  : tambahDokter();
        perintahValidDijalankan      <- true

        inputUser = "ASSIGN_DOKTER"  : assignDokter();
        perintahValidDijalankan      <- true

        inputUser = "PINDAH_DOKTER"  : pindahDokter();
        perintahValidDijalankan      <- true

        inputUser = "UBAH_DENAH"    : ubahDenah(); perintahValidDijalankan
        <- true

```

```

inputUser = "LOGIN" : login(); perintahValidDijalankan <- true
inputUser = "LOGOUT" : logout()
                     : if (globalCurrentUserGD.type !=
DATA_TYPE_MANAGER) then
                     keluarMenu <- true
                     perintahValidDijalankan <- true
inputUser = "EXIT" : exitFromHospital()
                  : keluarMenu <- true
                  : perintahValidDijalankan <- true

if (not perintahValidDijalankan) then
  output("")
  output("Input tidak valid. Masukan kembali input yang valid.✗")
  output("")
until (keluarMenu)

```

procedure menuBelumLogin()
 {I.S. Command dari terminal (seperti HELP atau LIHAT_DENAH)}
 {F.S. Menuju prosedur sesuai command yang diinput atau output "Input tidak valid. Masukan kembali input yang valid." jika memasukan input yang invalid}

KAMUS LOKAL

inputUser: string { Menyimpan input dari pengguna }
 keluarMenu: boolean { Flag untuk mengontrol keluarnya dari loop menu }
 perintahValidDijalankan: boolean { Flag untuk menandai jika ada perintah valid yang dijalankan }

ALGORITMA

```

output("Anda belum masuk ke suatu akun.")
output("Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.")
output("")

keluarMenu <- false
repeat
  output(">>> ")
  input(inputUser)
  perintahValidDijalankan <- false

  depend on (inputUser)
  <- inputUser = "HELP" : helpBelumLogin(); perintahValidDijalankan
  <- true
  inputUser = "LUPA_PASSWORD" : lupaPassword(); perintahValidDijalankan <-
  <- true
  inputUser = "LOGIN" : login()
  <- if (globalCurrentUserGD.type !=
DATA_TYPE_UNKNOWN) then
  <- keluarMenu <- true
  <- perintahValidDijalankan <- true
  inputUser = "LOGOUT" : logout(); perintahValidDijalankan <- true
  inputUser = "REGISTER" : registerPasien(); perintahValidDijalankan
  <- true
  inputUser = "EXIT" : exitFromHospital()
  <- keluarMenu <- true
  <- perintahValidDijalankan <- true

  if (not perintahValidDijalankan) then
    output("")
    output("Input tidak valid. Masukan kembali input yang valid.✗")
    output("")
  until (keluarMenu)

```

procedure helpPasien()
 {F.S. Output list prosedur yang dapat digunakan oleh pasien}

KAMUS LOKAL

-

ALGORITMA

```

output("Terimakasih telah memanggil fungsi Help")
output("Berikut merupakan fungsi-fungsi yang dapat anda gunakan")
output("1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta

```



```

    penjelasannya")
    output("2) LIHAT_DENAH : Memunculkan denah rumah sakit")
    output("3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode
ruangan)")
    output("4) PULANGDOK : Bertanya ke dokter apakah kamu sudah boleh pulang")
    output("5) DAFTAR_CHECKUP : Mendaftarkan check-up dengan dokter")
    output("6) ANTRIAN : Menunjukkan status antrian pasien")
    output("7) MINUM_OBAT : Meminum obat yang berada di inventory")
    output("8) PENAWAR : Meminum penawar untuk memuntahkan obat yang berada di
perut")
    output("9) LOGOUT : Keluar dari akun yang sedang digunakan")
    output("10) EXIT : Keluar dari program")
    output("")
    output("Footnote:")
    output("1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang
terdaftar")
    output("2) Jangan lupa untuk memasukkan input yang valid")

```

```

procedure helpDokter()
{F.S. Output list prosedur yang dapat digunakan oleh dokter}

```

KAMUS LOKAL

-

ALGORITMA

```

    output("Terimakasih telah memanggil fungsi Help")
    output("Berikut merupakan fungsi-fungsi yang dapat anda gunakan")
    output("1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta
penjelasannya")
    output("2) LIHAT_DENAH : Memunculkan denah rumah sakit")
    output("3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode
ruangan)")
    output("4) DIAGNOSIS : Mendiagnosis pasien yang berada di depan antrian")
    output("5) NGOBATIN : Mengobati pasien yang berada di depan antrian")
    output("6) LOGOUT : Keluar dari akun yang sedang digunakan")
    output("7) EXIT : Keluar dari program")
    output("")
    output("Footnote:")
    output("1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang
terdaftar")
    output("2) Jangan lupa untuk memasukkan input yang valid")

```

```

procedure helpManager()
{F.S. Output list prosedur yang dapat digunakan oleh manager}

```

KAMUS LOKAL

-

ALGORITMA

```

    output("Terimakasih telah memanggil fungsi Help")
    output("Berikut merupakan fungsi-fungsi yang dapat anda gunakan")
    output("1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta
penjelasannya")
    output("2) LIHAT_DENAH : Memunculkan denah rumah sakit")
    output("3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode
ruangan)")
    output("4) LIHAT_USER : Melihat data seluruh pengguna")
    output("5) LIHAT_PASIEN : Melihat data seluruh pasien")
    output("6) LIHAT_DOKTER : Melihat data seluruh dokter")
    output("7) CARI_USER : Mencari data pengguna secara spesifik berdasarkan ID
atau
Nama")
    output("8) CARI_PASIEN : Mencari data pengguna secara spesifik berdasarkan ID,
Nama, atau Penyakit")
    output("9) CARI_DOKTER : Mencari data pengguna secara spesifik berdasarkan ID,
atau Nama")
    output("10) LIHAT_SEMUA_ANTRIAN : Melihat rincian di seluruh ruangan saat
ini")
    output("11) TAMBAH_DOKTER : Menambahkan dokter baru")
    output("12) ASSIGN_DOKTER : Melakukan assign ruangan ke dokter tertentu yang
belum memiliki ruangan")
    output("13) PINDAH_DOKTER XX YY: Melakukan pemindahan dokter dari ruangan XX
ke
YY (XX, YY: kode ruangan)")
    output("14) UBAH_DENAH X Y : Mengubah ukuran denah menjadi Y X (Y: Jumlah

```

```

barisan, X : Jumlah kolom)")
output("15) LOGOUT : Keluar dari akun yang sedang digunakan")
output("16) EXIT : Keluar dari program")
output("")
output("Footnote:")
output("1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang
terdaftar")
output("2) Jangan lupa untuk memasukkan input yang valid")

```

```

procedure helpBelumLogin()
{F.S. Output list prosedur yang dapat digunakan oleh user yang belum login}

```

KAMUS LOKAL

-

ALGORITMA

```

output("Terimakasih telah memanggil fungsi Help")
output("Berikut merupakan fungsi-fungsi yang dapat anda gunakan")
output("1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta
penjelasannya")
output("2) LOGIN : Masuki suatu akun")
output("3) LUPA_PASSWORD : Mengganti atau memperbarui password akun")
output("4) REGISTER : Membuat akun baru")
output("5) EXIT : Keluar dari program")
output("")
output("Footnote:")
output("1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang
terdaftar")
output("2) Jangan lupa untuk memasukkan input yang valid")

```

6. F06

```

procedure kodeRuanganKonverter(input kodeRuangan: string, output rowRuangan:
integer, output columnRuangan: integer)
{I.S. String yang merupakan kode ruangan seperti A1, B6, atau G10}
{F.S. Koordinat lokasi ruangan di sebuah matriks 2 dimensi yang berupa integer
rowRuangan dan columnRuangan}

```

KAMUS LOKAL

-

ALGORITMA

```

{karakterKeInteger merupakan fungsi yang merubah suatu karakter menjadi nilai
ASCII nya }
rowRuangan <- karakterKeInteger(kodeRuangan[1]) - karakterKeInteger('A')
columnRuangan <- karakterKeInteger(kodeRuangan[2]) - karakterKeInteger('0')

if (panjangString(kodeRuangan) >= 3) and (isDigit(kodeRuangan[3])) then
    columnRuangan <- (columnRuangan * 10) + (karakterKeInteger(kodeRuangan[3]) -
    karakterKeInteger('0'))

columnRuangan <- columnRuangan - 1

```

```

procedure lihatDenah()
{I.S. Isi file config yang sudah dimasukkan ke variable global
globalDenahRumahSakit}
{F.S. output Denah Rumah sakit di terminal}

```

KAMUS LOKAL

row, column: integer

ALGORITMA

```

output("")
output(" ")
column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
if (column < 9) then
    output("      ", column + 1)
else
    output("      ", column + 1)

output("")

```

```

output(" ")
column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
  output("+-----")
output("+")
output("")

row traversal [0..(globalDenahRumahSakit.nRow - 1)]
  output(" ", karakterDariInteger(row + 65), " ")

  column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
    output("| ", karakterDariInteger(row + 65), column + 1)
    if (column < 9) then
      output(" ")

    output("|")
    output("")

  output(" ")
  column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
    output("+-----")

  output("+")
  output("")

```

procedure lihatRuangan(input row: integer, input column: integer)
 {I.S. Nomor ruangan, isi file config yang sudah dimasukan ke variable global
 globalDenahRumahSakit, serta list dokter dan list pasien}
 {F.S. output kondisi ruangan seperti kapasitas, dokter di ruangan, dan pasien
 dalam ruangan}

KAMUS LOKAL

kodeRuanganInput: string
 rowRuangan, columnRuangan: integer
 idDokter: integer
 namaDokter: string
 forTraverse: Address {Asumsi Address adalah tipe untuk node linked list
 antrian}
 i: integer
 kapasitasTercapai: integer
 prosesLanjut: boolean {Flag untuk menentukan apakah detail ruangan akan
 diproses}

ALGORITMA

```

prosesLanjut <- true { Asumsi awal, detail akan diproses }

if (row = -1) or (column = -1) then
  input(kodeRuanganInput) { Membaca kode ruangan dari pengguna }

  kodeRuanganKonverter(kodeRuanganInput, rowRuangan, columnRuangan)

  if (rowRuangan < 0) or (columnRuangan < 0) or
    (rowRuangan >= globalDenahRumahSakit.nRow) or
    (columnRuangan >= globalDenahRumahSakit.nColumn) then
    output("")
    output("Tidak ada ruangan dengan kode ruangan ", kodeRuanganInput)
    output("")
    prosesLanjut <- false { Kode ruangan tidak valid, jangan proses detail }
  else
    output("")
    output("--- Detail Ruangan ", kodeRuanganInput, " ---")
  else
    rowRuangan <- row
    columnRuangan <- column
    { Format kodeRuangan dari row dan column numerik }
    kodeRuanganInput <- formatString(karakterDariInteger(rowRuangan + 65),
    columnRuangan + 1)
    output("===== ", kodeRuanganInput, " =====")

  if (prosesLanjut) then
    {Bagian ini hanya dieksekusi jika kode ruangan valid dan prosesLanjut adalah
    true}
    output("Kapasitas : ", globalDenahRumahSakit.kapasitasRuangan)
    idDokter <- globalDenahRumahSakit.Ruangan[rowRuangan][columnRuangan].idDokter

    if (idDokter = 0) then

```

```

        output("Dokter      : -")
    else
        namaDokter <- getAccountName(idDokter, DATA_TYPE_DOCTOR)
        output("Dokter      : ", namaDokter)

    output("Pasien di dalam ruangan :")
    if(isQueueEmpty(globalDenahRumahSakit.Ruangan[rowRuangan][columnRuangan].idAntrian)) then
        output(" Tidak ada pasien di dalam ruangan saat ini.")
    else
        forTraverse<-globalDenahRumahSakit.Ruangan[rowRuangan][columnRuangan].idAntrian.front
        if globalDenahRumahSakit.Ruangan[rowRuangan][columnRuangan].idAntrian.size <
            globalDenahRumahSakit.kapasitasRuangan then
            kapasitasTercapai <-
                globalDenahRumahSakit.Ruangan[rowRuangan][columnRuangan].idAntrian.size
        else
            kapasitasTercapai <- globalDenahRumahSakit.kapasitasRuangan

        i traversal [0..(kapasitasTercapai - 1)]
        output(" ", i + 1, ". ", getAccountName(forTraverse.id,
            DATA_TYPE_PATIENT))
        forTraverse <- forTraverse.next

    if (row = -1) or (column = -1) then
        output("-----")
        output("")

```

7. F07

```

procedure lihatUser(input globalUserDatabase: ListDin)
{I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
{F.S. output tabel yang berisi data seluruh user meliputi id, nama,
role(dokter/pasien), dan penyakit}

```

KAMUS LOKAL

```

urt, srt, tabelEff, idxPrinted, i, j, count: integer
save: GenericData*
p: Patient*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]
urutan: array of char[0..12]
sortin: array of char[0..12]

```

ALGORITMA

```

    output("Urutkan berdasarkan?")
    output("1. ID")
    output("2. Nama")
    output(">>> Pilihan: ")

    input(urt)

    output("Urutan sorting?")
    output("1. ASC (A-Z)")
    output("2. DESC (Z-A)")
    output(">>> Pilihan: ")

    input(srt)

    tabelEff <- 0
    idxPrinted <- 0

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]
        if(save.type = DATA_TYPE_PATIENT) then
            tabelEff <- tabelEff + 1
        else if(save.type = DATA_TYPE_DOCTOR) then
            tabelEff <- tabelEff + 1

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]

```

```

if(save.type = DATA_TYPE_PATIENT) then
  p <- (Patient*)(save.data)
  printed[idxPrinted].ID <- p.id
  printed[idxPrinted].nama <- p.username
  printed[idxPrinted].role <- 'PASIEN'

  count <- 0
  while(printed[idxPrinted].penyakit[count] = '' || count < 50)do
    count++
  if(count != 50) then
    printed[idxPrinted].penyakit <- p.rwayatpenyakit
  else
    printed[idxPrinted].penyakit <- "(Belum diperiksa dokter)"

  idxPrinted <- idxPrinted + 1

else if(save.type = DATA_TYPE_DOCTOR) then
  d <- (Doctor*)(save.data)
  printed[idxPrinted].ID <- d.id
  printed[idxPrinted].nama <- d.username
  printed[idxPrinted].role <- 'DOKTER'
  printed[idxPrinted].penyakit <- '-'
  idxPrinted <- idxPrinted + 1

if(urut = 1) then
  urutan <- "ID"
  if(srt = 1) then
    sortin <- "ascending"
    i traversal[0..tabelEff - 1]
    j traversal[0..tabelEff - i - 1]
    if(printed[j].ID > printed[j + 1].ID) then
      temp <- printed[j + 1]
      printed[j + 1] <- printed[j]
      printed[j] <- temp
  else if(srt = 2) then
    sortin <- "descending"
    i traversal[0..tabelEff - 1]
    j traversal[0..tabelEff - i - 1]
    if(printed[j].ID < printed[j + 1].ID) then
      temp <- printed[j + 1]
      printed[j + 1] <- printed[j]
      printed[j] <- temp
  else if(urut = 2) then
    urutan <- "nama"
    if(srt = 1) then
      sortin <- "ascending"
      i traversal[0..tabelEff - 1]
      j traversal[0..tabelEff - i - 1]
      if(printed[j].nama > printed[j + 1].nama) then
        temp <- printed[j + 1]
        printed[j + 1] <- printed[j]
        printed[j] <- temp
    else if(srt = 2) then
      sortin <- "descending"
      i traversal[0..tabelEff - 1]
      j traversal[0..tabelEff - i - 1]
      if(printed[j].nama > printed[j + 1].nama) then
        temp <- printed[j + 1]
        printed[j + 1] <- printed[j]
        printed[j] <- temp

output("Menampilkan data seluruh user berdasarkan", urutan, "terurut", sortin,
"...")
output("+-----+-----+-----+-----+-----+")
output("| ID      | Nama      | Role      | Penyakit  |")
output("+-----+-----+-----+-----+-----+")
i traversal[0..tabelEff - 1]
  output("|", printed[i].ID, "|", printed[i].nama, "|",
    printed[i].role, "|",
    printed[i].penyakit)
output("+-----+-----+-----+-----+-----+")
output("")

```

```

procedure lihatPasien(input globalUserDatabase: ListDin)
{I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
{F.S. output tabel yang berisi data seluruh pasien meliputi id, nama, dan

```

```
penyakit}
```

KAMUS_LOKAL

```
urt, srt, tabelEff, idxPrinted, i, j, count: integer  
save: GenericData*  
p: Patient*  
printed: array of tabel[0..tabelEff - 1]  
urutan: array of char[0..12]  
sortin: array of char[0..12]
```

ALGORITMA

```
output("Urutkan berdasarkan?")  
output("1. ID")  
output("2. Nama")  
output(">>> Pilihan: ")  
  
input(urt)  
  
output("Urutan sorting?")  
output("1. ASC (A-Z)")  
output("2. DESC (Z-A)")  
output(">>> Pilihan: ")  
  
input(srt)  
  
tabelEff <- 0  
idxPrinted <- 0  
  
i traversal[0..globalUserDatabase.nEff]  
  save <- globalUserDatabase.buffer[i]  
  if(save.type = DATA_TYPE_PATIENT) then  
    tabelEff <- tabelEff + 1  
  
i traversal[0..globalUserDatabase.nEff]  
  save <- globalUserDatabase.buffer[i]  
  
  if(save.type = DATA_TYPE_PATIENT) then  
    p <- (Patient*)(save.data)  
    printed[idxPrinted].ID <- p.id  
    printed[idxPrinted].nama <- p.username  
    printed[idxPrinted].role <- 'PASIEN'  
  
    count <- 0  
    while(printed[idxPrinted].penyakit[count] = '\\0' || count < 50) do  
      count++  
    if(count != 50) then  
      printed[idxPrinted].penyakit <- p.rwayatpenyakit  
    else  
      printed[idxPrinted].penyakit <- "(Belum diperiksa dokter)"  
  
    idxPrinted <- idxPrinted + 1  
  
if(urt = 1) then  
  urutan <- "ID"  
  if(srt = 1) then  
    sortin <- "ascending"  
    i traversal[0..tabelEff - 1]  
      j traversal[0..tabelEff - i - 1]  
        if(printed[j].ID > printed[j + 1].ID) then  
          temp <- printed[j + 1]  
          printed[j + 1] <- printed[j]  
          printed[j] <- temp  
    else if(srt = 2) then  
      sortin <- "descending"  
      i traversal[0..tabelEff - 1]  
        j traversal[0..tabelEff - i - 1]  
          if(printed[j].ID < printed[j + 1].ID) then  
            temp <- printed[j + 1]  
            printed[j + 1] <- printed[j]  
            printed[j] <- temp  
  else if(urt = 2) then  
    urutan <- "nama"  
    if(srt = 1) then  
      sortin <- "ascending"  
      i traversal[0..tabelEff - 1]
```

```

        j traversal[0..tabelEff - i - 1]
        if(printed[j].nama > printed[j + 1].nama) then
            temp <- printed[j + 1]
            printed[j + 1] <- printed[j]
            printed[j] <- temp
    else if(srt = 2) then
        sortin <- "descending"
        i traversal[0..tabelEff - 1]
        j traversal[0..tabelEff - i - 1]
        if(printed[j].nama > printed[j + 1].nama) then
            temp <- printed[j + 1]
            printed[j + 1] <- printed[j]
            printed[j] <- temp

output("Menampilkan data pasien berdasarkan", urutan, "terurut", sortin, "...")
output("+-----+-----+-----+")
output("| ID      | Nama      | Penyakit      |")
output("+-----+-----+-----+")
i traversal[0..tabelEff - 1]
    output("|", printed[i].ID, "|", printed[i].nama, "|", printed[i].penyakit, "|")
output("+-----+-----+-----+")
output("")

```

```

procedure lihatDokter(input globalUserDatabase: ListDin)
{I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
{F.S. output tabel yang berisi data dokter meliputi id dan nama}

```

KAMUS LOKAL

```

urt, srt, tabelEff, idxPrinted, i, j: integer
save: GenericData*
d: Doctor*
printed: array of tabel[0..tabelEff - 1]
urutan: array of char[0..12]
sortin: array of char[0..12]

```

ALGORITMA

```

output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")

input(urt)

output("Urutan sorting?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")

input(srt)

tabelEff <- 0
idxPrinted <- 0

i traversal[0..globalUserDatabase.nEff]
    save <- globalUserDatabase.buffer[i]
    if(save.type == DATA_TYPE_DOCTOR)
        tabelEff <- tabelEff + 1

i traversal[0..globalUserDatabase.nEff]
    save <- globalUserDatabase.buffer[i]

    if(save.type == DATA_TYPE_DOCTOR)
        d <- (Doctor*)(save.data)
        printed[idxPrinted].ID <- d.id
        printed[idxPrinted].nama <- d.username
        printed[idxPrinted].role <- 'DOKTER'
        printed[idxPrinted].penyakit <- '-'
        idxPrinted <- idxPrinted + 1

if(urt = 1) then
    urutan <- "ID"
    if(srt = 1) then
        sortin <- "ascending"
        i traversal[0..tabelEff - 1]

```

```

        j traversal[0..tabelEff - i - 1]
        if(printed[j].ID > printed[j + 1].ID) then
            temp <- printed[j + 1]
            printed[j + 1] <- printed[j]
            printed[j] <- temp
    else if(srt = 2) then
        sortin <- "descending"
        i traversal[0..tabelEff - 1]
        j traversal[0..tabelEff - i - 1]
        if(printed[j].ID < printed[j + 1].ID) then
            temp <- printed[j + 1]
            printed[j + 1] <- printed[j]
            printed[j] <- temp
    else if(urt = 2) then
        urutan <- "nama"
        if(srt = 1) then
            sortin <- "ascending"
            i traversal[0..tabelEff - 1]
            j traversal[0..tabelEff - i - 1]
            if(printed[j].nama > printed[j + 1].nama) then
                temp <- printed[j + 1]
                printed[j + 1] <- printed[j]
                printed[j] <- temp
        else if(srt = 2) then
            sortin <- "descending"
            i traversal[0..tabelEff - 1]
            j traversal[0..tabelEff - i - 1]
            if(printed[j].nama > printed[j + 1].nama) then
                temp <- printed[j + 1]
                printed[j + 1] <- printed[j]
                printed[j] <- temp

    output("Menampilkan data dokter berdasarkan", urutan, "terurut", sortin, "...")
    output("+-----+-----+")
    output("| ID      | Nama          |")
    output("+-----+-----+")
    i traversal[0..tabelEff - 1]
    output("|", printed[i].ID, "|", printed[i].nama, "|")
    output("+-----+-----+")
    output("")

```

8. F08

procedure cariUser(input globalUserDatabase: ListDin)
 {I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
 {F.S. output tabel yang berisi data user yang dicari(baik berdasarkan ID maupun nama) meliputi id, nama, role(dokter/pasien), dan penyakit}

KAMUS_LOKAL

eff, src, tabelEff, idxPrinted, count, i: integer
 idx, mid, low, high, cari: integer
 save: GenericData*
 p: Patient*
 d: Doctor*
 printed: array of tabel[0..tabelEff - 1]
 search: array of char[0..50]

ALGORITMA

```

output("Cari berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")

input(src)
tabelEff <- 0
idxPrinted <- 0

i traversal[0..globalUserDatabase.nEff]
    save <- globalUserDatabase.buffer[i]
    if(save.type = DATA_TYPE_PATIENT) then
        tabelEff <- tabelEff + 1
    else if(save.type = DATA_TYPE_DOCTOR)

```



```

        tabelEff <- tabelEff + 1

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]

        if(save.type = DATA_TYPE_PATIENT) then
            p <- (Patient*)(save.data)
            printed[idxPrinted].ID <- p.id
            printed[idxPrinted].nama <- p.username
            printed[idxPrinted].role <- 'PASIEN'

            eff <- 0
            while(printed[idxPrinted].penyakit[eff] = '\0' || count < 50)do
                eff++
            if(eff != 50) then
                printed[idxPrinted].penyakit <- p.rwayatpenyakit
            else
                printed[idxPrinted].penyakit <- "(Belum diperiksa dokter)"

            idxPrinted <- idxPrinted + 1

        else if(save.type = DATA_TYPE_DOCTOR)
            d <- (Doctor*)(save.data)
            printed[idxPrinted].ID <- d.id
            printed[idxPrinted].nama <- d.username
            printed[idxPrinted].role <- 'DOKTER'
            printed[idxPrinted].penyakit <- '-'
            idxPrinted <- idxPrinted + 1

    if(src = 1) then
        output("Masukkan nomor ID user: ")
        input(cari)
        i traversal[0..tabelEff - 1]
            j traversal[0..tabelEff - i - 1]
                if(printed[j].ID > printed[j + 1].ID) then
                    temp <- printed[j + 1]
                    printed[j + 1] <- printed[j]
                    printed[j] <- temp
        high <- tabelEff - 1
        low <- 0
        idx <- -1
        while(low <= high) do
            mid <- (high + low)/2
            if(printed[mid].ID = cari) then
                idx <- mid
                break
            else if(cari < printed[mid].ID) then
                high <- mid - 1
            else
                low <- mid + 1
        if(idx = -1) then
            output("Tidak ditemukan pengguna dengan nomor ID ", cari, "!")
        else
            output("Menampilkan pengguna dengan nomor ID ", cari, "...")
            output("+-----+-----+-----+-----+")
            output("| ID      | Nama      | Role      | Penyakit      |")
            output("+-----+-----+-----+-----+")
            output("|", printed[idx].ID, "|", printed[idx].nama, "|",
                printed[idx].role, "|", printed[idx].penyakit, "|")
            output("+-----+-----+-----+-----+")

    else if(src = 2) then
        output("Masukkan nama user: ")
        input(search)
        count <- 0
        i traversal [0..tabelEff - 1]
            if(printed[i].nama = search) then
                count++
        if(count < 1) then
            output("Tidak ditemukan pengguna dengan nama", search, "!")
        else
            output("Menampilkan pengguna dengan nama ", search, "...")
            output("+-----+-----+-----+-----+")
            output("| ID      | Nama      | Role      | Penyakit      |")
            output("+-----+-----+-----+-----+")

```

```

        output("|", printed[idx].ID, "|", printed[idx].nama, "|",
               printed[idx].role, "|", printed[idx].penyakit, "|")
        output("+-----+-----+-----+-----+-----+")
    output("")

```

```

procedure cariPasien(input globalUserDatabase: ListDin)
{I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
{F.S. output tabel yang berisi data pasien yang dicari(baik berdasarkan ID maupun
nama) meliputi id, nama, dan penyakit}

```

KAMUS_LOKAL

```

    eff, src, tabelEff, idxPrinted, count, i, j: integer
    idx, mid, low, high, cari, urt, srt: integer
    save: GenericData*
    p: Patient*
    printed: array of tabel[0..tabelEff - 1]
    arr: array of tabel[0..count - 1]
    search1: array of char[0..50]
    search2: array of char[0..50]
    urutan: array of char[0..12]
    sortin: array of char[0..12]

```

ALGORITMA

```

    output("Cari berdasarkan?")
    output("1. ID")
    output("2. Nama")
    output("3. Penyakit")
    output(">>> Pilihan: ")

    input(src)
    tabelEff <- 0
    idxPrinted <- 0

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]
        if(save.type = DATA_TYPE_PATIENT) then
            tabelEff <- tabelEff + 1

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]

        if(save.type = DATA_TYPE_PATIENT) then
            p <- (Patient*)(save.data)
            printed[idxPrinted].ID <- p.id
            printed[idxPrinted].nama <- p.username
            printed[idxPrinted].role <- 'PASIEN'

            eff <- 0
            while(printed[idxPrinted].penyakit[eff] = '\0' || count < 50) do
                eff++
            if(eff != 50) then
                printed[idxPrinted].penyakit <- p.rwayatpenyakit
            else
                printed[idxPrinted].penyakit <- "(Belum diperiksa dokter)"

            idxPrinted <- idxPrinted + 1

    if(src = 1) then
        output("Masukkan nomor ID pasien: ")
        input(cari)
        i traversal[0..tabelEff - 1]
            j traversal[0..tabelEff - i - 1]
                if(printed[j].ID > printed[j + 1].ID) then
                    temp <- printed[j + 1]
                    printed[j + 1] <- printed[j]
                    printed[j] <- temp
            high <- tabelEff - 1
            low <- 0
            idx <- -1
            while(low <= high) do
                mid <- (high + low)/2
                if(printed[mid].ID = cari) then
                    idx <- mid

```

```

        break
    else if(cari < printed[mid].ID) then
        high <- mid - 1
    else
        low <- mid + 1
if(idx = -1) then
    output("Tidak ditemukan pengguna dengan nomor ID ", cari, "!")
else
    output("Menampilkan pengguna dengan nomor ID ", cari, "...")
    output("+-----+-----+-----+-----+-----+-----+")
    output("| ID      | Nama          | Role      | Penyakit          |")
    output("+-----+-----+-----+-----+-----+-----+")
    output("|", printed[idx].ID, "|", printed[idx].nama, "|",
        printed[idx].role, "|", printed[idx].penyakit, "|")
    output("+-----+-----+-----+-----+-----+-----+")

else if(src = 2) then
    output("Masukkan nama pasien: ")
    input(search1)
    count <- 0
    i traversal[0..tabelEff - 1]
    if(printed[i].nama = search1) then
        count++
    if(count < 1) then
        output("Tidak ditemukan pengguna dengan nama", search, "!")
    else
        output("Menampilkan pengguna dengan nama ", search, "...")
        output("+-----+-----+-----+-----+-----+-----+")
        output("| ID      | Nama          | Role      | Penyakit          |")
        output("+-----+-----+-----+-----+-----+-----+")
        output("|", printed[idx].ID, "|", printed[idx].nama, "|",
            printed[idx].role, "|", printed[idx].penyakit, "|")
        output("+-----+-----+-----+-----+-----+-----+")

else if(src = 3) then
    output("Masukkan nama penyakit: ")
    input(search2)
    count <- 0
    i traversal[0..tabelEff - 1]
    if(printed[i].penyakit = search2) then
        count++
    if(count < 1) then
        output("Tidak ditemukan pengguna dengan nama", search, "!")
    else if(count = 1)
        output("Menampilkan pengguna dengan nama ", search, "...")
        output("+-----+-----+-----+-----+-----+-----+")
        output("| ID      | Nama          | Role      | Penyakit          |")
        output("+-----+-----+-----+-----+-----+-----+")
        output("|", printed[idx].ID, "|", printed[idx].nama, "|",
            printed[idx].role, "|", printed[idx].penyakit, "|")
        output("+-----+-----+-----+-----+-----+-----+")
    else
        i <- 0
        j traversal[0..tabelEff]
        if(printed[j].penyakit = search2) then
            arr[i] <- printed[j]
            i++
        output("Urutkan berdasarkan?")
        output("1. ID")
        output("2. Nama")
        output(">>> Pilihan: ")

        input(urt)

        output("Urutan sorting?")
        output("1. ASC (A-Z)")
        output("2. DESC (Z-A)")
        output(">>> Pilihan: ")

        input(srt)

        if(urt = 1) then
            urutan <- "ID"
            if(srt = 1) then
                sortin <- "ascending"

```

```

        i traversal[0..count - 1]
        j traversal[0..count - i - 1]
        if(arr[j].ID > arr[j + 1].ID) then
            temp <- arr[j + 1]
            arr[j + 1] <- arr[j]
            arr[j] <- temp
    else if(srt = 2) then
        sortin <- "descending"
        i traversal[0..count - 1]
        j traversal[0..count - i - 1]
        if(arr[j].ID < arr[j + 1].ID) then
            temp <- arr[j + 1]
            arr[j + 1] <- arr[j]
            arr[j] <- temp
    else if(urt = 2) then
        urutan <- "nama"
        if(srt = 1) then
            sortin <- "ascending"
        i traversal[0..count - 1]
        j traversal[0..count - i - 1]
        if(arr[j].nama > arr[j + 1].nama) then
            temp <- printed[j + 1]
            arr[j + 1] <- arr[j]
            arr[j] <- temp
        else if(srt = 2) then
            sortin <- "descending"
        i traversal[0..count - 1]
        j traversal[0..count - i - 1]
        if(arr[j].nama > arr[j + 1].nama) then
            temp <- arr[j + 1]
            arr[j + 1] <- arr[j]
            arr[j] <- temp
    output("Menampilkan data pasien dengan penyakit ", search2,
"berdasarkan",
        urutan, "terurut", sortin, "...")
    output("+-----+-----+-----+-----+")
    output("| ID      | Nama      | Penyakit      |")
    output("+-----+-----+-----+-----+")
    i traversal[0..tabelEff - 1]
    output("|", printed[i].ID, "|", printed[i].nama,"|",
        printed[i].penyakit, "|")
    output("+-----+-----+-----+-----+")

output("")

```

procedure cariUser(input globalUserDatabase: ListDin)
 {I.S. globalUserDatabase yang terdefinisi dan tidak kosong}
 {F.S. output tabel yang berisi data user yang dicari(baik berdasarkan ID maupun nama) meliputi id, nama, role(dokter/pasien), dan penyakit}

KAMUS_LOKAL

eff, src, tabelEff, idxPrinted, count, i: integer
 idx, mid, low, high, cari: integer
 save: GenericData*
 d: Doctor*
 printed: array of tabel[0..tabelEff - 1]
 search: array of char[0..50]

ALGORITMA

```

    output("CAri berdasarkan?")
    output("1. ID")
    output("2. Nama")
    output(">>> Pilihan: ")

    input(src)
    tabelEff <- 0
    idxPrinted <- 0

    i traversal[0..globalUserDatabase.nEff]
        save <- globalUserDatabase.buffer[i]
        if(save.type = DATA_TYPE_DOCTOR)
            tabelEff <- tabelEff + 1

    i traversal[0..globalUserDatabase.nEff]

```

```

save <- globalUserDatabase.buffer[i]

if(safe.type = DATA_TYPE_DOCTOR)
d <- (Doctor*)(save.data)
printed[idxPrinted].ID <- d.id
printed[idxPrinted].nama <- d.username
printed[idxPrinted].role <- 'DOKTER'
printed[idxPrinted].penyakit <- '-'
idxPrinted <- idxPrinted + 1

if(src = 1) then
  output("Masukkan nomor ID dokter: ")
  input(cari)
  i traversal[0..tabelEff - 1]
  j traversal[0..tabelEff - i - 1]
  if(printed[j].ID > printed[j + 1].ID) then
    temp <- printed[j + 1]
    printed[j + 1] <- printed[j]
    printed[j] <- temp
  high <- tabelEff - 1
  low <- 0
  idx <- -1
  while(low <= high) do
    mid <- (high + low)/2
    if(printed[mid].ID = cari) then
      idx <- mid
      break
    else if(cari < printed[mid].ID) then
      high <- mid - 1
    else
      low <- mid + 1
  if(idx = -1) then
    output("Tidak ditemukan dokter dengan nomor ID ", cari, "!")
  else
    output("Menampilkan dokter dengan nomor ID ", cari, "...")
    output("+-----+-----+")
    output("| ID      | Nama      |")
    output("+-----+-----+")
    output("|", printed[idx].ID, "|", printed[idx].nama, "|")
    output("+-----+-----+")

else if(src = 2) then
  output("Masukkan nama dokter: ")
  input(search)
  count <- 0
  i traversal [0..tabelEff - 1]
  if(printed[i].nama = search) then
    count++
  if(count < 1) then
    output("Tidak ditemukan dokter dengan nama", search, "!")
  else
    output("Menampilkan dokter dengan nama ", search, "...")
    output("+-----+-----+")
    output("| ID      | Nama      |")
    output("+-----+-----+")
    output("|", printed[idx].ID, "|", printed[idx].nama, "|")
    output("+-----+-----+")

output("")

```

9. F09

```

procedure lihatSemuaAntrian()
{I.S. Kondisi rumah sakit yang disimpan di dalam globahDenahRumahSakit}
{F.S. Menampilkan seluruh kondisi ruangan yang tidak kosong beserta antrian nya}

```

KAMUS LOKAL

```

row, column, i: integer
forTraverse: Address
pasienDiAntrianKe: integer

```

ALGORITMA

```

lihatDenah()
output("")

row traversal [0..(globalDenahRumahSakit.nRow - 1)]
column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
if (globalDenahRumahSakit.Ruangan[row][column].idDokter != 0) then
    lihatRuangan(row, column)

    output("Pasien di antrian:")
    if (globalDenahRumahSakit.Ruangan[row][column].idAntrian.size <=
        globalDenahRumahSakit.kapasitasRuangan) then
        output(" Tidak ada pasien di antrian saat ini.")
    else
        forTraverse <-
globalDenahRumahSakit.Ruangan[row][column].idAntrian.front
        pasienDiAntrianKe <- 0
        i traversal
[0..(globalDenahRumahSakit.Ruangan[row][column].idAntrian.size
- 1)]
        if (i < globalDenahRumahSakit.kapasitasRuangan) then
            forTraverse <- forTraverse.next
        else
            pasienDiAntrianKe <- pasienDiAntrianKe + 1
            output(" ", pasienDiAntrianKe, ". ",
getAccountName(forTraverse.id,
DATA_TYPE_PATIENT))
            forTraverse <- forTraverse.next
        output("")

```

10. F10

```

procedure tambahDokter(input/output globalUserDatabase: ListDin, input/output
usernameSet: Set)
{I.S. globalUserDatabase dan usernameSet terdefinisi}
{F.S. Seorang dokter baru ditambahkan ke database jika username valid dan unik}

```

KAMUS LOKAL

```

nama, username, password, confirm: array of char[0..50]
d: Doctor
newData: GenericData
valid: boolean

```

ALGORITMA

```

output("Masukkan nama: ")
(nama)

valid <- false
while(not valid) do
    output("Masukkan username: ")
    (username)

    toLowerString(username)
    if(usernameSet.contains(username)) then
        output("Username sudah terpakai. Silakan masukkan username lain.")
    else
        valid <- true

output("Masukkan password: ")
(password)

output("Konfirmasi password: ")
(confirm)

while(password != confirm) do
    output("Password tidak cocok. Silakan ulangi.")
    output("Masukkan password: ")
    (password)
    output("Konfirmasi password: ")
    (confirm)

d.id <- generateNewId(globalUserDatabase)
d.username <- username
d.password <- password

```

```

d.nama <- nama

newData.type <- DATA_TYPE_DOCTOR
newData.data <- &d

insertLast(globalUserDatabase, newData)
insert(usernameSet, toLowerString(username))

output("Akun dokter berhasil ditambahkan!")

```

procedure assignDokter(input/output globalUserDatabase: ListDin)
 {I.S. globalUserDatabase terdefinisi, minimal ada 1 pasien dan 1 dokter}
 {F.S. Dokter ditugaskan untuk menangani pasien yang belum memiliki riwayat penyakit}

KAMUS_LOKAL

```

i, j: integer
pasienDitemukan, dokterDitemukan: boolean
p: Patient*
d: Doctor*
foundPasien: Patient*
foundDokter: Doctor*

```

ALGORITMA

```

pasienDitemukan <- false
dokterDitemukan <- false

i traversal [0..globalUserDatabase.nEff - 1]
  if (globalUserDatabase.buffer[i].type = DATA_TYPE_PATIENT) then
    p <- (Patient*)(globalUserDatabase.buffer[i].data)
    if (p.riwayatpenyakit[0] = '\0') then
      foundPasien <- p
      pasienDitemukan <- true
      break

if (not pasienDitemukan) then
  output ("Semua pasien sudah memiliki riwayat penyakit.")
  return

j traversal [0..globalUserDatabase.nEff - 1]
  if (globalUserDatabase.buffer[j].type = DATA_TYPE_DOCTOR) then
    d <- (Doctor*)(globalUserDatabase.buffer[j].data)
    foundDokter <- d
    dokterDitemukan <- true
    break

if (not dokterDitemukan) then
  output ("Tidak ada dokter yang tersedia.")
  return

output ("Pasien yang ditemukan:")
output ("ID :", foundPasien.id)
output ("Nama :", foundPasien.username)

output ("Masukkan penyakit yang diderita pasien: ")
(foundPasien.riwayatpenyakit)

output ("Dokter ", foundDokter.nama, " berhasil ditugaskan ke pasien ",
foundPasien.username, ".")

```

11. F11

Procedure diagnosis(Input/Output Q: Queue of Patient)
 {I.S globalPenyakitDatabase terdefinisi dan memiliki daftar penyakit dengan nilai min dan max }
 {F.S output penyakit yang terdiagnosis, kondisi pasien ada penyakit atau

tidak }.

KAMUS LOKAL

pasien : Patient
i, j : integer
nilai, batasMin, batasMax : real
cocok : boolean

ALGORITMA

```
If (IsEmpty(Q)) then  
    output("Tidak ada pasien untuk diperiksa!")  
    return  
  
pasien ← getPatientFromNode(Front(Q))  
  
If (pasien = NULL) then  
    Output("Pasien tidak ditemukan")  
    return  
  
If (pasien.sudahDiDiagnosis) then  
    Output(pasien.username + " Telah Didiagnosa")  
    return  
  
If (pasien.riwayatPenyakit ≠ "") then  
    Output(pasien.username + " terdiagnosa penyakit " +  
pasien.riwayatPenyakit)  
    pasien.sudahDiDiagnosis ← true  
    return  
  
cocok ← false  
For (i ← 0 to globalPenyakitDatabase.nEff - 1) do  
    penyakit ← globalPenyakitDatabase.contents[i]  
    cocok ← true  
  
    For (j ← 0 to KONDISI_TUBUH_SIZE - 1) do  
        nilai ← pasien.kondisiTubuh[j]  
        batasMin ← penyakit.threshold[j * 2]  
        batasMax ← penyakit.threshold[j * 2 + 1]  
  
        If (nilai < batasMin) or (nilai > batasMax) then  
            cocok ← false  
            break  
  
    If (cocok) then  
        pasien.riwayatPenyakit ← penyakit.name  
        output(pasien.username + " terdiagnosa penyakit " +  
pasien.riwayatPenyakit)  
        pasien.sudahDiDiagnosis ← true  
        return  
  
    output(pasien.username + " tidak terdiagnosis penyakit apapun!")  
    pasien.sudahDiDiagnosis ← true
```


Procedure ngobatin (Input/Output Q : Queue of Patient
{I.S database globalPenyakitDatabase terdefinisi, antrian Q mungkin berisi pasien}
{F.S Output pasien menerima obat sesuai penyakit, obat tersimpan di inventory, informasi pasien}

KAMUS LOKAL

pasien : Patient
idPenyakit : integer
i, j, l : integer
found : boolean
idObat : integer
obat : Obat

ALGORITMA

```
If (IsEmpty(Q)) then  
    output("Tidak ada pasien untuk diobatin!")  
    return  
  
pasien ← GetPatientFromNode(Front(Q))  
  
If (pasien = NULL) then  
    return  
  
If (pasien.sudahDiDiagnosis = false) then  
    output("Pasien belum menerima diagnosis!")  
    return  
  
if (pasien.sudahDiObatin = true) then  
    output("Pasien sudah diobatin!")  
    return  
  
idPenyakit ← -1  
for (i ← 0 to globalPenyakitDatabase.nEff - 1) do  
    if (globalPenyakitDatabase.contents[i].name =  
pasien.rwayatPenyakit) then  
        idPenyakit ← globalPenyakitDatabase.contents[i].id  
  
if (idPenyakit = -1) then  
    output("Penyakit tidak ditemukan dalam database!")  
    return  
  
found ← false  
for (i ← 0 to globalOPDatabase.nEff - 1) do  
    if (globalOPDatabase.contents[i].idPenyakit = idPenyakit) and  
(not(found)) then  
        found ← true  
        output("Dokter sedang mengobati pasien " + pasien.username)  
        output("Pasien memiliki penyakit " + pasien.rwayatPenyakit)  
        output("Obat yang harus diberikan:")  
  
        for (j ← 0 to globalOPDatabase.contents[i].nEff - 1) do  
            idObat ← globalOPDatabase.contents[i].idObat[j]  
            obat ← GetObatById(idObat)  
  
            if(obat ≠ NULL) then
```

```

        l ← 0
        While (l < INVENTORY_SIZE) and (pasien.inventory[l] ≠
UNDEF_INT_DATA) do
            l ← l + 1
            pasien.inventory[l] ← idObat
            output(j + 1 + ". " + obat.name)
        else
            output(j + 1 + ". Obat dengan ID " + idObat + " tidak
ditemukan!")

    if (not(found)) then
        output("Tidak ada daftar obat untuk penyakit ini.")

    pasien.sudahDiObatin ← true

```

13. F13

Procedure pulangDok (Input/Output Q : Queue of Patient)
 {I.S antrian Q terdefinisi mungkin kosong}
 {F.S Output pasien diperbolehkan pulang atau belum dengan ketentuan
 seperti urutan obatnya salah, belum minum obat atau belum didiagnosis}

KAMUS LOKAL

idPenyakit : integer
 i : integer
 urutanBenar : boolean
 op : pointer to ObatPenyakit

ALGORITMA

```

    if (globalCurrentPatient = NULL) then
        output("Pasien tidak ditemukan dalam database!")
        return

    if (globalCurrentPatient.sudahDiDiagnosis = false) then
        output("Kamu belum menerima diagnosis apapun dari dokter, jangan
buru-buru pulang!")
        return

    if (globalCurrentPatient.sudahDiObatin = false) then
        output("Kamu belum menerima obat dari dokter, jangan buru buru
pulang!")
        return

    output("Dokter sedang memeriksa keadaanmu...")

    idPenyakit ← -1
    For (i ← 0 to globalPenyakitDatabase.nEff - 1) do
        if (globalPenyakitDatabase.contents[i].name =
globalCurrentPatient.riwayatPenyakit) then
            idPenyakit ← globalPenyakitDatabase.contents[i].id

```

```

if (idPenyakit = -1) then
    output("Penyakit tidak ditemukan dalam database!")
    return

op ← NULL
For (i ← 0 to globalOPDatabase.nEff - 1) do
    if (globalOPDatabase.contents[i].idPenyakit = idPenyakit) then
        op ← &globalOPDatabase.contents[i]

If (op = NULL) then
    output("Data obat tidak ditemukan untuk penyakit ini!")
    return

if (stackSize(globalCurrentPatient.perut) < op.nEff) then
    output("Masih ada obat yang belum kamu habiskan, minum semuanya dulu yukk!")
    return

urutanBenar ← true
For (i ← 0 to op.nEff - 1) do
    if (globalCurrentPatient.perut.obat[i].id ≠ op.idObat[i]) then
        urutanBenar ← false

If (urutanBenar) then
    output("Selamat! Kamu sudah dinyatakan sembuh oleh dokter.
    Silahkan pulang dan semoga sehat selalu!")
    resetPatientData()
    deQueue(Q)
    return

output("Maaf, tapi kamu masih belum bisa pulang!")
output("Urutan peminuman obat yang diharapkan:")
For (i ← 0 to op.nEff - 1) do
    output(globalObatDatabase.contents[op.idObat[i]].name)
    if (i < op.nEff - 1) then
        output(" -> ")

output("Urutan obat yang kamu minum:")
For (i ← 0 to op.nEff - 1) do
    output(globalCurrentPatient.perut.obat[i].name)
    if (i < stackSize(globalCurrentPatient.perut) - 1) then
        output(" -> ")

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

```

14. F14

```

procedure daftarCheckup(input globalCurrentUserGD: GenericData)
{I.S. globalCurrentUserGD yang terdefinisi}
{F.S. Apabila pengguna belum terdaftar dalam antrian, program akan menginput data
kondisi tubuh pengguna dan memasukkannya ke dalam antrian, dan apabila sudah
masuk dalam antrian, program akan menampilkan pesan yang menunjukkan kalau
pengguna sudah terdaftar}

KAMUS LOKAL
    lokasiRuangan: Point

```

```

globalUserDatabase: ListDin
globalDenahRumahSakit: DataTypeDenah
count, col, row, idx, idxtemp, idokter, i, j: integer
rowtemp: array of integer[0..100]
coltemp: array of integer[0..100]
pasien: LinkedListNode*
baris: char
name: array of array of char[0..100][0..100]
spes: array of array of char[0..100][0..100]

```

ALGORITMA

```

if (lokasiRuangan.antrian = -1) then
    output("Silahkan masukkan data checkup anda")
    output("Suhu tubuh (celecius): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[0])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[0] < 1) do
        output("Suhu tubuh harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[0])
    output("Tekanan darah (sistol/diastol, contoh 120 80: ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[1],
    ((Patient*)globalCurrentUserGD.data).kondisiTubuh[2])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[1] < 1 or
    ((Patient*)globalCurrentUserGD.data).kondisiTubuh[2]) do
        output("Tekanan darah harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[1],
        ((Patient*)globalCurrentUserGD.data).kondisiTubuh[2])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[1] <
    ((Patient*)globalCurrentUserGD.data).kondisiTubuh[2]) do
        output("Tekanan sistolik harus lebih besar dibanding diastolik!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[1],
        ((Patient*)globalCurrentUserGD.data).kondisiTubuh[2])
    output("Detak jantung (bpm): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[3])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[3] < 1) do
        output("Detak jantung harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[3])
    output("Saturasi oksigen: ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[4])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[4] < 1 or
    ((Patient*)globalCurrentUserGD.data).kondisiTubuh[4] > 100) do
        output("Saturasi oksigen harus dalam rentang 1 - 100!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[4])
    output("Kadar gula darah (mg/dL): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[5])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[5] < 1) do
        output("Kadar gula darah harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[5])
    output("Berat badan (kg): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[6])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[6] < 1) do
        output("Berat badan harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[6])
    output("Tinggi badan (cm): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[7])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[7] < 1) do
        output("Tinggi badan harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[7])
    output("Kadar kolesterol (mg/dL): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[8])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[8] < 1) do
        output("Kadar kolesterol harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[8])
    output("Trombosit (ribu/ $\mu$ L): ")
    input((Patient*)globalCurrentUserGD.data).kondisiTubuh[9])
    while((Patient*)globalCurrentUserGD.data).kondisiTubuh[9] < 1) do
        output("Trombosit harus berupa angka positif!")
        input((Patient*)globalCurrentUserGD.data).kondisiTubuh[9])

count <- 0
row traversal[0..globalDenahRumahSakit.nRow]
col traversal[0..globalDenahRumahSakit.nColumn]
if (globalDenahRumahSakit.Ruangan[row][col].idAntrian.size <
globalDenahRumahSakit.Ruangan[row][col].idAntrian.capacity and
globalDenahRumahSakit.Ruangan[row][col].idDokter != 0) then
    count++

```

```

    if(count >= 1) then
        output("Berikut adalah daftar dokter yang tersedia: ")
        idxtemp <- 0
        row traversal[0..globalDenahRumahSakit.nRow]
        col traversal[0..globalDenahRumahSakit.nColumn]
        if(globalDenahRumahSakit.Ruangan[row][col].idAntrian.size <
            globalDenahRumahSakit.Ruangan[row][col].idAntrian.capacity and
            globalDenahRumahSakit.Ruangan[row][col].idDokter != 0) then
            idokter <- globalDenahRumahSakit.Ruangan[row][col].idDokter
            rowtemp[idxtemp] <- row
            coltemp[idxtemp] <- col

            i traversal[0..globalUserDatabase.Neff]
            if(globalUserDatabase.buffer[i]->type = DATA_TYPE_DOCTOR)
then
                if(((Doctor*)(globalUserDatabase.buffer[i]).data).id =
                    idokter
                    j traversal[0..MAX_CAPACITY]
                    name[idxtemp][j] <- ((Doctor*)(globalUserDatabase.
                        buffer[i]).data).username[j]
                    spes[idxtemp][j] <- ((Doctor*)(globalUserDatabase.
                        buffer[i]).data).spesialisasi[j]

                    baris <- (char)(row + 65)
                    output(idxtemp + 1, "Dr.", name[idxtemp], " - Spesialisasi",
                        Spes[idxtemp], " - Ruangan" baris, col + 1

                    if(globalDenahRumahSakit.Ruangan[row][col].idAntrian.size <
                        globalDenahRumahSakit.kapasitasRuangan)then
                        output(" (Ruangan belum penuh)")
                    else
                        output(" (Antrian: ",
globalDenahRumahSakit.Ruangan[row][col].
                            idAntrian.size - globalDenahRumahSakit.kapasitasRuangan,
                            "orang")

                            idxtemp++

                            if(idxtemp = 1)then
                                output("Pilih dokter (1): ")
                            else if(idxtemp > 1) then
                                output("Pilih dokter (1 - ", idxtemp, ")")

                            input(idx)
                            pasien <- createLLNode(((Patient*)globalCurrentUserGD->data)->id,
                                ((Patient*)globalCurrentUserGD->data)->username)

                            enqueue(&globalDenahRumahSakit.Ruangan[rowtemp[idx - 1]][coltemp
                                [idx - 1]].idAntrian, pasien)

                            if(globalDenahRumahSakit.Ruangan[rowtemp[idx - 1]][coltemp[idx - 1]].
                                idAntrian.size <= globalDenahRumahSakit.kapasitasRuangan)then
                                output("Pendaftaran check-up berhasil!")
                                output("Anda terdaftar pada antrian Dr. ", name[idx - 1], "di ruangan
",
                                    (char)(rowtemp[idx - 1] + 65), (coltemp[idx - 1] + 1))
                                output("Anda dapat langsung masuk ke dalam ruangan.")
                            else
                                output("Pendaftaran check-up berhasil!")
                                output("Anda terdaftar pada antrian Dr. ", name[idx - 1], "di ruangan
",
                                    (char)(rowtemp[idx - 1] + 65), (coltemp[idx - 1] + 1));
                                output("Posisi antrian anda: ", globalDenahRumahSakit.Ruangan[rowtemp
                                    [idx - 1]][coltemp[idx - 1]].idAntrian.size-globalDenahRumahSakit.
                                    kapasitasRuangan)

                                else
                                    output("Maaf, tidak ada dokter yang tersedia")

                                else
                                    output("Anda sudah terdaftar dalam antrian check-up!")
                                    output("Silahkan selesaikan check-up yang sudah terdaftar terlebih
                                        dahulu.")

                                    output("")

```

15. F15

```
procedure antrianSaya()
{I.S.: lokasiRuangan sudah terdefinisi pada fungsi di atas}
{F.S.: Menampilkan posisi antrian pasien}
KAMUS LOKAL
lokasiRuangan: Point
{variabel yang menunjukkan lokasi ruangan dan posisi antrian}
procedure getAccountName(input id: integer, dataType: DataType)
{untuk menampilkan nama dokter}
function posisiRuanganAntrianPasien(input userId: integer) -> Point
{untuk mencari posisi ruangan dan antrian pasien dan disimpan dalam
variabel lokasiRuangan}

ALGORITMA
lokasiRuangan <-
posisiRuanganAntrian((*Patient*)globalCurrentUserGD↑.data).id)
if(lokasiRuangan.antrian = -1)then
    output("Anda belum terdaftar dalam antrian check-up!")
    output("Silakan daftar terlebih dahulu dengan command
    DAFTAR_CHECKUP.")
else if(lokasiRuangan.antrian = -2)then
    output("Anda sedang berada di ruangan dokter!")
else{sedang berada di antrian}
    output("Status antrian: ")
    output("Dokter: ",
    getAccountName(globalDenahRumahSakit.Ruangan[lokasiRuangan.row][loka
    siRuangan.column].idDokter, DATA_TYPE_DOCTOR))
    output("Ruangan: ", lokasiRuangan.row+'A', lokasiRuangan.column+1)
    output("Posisi antrian: ", lokasiRuangan.antrian, " dari ",
    globalDenahRumahSakit.Ruangan[lokasiRuangan.row][lokasiRuangan.colum
    n].idAntrian.size-globalDenahRumahSakit.kapasitasRuangan))
```

16. F16

```
procedure minumObat()
{I.S.: sudahDiDiagnosis, sudahDiObatin, array inventory, stack perut
sudah terdefinisi}
{F.S.: jika sesuai syarat, obat yang dipilih akan dipindahkan ke perut
dan list obat yang baru sudah rata kiri}
KAMUS LOKAL
jlhObat, pilihan, i, j: integer
{jlhObat digunakan untuk validasi isi inventory dan obat yang akan
dipilih, pilihan digunakan untuk menyimpan nomor urut obat yang dipilih
dari daftar obat, i dan j digunakan sebagai variabel pembantu dalam loop}
o, iniObat: Obat
{o digunakan untuk mencetak daftar obat, iniObat digunakan untuk push
obat ke perut}
```

```

procedure pushStack(input s: Address, o: Obat)
{prosedur untuk memindahkan obat ke perut}
procedure goToLeft()
{prosedur untuk merata-kirikan array inventory setelah minum obat}

```

ALGORITMA

```

    if(globalCurrentPatient↑.sudahDiDiagnosis)then
        if(globalCurrentPatient↑.sudahDiObatin)then
            jlhObat <- 0
            i traversal [0..INVENTORY_SIZE-1]
                if(globalCurrentPatient↑.inventory[i] ≠ UNDEF_INT_DATA)then
                    j traversal [0..globalObatDatabase.nEff]
                        if(globalObatDatabase.contents[j].id =
                            globalCurrentPatient↑.inventory[i])then
                            o <- globalObatDatabase.contents[j]
                            break
                        j <- j+1
                    jlhObat <- jlhObat + 1
                i <- i + 1
            if(jlhObat = 0)then {inventory kosong}
                output("Inventorynya udah kosong. Minum penawar dengan
                    command PENAWAR jika perlu mengulang minum obat.")
                return
            else{inventory tidak kosong}
                output("===== DAFTAR OBAT =====")
                i traversal [0..INVENTORY_SIZE-1]
                    if(globalCurrentPatient↑.inventory[i] ≠
                        UNDEF_INT_DATA)then
                        j traversal [0..globalObatDatabase.nEff]
                            if(globalObatDatabase.contents[j].id =
                                globalCurrentPatient↑.inventory[i])then
                                    o <- globalObatDatabase.contents[j]
                                    break
                            output(i+1, ". ", o.name)
                        j <- j+1
                    jlhObat <- jlhObat + 1
                i <- i + 1
            pilihan <- UNDEF_INT_DATA
            while(pilihan < 1 or pilihan > jlhObat)do
                output("Pilih obat untuk diminum: ")
                input(pilihan)
                if(pilihan < 1 or pilihan > jlhObat)
                    output("Pilihan obat tidak tersedia!")
            i traversal [0..globalObatDatabase.nEff-1]
                if(globalObatDatabase.contents[i].id ==
                    globalCurrentPatient↑.inventory[pilihan - 1])then
                    iniObat <- globalObatDatabase.contents[i]
                    pushStack(globalCurrentPatient↑.perut, iniObat)

```

```

        globalCurrentPatient↑.inventory[pilihan - 1] <-
        UNDEF_INT_DATA
        goToLeft()
        output("GLEKGLEKGLEK... ", iniObat.name, " berhasil
        diminum!!!")
    else{belum diobatin}
        output("Anda belum meminta obat dari dokter. Minta obat
        dengan command NGOBATIN.")
    else{belum didiagnosis}
        output("Anda belum melakukan diagnosis. Lakukan diagnosis dengan
        command DIAGNOSIS.")

```

17. F17

```

procedure minumPenawar()
{I.S.: stack perut sudah terdefinisi, mungkin kosong, array inventory
sudah terdefinisi, mungkin kosong}

KAMUS LOKAL
i: integer
{i adalah variabel yang digunakan untuk membantu dalam loop}
backToInventory: Obat
{backToInventory adalah variabel yang digunakan untuk menandai obat yang
akan dikembalikan ke perut setelah minum penawar}
function popStack(input s: Address) -> Obat
{fungsi untuk menghapus obat dari suatu stack dan mengembalikan obat yang
dikembalikan tersebut}
function isEmptyStack(input s: Address) -> boolean
{fungsi untuk mengecek apakah suatu stack kosong(true jika kosong, false
jika tidak)}

ALGORITMA
    if(isEmptyStack(globalCurrentPatient↑.perut))then
        output("Perut kosong. Belum ada obat yang diminum.")
        return
    backToInventory <- popStack(globalCurrentPatient↑.perut)
    i traversal [0..INVENTORY_SIZE]
        if(globalCurrentPatient↑.inventory[i] = UNDEF_INT_DATA)then
            globalCurrentPatient↑.inventory[i] <- backToInventory.id
            break
    output("Uwekkk!!! ", backToInventory.name, " keluar dan kembali ke
    inventory.")

```

18. F18

```

procedure exitFromHospital()
{ I.S. Dalam keadaan tidak error pada program }
{ F.S. Menulis ke file-file jika pengguna meminta untuk save. Seluruh
memori dinamis di-dealokasi. Keluar dari program }

```


KAMUS LOKAL

input : character

ALGORITMA

```
{ Tanya jika ingin save, input diulang hingga valid }
input ← 'a'
while (input ≠ 'y' and input ≠ 'n') do
    output("Apakah Anda mau melakukan penyimpanan file yang sudah
diubah? (y/n): ")
    input(input)

{ Jika memilih save }
if (input = 'y' or input = 'Y') then
    saveCSV() { writeConfig termasuk dalam saveCSV }

output("Sampai jumpa, Niemons!")

{ Dealokasi memori }
deallocateLD(globalUserDatabase)
freeSet(globalUsernames)

{ Dealokasi matrix ruangan }
i traversal [0..DENAH_RUANGAN_MATRIX_CAPACITY - 1]
    j traversal [0..DENAH_RUANGAN_MATRIX_CAPACITY - 1]
        freeQueue(globalDenahRumahSakit.Ruangan[i][j].idAntrian)

{ Keluar dari program }
exit(0)
```

19. D03

```
{ Modul Pembaca CSV }
{ Berisi primitif untuk memuat database dari file CSV }
{ Menggunakan list dinamis dan struktur data SEQFILE }
```

KAMUS

```
const filenameList : array[0..3] of string = ["user.csv", "obat.csv",
"penyakit.csv", "obat_penyakit.csv"]
const MARK : string = "NULL"
```

{ TIPE DATA }

type CSVRow : <

fields : array[0..MAX_FIELDS-1] of string,
fieldCount : integer

>

{ VARIABEL GLOBAL }

globalUserDatabase : ListDin

globalUsernames : Set

isAllReadSuccessfully : boolean

{ *** PRIMITIF UTAMA *** }

procedure loadCSV(input argc : integer, input argv : array of string)

{ I.S.: Program dijalankan dengan parameter command line }

{ F.S.: Database dimuat dari folder yang ditentukan atau keluar }

```

dengan error }
    KAMUS LOKAL
        folderPath : string
    ALGORITMA
        createLD(globalUserDatabase, 20)
        createSet(globalUsernames, 20)

        if (argc < 2) then
            output("ERROR: TIDAK ADA NAMA FOLDER YANG DIBERIKAN!")
            exit(1)
        else
            folderPath ← argv[1]
            if not isFileInPath(folderPath) then
                output("ERROR: FILE ATAU FOLDER TIDAK DITEMUKAN")
                exit(1)
            else
                processAllCSVInFolder(folderPath)

procedure processAllCSVInFolder(input folderPath : string)
{ I.S.: Folder path valid }
{ F.S.: Semua file CSV dalam filenameList diproses }
KAMUS LOKAL
    i : integer
ALGORITMA
    output("Loading...")
    isAllReadSuccessfully ← true

    i ← 0
    while (filenameList[i] ≠ MARK) do
        processCSV(folderPath, filenameList[i])
        i ← i + 1

    if isAllReadSuccessfully then
        output("Finished Loading!")
        output("Selamat datang di rumah sakit Niemons!")
    else
        output("ERROR: TERDAPAT DATA YANG TIDAK DIBACA")
        exit(1)

{ *** PRIMITIF PEMROSESAN FILE *** }
procedure processCSV(input folder : string, input filename : string)
{ I.S.: Folder dan filename valid }
{ F.S.: Data dari file CSV dimuat ke database yang sesuai }
KAMUS LOKAL
    fullPath : string
    file : SEQFILE of
        (*) row : CSVRow
        (1) MARK
    row : CSVRow
    gd : ↑GenericData
ALGORITMA
    { Bangun path lengkap }
    fullPath ← folder + "/" + filename

    { Buka file }
    assign(file, fullPath)
    open(file, line)

```

```

    if (file = NULL) then
        output("ERROR DALAM MEMBUKA FILE " + fullPath)
        isAllReadSuccessfully ← false
        return

    { Lewati header }
    read(file, line)

    { Proses berdasarkan jenis file }
    if (filename = "user.csv") then
        processUserCSV(file)
    else if (filename = "obat.csv") then
        processObatCSV(file)
    else if (filename = "penyakit.csv") then
        processPenyakitCSV(file)
    else if (filename = "obat_penyakit.csv") then
        processObatPenyakitCSV(file)
    else
        output("ERROR: FILE " + filename + " NOT FOUND")
        isAllReadSuccessfully ← false

    close(file)

{ *** SUBPROSES KHUSUS *** }
procedure processUserCSV(input/output file : SEQFILE)
{ I.S.: File user.csv terbuka }
{ F.S.: Data user dimuat ke globalUserDatabase }
KAMUS LOKAL
    line : string
    row : CSVRow
    id : integer
    role : string
    gd : ↑GenericData
ALGORITMA
    while read(file, line) and (line ≠ MARK) do
        if isLDFull(globalUserDatabase) then
            expandLD(globalUserDatabase, 10)

        row ← parseCSVLine(line)

        if (row.fieldCount < NUM_OF_COL_GENERIC_USER) then
            continue

        id ← atoi(row.fields[0])
        role ← row.fields[3]

        { Tambahkan username ke set }
        if isSetFull(globalUsernames) then
            expandSet(globalUsernames, 10)
            addToSet(globalUsernames, row.fields[1])

        { Proses berdasarkan role }
        if (role = "pasien") and (row.fieldCount ≥
NUM_OF_COL_PATIENT) then
            gd ← createPatientFromRow(row)
        else if (role = "dokter") then
            gd ← createDoctorFromRow(row)
        else if (role = "manager") then

```

```

        gd ← createManagerFromRow(row)
    else
        output("ERROR: ROLE TIDAK DIKENALI: " + role + ", ID: " +
id)

        isAllReadSuccessfully ← false
        continue

    { Tambahkan ke database }
    if (gd = NULL) then
        output("ERROR ON PROCESSING DATA WITH ID " + id)
        isAllReadSuccessfully ← false
    else
        insertLastLD(globalUserDatabase, gd)

    compressLD(globalUserDatabase)
    compressSet(globalUsernames)
    output("LOADED USER DATABASE!")

    { Prosedur lainnya (processObatCSV, processPenyakitCSV,
processObatPenyakitCSV) }
    { mengikuti pola yang sama dengan penyesuaian struktur data
masing-masing }

    { *** SUBPROSES KHUSUS *** }
    procedure processUserCSV(input/output file : SEQFILE)
    { I.S.: File user.csv terbuka }
    { F.S.: Data user dimuat ke globalUserDatabase }
    KAMUS LOKAL
    line : string
    row : CSVRow
    id : integer
    role : string
    gd : ↑GenericData
    ALGORITMA
    while read(file, line) and (line ≠ MARK) do
        if isLDFull(globalUserDatabase) then
            expandLD(globalUserDatabase, 10)

        row ← parseCSVLine(line)

        if (row.fieldCount < NUM_OF_COL_GENERIC_USER) then
            continue

        id ← atoi(row.fields[0])
        role ← row.fields[3]

        { Tambahkan username ke set }
        if isSetFull(globalUsernames) then
            expandSet(globalUsernames, 10)
            addToSet(globalUsernames, row.fields[1])

        { Proses berdasarkan role }
        if (role = "pasien") and (row.fieldCount ≥
NUM_OF_COL_PATIENT) then
            gd ← createPatientFromRow(row)
        else if (role = "dokter") then
            gd ← createDoctorFromRow(row)

```

```

        else if (role = "manager") then
            gd ← createManagerFromRow(row)
        else
            output("ERROR: ROLE TIDAK DIKENALI: " + role + ", ID: " +
id)

            isAllReadSuccessfully ← false
            continue

        { Tambahkan ke database }
        if (gd = NULL) then
            output("ERROR ON PROCESSING DATA WITH ID " + id)
            isAllReadSuccessfully ← false
        else
            insertLastLD(globalUserDatabase, gd)

        compressLD(globalUserDatabase)
        compressSet(globalUsernames)
        output("LOADED USER DATABASE!")

        { Prosedur lainnya (processObatCSV, processPenyakitCSV,
processObatPenyakitCSV) }
        { mengikuti pola yang sama dengan penyesuaian struktur data
masing-masing }

```

```

procedure readConfig(input path: string)
{ Membaca file konfigurasi "config.txt" dari direktori yang
ditetapkan oleh 'path'.
  File ini berisi data untuk menginisialisasi denah rumah sakit,
termasuk ukuran denah,
  kapasitas ruangan dan antrian, penempatan dokter di setiap ruangan,
antrian pasien awal
  untuk setiap dokter, serta status obat dan isi perut untuk beberapa
pasien. }

```

KAMUS LOKAL

```

    fullPath: string
    configFile: FilePointer
    nRow, nColumn, maxPasien, maxAntrian, nPasienObat, nPasienPerut:
integer
    row, column, i, id_pasien_idx, j: integer { Variabel iterasi }
    idDokterRuangan: integer { Menggunakan nama berbeda dari C untuk
kejelasan scope }
    tempIdPasienConfig: integer { Menggunakan nama berbeda dari C untuk
kejelasan scope }
    countPasienDiRuangan: integer { Menggunakan nama berbeda dari C
untuk kejelasan scope }
    pasienDenganObat, pasienDenganIsiPerut: PointerToPatient { Asumsi
tipe pointer ke data Patient }
    tempIdObatConfig: integer { Menggunakan nama berbeda dari C untuk
kejelasan scope }
    obatDataPtr: PointerToObat { Asumsi tipe pointer ke data Obat }
    tempStackObat: StackObat { Asumsi StackObat adalah ADT Stack dengan
elemen bertipe Obat }
    obatDariStack: Obat
    sizeCurrentStack: integer
    nodePasienBaru: Address { Alamat untuk node linked list pasien
dalam antrian }
    lanjutkanBacaPasienRuangan: boolean { Flag untuk kontrol loop }

```

ALGORITMA

```
fullPath <- path + "/" + "config.txt" { Operasi penyambungan string
}
assign(configFile, fullPath)
openForRead(configFile)

nRow <- scanNumber(configFile)
nColumn <- scanNumber(configFile)
globalDenahRumahSakit.nRow <- nRow
globalDenahRumahSakit.nColumn <- nColumn
skipNextChar(configFile) { Melewati karakter newline }

maxPasien <- scanNumber(configFile)
maxAntrian <- scanNumber(configFile)
globalDenahRumahSakit.kapasitasRuangan <- maxPasien
globalDenahRumahSakit.kapasitasAntrian <- maxAntrian
skipNextChar(configFile)

row traversal [0..(nRow - 1)]
column traversal [0..(nColumn - 1)]

createQueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian,
maxAntrian + maxPasien)
    idDokterRuangan <- scanNumber(configFile)
    globalDenahRumahSakit.Ruangan[row][column].idDokter <-
idDokterRuangan

    if (idDokterRuangan = 0) then
        globalDenahRumahSakit.Ruangan[row][column].idAntrian.size <-
0
        skipNextChar(configFile)
        { Logika 'continue' di C berarti sisa iterasi untuk pasien di
ruangan ini dilewati }
    else
        { Dokter ada, proses pembacaan pasien untuk ruangan ini }
        countPasienDiRuangan <- 0
        lanjutkanBacaPasienRuangan <- true
        id_pasien_idx traversal [0..(maxPasien + maxAntrian - 1)]
        if (lanjutkanBacaPasienRuangan) then
            tempIdPasienConfig <- scanNumber(configFile)
            if (tempIdPasienConfig = 0) then

globalDenahRumahSakit.Ruangan[row][column].idAntrian.size <- 0
                skipNextChar(configFile)
                lanjutkanBacaPasienRuangan <- false { Efek 'break' dari
loop pasien }
            else if (tempIdPasienConfig = -1) then

globalDenahRumahSakit.Ruangan[row][column].idAntrian.size <-
countPasienDiRuangan
                lanjutkanBacaPasienRuangan <- false { Efek 'break' dari
loop pasien }
            else
                nodePasienBaru <- createLLNode(tempIdPasienConfig, "")

enqueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian,
nodePasienBaru)
                countPasienDiRuangan <- countPasienDiRuangan + 1
                if (id_pasien_idx = (maxPasien + maxAntrian - 1)) then
```

```

globalDenahRumahSakit.Ruangan[row][column].idAntrian.size <-
countPasienDiRuang
    skipNextChar(configFile)
    { akhir dari if id_pasien_idx terakhir }
    { akhir dari if tempIdPasienConfig }
    { akhir dari if lanjutkanBacaPasienRuang }
    { akhir dari traversal id_pasien_idx }
    { akhir dari if idDokterRuang = 0 }
    { akhir dari traversal column }
    { akhir dari traversal row }

nPasienObat <- scanNumber(configFile)
skipNextChar(configFile)

i traversal [0..(nPasienObat - 1)]
tempIdPasienConfig <- scanNumber(configFile)
pasienDenganObat <- getAccountAddress(tempIdPasienConfig)
pasienDenganObat.sudahDiObatin <- true
pasienDenganObat.sudahDiDiagnosis <- true
tempIdObatConfig <- scanNumber(configFile)
while (tempIdObatConfig != -1) do
    j <- 0
    while (pasienDenganObat.inventory[j] != UNDEF_INT_DATA) and (j
< INVENTORY_SIZE) do
        j <- j + 1
        { akhir dari while cari slot }
        if (j = INVENTORY_SIZE) then
            output("INVENTORY ", pasienDenganObat.username, " FULL")
        else
            pasienDenganObat.inventory[j] <- tempIdObatConfig
            { akhir dari if inventory full }
            tempIdObatConfig <- scanNumber(configFile)
            { akhir dari while tempIdObatConfig != -1 }
        { akhir dari traversal i untuk nPasienObat }

nPasienPerut <- scanNumber(configFile)
skipNextChar(configFile)

i traversal [0..(nPasienPerut - 1)]
tempIdPasienConfig <- scanNumber(configFile)
pasienDenganIsiPerut <- getAccountAddress(tempIdPasienConfig)
pasienDenganIsiPerut.sudahDiObatin <- true
pasienDenganIsiPerut.sudahDiDiagnosis <- true
tempIdObatConfig <- scanNumber(configFile)

CreateStack(tempStackObat)
while (tempIdObatConfig != -1) do
    obatDataPtr <- getObatById(tempIdObatConfig)
    push(tempStackObat, obatDataPtr.data) { Asumsi .data adalah
payload Obat }
    tempIdObatConfig <- scanNumber(configFile)
    { akhir dari while }

sizeCurrentStack <- length(tempStackObat)
while (sizeCurrentStack > 0) do
    pop(tempStackObat, obatDariStack)
    push(pasienDenganIsiPerut.perut, obatDariStack)
    sizeCurrentStack <- sizeCurrentStack - 1
    { akhir dari while }
    { akhir dari traversal i untuk nPasienPerut }

```

```
close(configFile)
```

20. D04

```
procedure saveFiles()  
{ I.S. Seluruh database terdefinisi, terdapat data. Folder dan file-file  
yang ingin ditulis mungkin tidak ada }  
{ F.S. Jika folder dan file-file tidak ada, maka dibuat. Jika ada, maka  
di overwrite. Setiap file diisi dengan data masing-masing dari database }
```

KAMUS LOKAL

```
path : string  
valid : boolean
```

ALGORITMA

```
{ Meminta input folder hingga valid }  
repeat  
    output("Masukkan nama folder (contoh: data/hari_ini): ")  
    input(path)  
  
    if (contains(path, "\")) then  
        output("Gunakan forward slashes (/), bukan backslashes (\)")  
        valid ← false  
    else if (contains(path, "..") or (contains(path, "~"))) then  
        output("Path tidak boleh mengandung '..' atau '~")  
        valid ← false  
    else  
        valid ← true  
until (valid)  
  
{ Membuat folder jika belum ada }  
if (not doesFolderExist(path)) then  
    output("Membuat folder ", path, "...")  
    createDir(path)  
  
{ Menulis ke semua file CSV }  
writeConfig(path)  
i ← 0  
while (filenameList[i] ≠ NULL) do  
    writeToCSV(path, filenameList[i])  
    i ← i + 1  
  
output("Data berhasil disimpan di folder ", path, "!")
```

```
procedure writeToCSV(input folder, filename : string)  
{ I.S. File CSV mungkin ada atau tidak ada }  
{ F.S. File CSV berisi data terbaru dari database sesuai filename }
```

KAMUS LOKAL

```
fullPath : string  
file : File  
i, j : integer  
numOfUsersDeleted : integer  
gd : GenericData  
gdDataType : DataType  
p : Patient
```



```
d : Doctor
m : Manager
```

ALGORITMA

```
{ Buka file }
fullPath ← folder + "/" + filename
file ← open(fullPath)

if (file = NULL) then
    output("ERROR: GAGAL MEMBUKA FILE ", fullPath)
    → { keluar prosedur }

{ Proses berdasarkan filename }
depend on (filename)
    "user.csv" :
        { Tulis header }
        write(file,
            "id;username;password;role;riwayat_penyakit;suhu_tubuh;tekanan_darah_sistolik;tek
            anan_darah_diastolik;detak_jantung;saturasi_oksigen;kadar_gula_darah;berat_badan;
            tinggi_badan;kadar_kolesterol;trombosit\n")

        numOfUsersDeleted ← 0

        { Proses setiap user }
        i ← 0
        while (i < globalUserDatabase.nEff) do
            gd ← getGDbyIdx(globalUserDatabase, i)
            gdDataType ← getDataTypeGD(gd)

            depend on (gdDataType)
                DATA_TYPE_PATIENT :
                    p ← getPatientInGD(gd)
                    write(file, p.id + ";" + p.username + ";" + p.password +
                        ";pasien;" + p.riwayatPenyakit)

                    j ← 0
                    while (j < KONDISI_TUBUH_SIZE) do
                        if (p.kondisiTubuh[j] - UNDEF_INT_DATA < 0.001) then
                            write(file, ";")
                        else if (p.kondisiTubuh[j] = floor(p.kondisiTubuh[j])) then
                            write(file, ";" + floor(p.kondisiTubuh[j]))
                        else
                            write(file, ";" + p.kondisiTubuh[j]:1) { format 1
decimal }
                        j ← j + 1
                    write(file, "\n")

                DATA_TYPE_DOCTOR :
                    d ← getDoctorInGD(gd)
                    write(file, d.id + ";" + d.username + ";" + d.password +
                        ";dokter;;;;;;;;;;\n")

                DATA_TYPE_MANAGER :
                    m ← getManagerInGD(gd)
                    write(file, m.id + ";" + m.username + ";" + m.password +
                        ";manager;;;;;;;;;;\n")

                DATA_TYPE_UNKNOWN :
                    numOfUsersDeleted ← numOfUsersDeleted + 1

            i ← i + 1

        if (numOfUsersDeleted > 0) then
            output("Jumlah pengguna yang dihapus dari hospital Niemons: ",
numOfUsersDeleted)
            output("SAVED USER DATABASE!")

        "obat.csv" :
            write(file, "obat_id;nama_obat\n")
            i ← 0
            while (i < globalObatDatabase.nEff) do
                write(file, globalObatDatabase.contents[i].id + ";" +
globalObatDatabase.contents[i].name + "\n")
                i ← i + 1
            output("SAVED OBAT DATABASE!")
```

```

        "penyakit.csv" :
        write(file,
        "id;nama_penyakit;suhu_tubuh_min;suhu_tubuh_max;...;trombosit_max\n")
        i ← 0
        while (i < globalPenyakitDatabase.nEff) do
            write(file, globalPenyakitDatabase.contents[i].id + ";" +
            globalPenyakitDatabase.contents[i].name)
            j ← 0
            while (j < THRESHOLD_SIZE) do
                if (globalPenyakitDatabase.contents[i].threshold[j] =
                floor(globalPenyakitDatabase.contents[i].threshold[j])) then
                    write(file, ";" +
                    floor(globalPenyakitDatabase.contents[i].threshold[j]))
                else
                    write(file, ";" +
                    globalPenyakitDatabase.contents[i].threshold[j]:1) { format 1 decimal }
                j ← j + 1
            write(file, "\n")
            i ← i + 1
        output("SAVED PENYAKIT DATABASE!")

        "obat_penyakit.csv" :
        write(file, "obat_id;penyakit_id;urutan_minum\n")
        i ← 0
        while (i < globalOPDatabase.nEff) do
            j ← 0
            while (j < globalOPDatabase.contents[i].nEff) do
                write(file, globalOPDatabase.contents[i].idObat[j] + ";" +
                globalOPDatabase.contents[i].idPenyakit + ";" + (j+1) + "\n")
                j ← j + 1
            i ← i + 1
        output("SAVED OBAT PENYAKIT DATABASE!")

    close(file)

```

procedure writeConfig(input path: string)
 { Menulis konfigurasi sistem rumah sakit saat ini ke file "config.txt" di dalam direktori 'path'.
 Ini termasuk dimensi denah, kapasitas ruangan/antrian, dokter di setiap ruangan beserta antrian pasiennya,
 data pasien yang memiliki obat di inventory, dan data pasien yang memiliki obat di perut (stack). }

KAMUS LOKAL

```

    fullPath: string
    configFile: FilePointer { Tipe abstrak untuk file }
    row, column, i, j: integer { Variabel iterasi }
    ruanganSaatIni: DataTypeRuangan { Asumsi tipe data untuk elemen matriks Ruangan }
}

    queueSizeSaatIni: integer

    idPasienObat: array [1..MAX_USERS] of integer { Asumsi MAX_USERS adalah kapasitas globalUserDatabase.nEff }
    nPasienObatInvent: integer { Menggunakan nama yang lebih deskriptif }
    pasienDenganInvent: PointerToPatient { Asumsi tipe pointer ke data Patient }

    idPasienPerutObat: array [1..MAX_USERS] of integer { Asumsi MAX_USERS }
    nPasienDenganPerut: integer { Menggunakan nama yang lebih deskriptif }
    pasienDenganPerutData: PointerToPatient
    ukuranPerutStack: integer
    obatDariPerut: Obat { Asumsi Obat adalah tipe data elemen di stack perut }
    pasienDiAntrianNode: Address { Alamat node pasien dalam antrian }

```

ALGORITMA

```

    fullPath <- path + "/" + "config.txt" { Operasi penyambungan string }
    assign(configFile, fullPath)
    rewrite(configFile) { Membuka file untuk ditulis, membuat baru atau menimpa }

    { Menulis dimensi denah dan kapasitas }
    write(configFile, globalDenahRumahSakit.nRow, " ",
    globalDenahRumahSakit.nColumn)
    writeNewline(configFile)
    write(configFile, globalDenahRumahSakit.kapasitasRuangan, " ",
    globalDenahRumahSakit.kapasitasAntrian)

```

```

writeNewline(configFile)

{ Menulis data setiap ruangan (dokter dan antrian pasien) }
row traversal [0..(globalDenahRumahSakit.nRow - 1)]
  column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
    ruanganSaatIni <- globalDenahRumahSakit.Ruangan[row][column]
    write(configFile, ruanganSaatIni.idDokter)

    if (ruanganSaatIni.idDokter = 0) then
      writeNewline(configFile)
    else
      queueSizeSaatIni <- ruanganSaatIni.idAntrian.size
      if (isEmpty(queueSizeSaatIni.idAntrian)) then
        write(configFile, " ", 0)
      else
        { Penting: Operasi dequeue di C bersifat destruktif. Untuk menulis
        konfigurasi, seharusnya antrian tidak diubah. Jika ini adalah antrian sementara
        untuk ditulis, maka tidak apa-apa.
        Jika ini adalah antrian utama, maka ini akan mengosongkan antrian.
        Saya akan menerjemahkan logika C yang destruktif ini. }
        pasienDiAntrianNode <- ruanganSaatIni.idAntrian.front { Akses awal ke
        front }
        i traversal [0..(queueSizeSaatIni - 1)]
          write(configFile, " ", pasienDiAntrianNode.id)
          dequeue(ruanganSaatIni.idAntrian, pasienDiAntrianNode) { dequeue
          memodifikasi antrian & mungkin mengembalikan node/value }
          pasienDiAntrianNode <- ruanganSaatIni.idAntrian.front { Update ke
          front baru setelah dequeue }
          { akhir dari traversal i }
          { akhir dari if isEmpty }
          writeNewline(configFile)
          { akhir dari if ruanganSaatIni.idDokter = 0 }
          { akhir dari traversal column }
          { akhir dari traversal row }

        { Menulis data pasien yang memiliki obat di inventory }
        nPasienObatInvent <- countBanyakPasienInventory(idPasienObat) { Fungsi ini
        mengisi idPasienObat }
        write(configFile, nPasienObatInvent)
        writeNewline(configFile)

        i traversal [0..(nPasienObatInvent - 1)]
          pasienDenganInvent <- getAccountAddress(idPasienObat[i+1]) { Asumsi
          idPasienObat 1-indexed atau perlu penyesuaian }
          write(configFile, pasienDenganInvent.id)
          j <- 0
          while (pasienDenganInvent.inventory[j+1] != UNDEF_INT_DATA) and (j <
          INVENTORY_SIZE) do { Asumsi inventory 1-indexed atau perlu penyesuaian }
            write(configFile, " ", pasienDenganInvent.inventory[j+1])
            j <- j + 1
          { akhir dari while }
          writeNewline(configFile)
          { akhir dari traversal i }

        { Menulis data pasien yang memiliki obat di perut (stack) }
        nPasienDenganPerut <- countBanyakPasienPerut(idPasienPerutObat) { Fungsi ini
        mengisi idPasienPerutObat }
        write(configFile, nPasienDenganPerut)
        writeNewline(configFile)

        i traversal [0..(nPasienDenganPerut - 1)]
          pasienDenganPerutData <- getAccountAddress(idPasienPerutObat[i+1]) { Asumsi
          idPasienPerutObat 1-indexed }
          write(configFile, pasienDenganPerutData.id)
          ukuranPerutStack <- length(pasienDenganPerutData.perut) { Menggunakan length
          dari ADT Stack }

          { Untuk menulis isi stack tanpa mengubahnya secara permanen, idealnya stack
          disalin dulu.
          Logika C (fprintf top lalu pop) bersifat destruktif. Saya terjemahkan
          secara destruktif. }
          while (ukuranPerutStack > 0) do
            obatDariPerut <- top(pasienDenganPerutData.perut) { Mengintip elemen
            teratas }

```

```

        write(configFile, " ", obatDariPerut.id)
        pop(pasienDenganPerutData.perut, obatDariPerut) { Menghapus elemen teratas
    }

    ukuranPerutStack <- ukuranPerutStack - 1
    { akhir dari while }
    writeNewline(configFile)
    { akhir dari traversal i }

    close(configFile)

```

21. D04

22. B02

procedure ubahDenah()
 {I.S. Denah rumah sakit dengan banyak row dan column tertentu}
 {F.S. Mengubah ukuran denah rumah sakit (jumlah baris dan kolom).
 Memeriksa apakah perubahan aman (tidak ada dokter di ruangan yang akan dihapus).
 Jika aman, antrian di ruangan yang dihapus dibebaskan, antrian untuk ruangan baru
 dibuat, dan ukuran denah diperbarui.}

KAMUS LOKAL

newRow, newColumn: integer
 tempRow: integer
 row, column: integer
 bisaUbahDenah: Point <row:integer, column:integer, antrian: integer>
 tempPoint: Point
 lanjutkanPencarianBaris, lanjutkanPencarianKolom: boolean { Flag untuk kontrol
 loop }

ALGORITMA

```

    bisaUbahDenah.row <--1
    bisaUbahDenah.column <--1
    bisaUbahDenah.antrian <-- 1 { Inisialisasi: 1 berarti true (bisa diubah) }
    input(newRow, newColumn)

    if (newRow > 26) or (newColumn > 26) then
        output("Banyak baris atau kolom tidak boleh lebih dari 26")
    else
        {Cek apakah ada dokter di area yang akan terpotong jika denah mengecil di
        kolom}
        if (newColumn < globalDenahRumahSakit.nColumn) then
            if (newRow < globalDenahRumahSakit.nRow) then
                tempRow <- newRow
            else
                tempRow <- globalDenahRumahSakit.nRow

        lanjutkanPencarianBaris <- true
        row traversal [0..(tempRow - 1)]
        if (lanjutkanPencarianBaris) then
            lanjutkanPencarianKolom <- true
            column traversal [newColumn..(globalDenahRumahSakit.nColumn - 1)]
            if (lanjutkanPencarianKolom) then
                if (globalDenahRumahSakit.Ruangan[row][column].idDokter != 0) then
                    tempPoint.row <- row
                    tempPoint.column <- column
                    tempPoint.antrian <- 0
                    bisaUbahDenah <- tempPoint
                    lanjutkanPencarianKolom <- false
                    lanjutkanPencarianBaris <- false

        {Cek apakah ada dokter di area yang akan terpotong jika denah mengecil di
        baris}
        if (newRow < globalDenahRumahSakit.nRow) and (bisaUbahDenah.antrian = 1) then
            lanjutkanPencarianBaris <- true

```

```

row traversal [newRow..(globalDenahRumahSakit.nRow - 1)]
  if (lanjutkanPencarianBaris) then
    lanjutkanPencarianKolom <- true
    column traversal [0..(globalDenahRumahSakit.nColumn - 1)] {Kolom dicek
    semua untuk baris yang dipotong}
    if (lanjutkanPencarianKolom) then
      if (globalDenahRumahSakit.Ruangan[row][column].idDokter != 0) then
        tempPoint.row <- row
        tempPoint.column <- column
        tempPoint.antrian <- 0
        bisaUbahDenah <- tempPoint
        lanjutkanPencarianKolom <- false
        lanjutkanPencarianBaris <- false

if (not (bisaUbahDenah.antrian = 1)) then
  output("Tidak dapat mengubah ukuran denah. Ruang ",
  karakterDariInteger(bisaUbahDenah.row + 65), bisaUbahDenah.column + 1, "
  masih ditempati oleh Dr. ",
  getAccountName(globalDenahRumahSakit.Ruangan[bisaUbahDenah.row][bisaUbahDen
  ah.column].idDokter, DATA_TYPE_DOCTOR), ". Silakan pindahkan dokter
  terlebih dahulu.")
else
  { Perubahan ukuran aman, lanjutkan dengan dealokasi/alokasi antrian }
  if (newColumn < globalDenahRumahSakit.nColumn) then
    if (newRow < globalDenahRumahSakit.nRow) then
      tempRow <- newRow
    else
      tempRow <- globalDenahRumahSakit.nRow
    row traversal [0..(tempRow - 1)]
    column traversal [newColumn..(globalDenahRumahSakit.nColumn - 1)]
    freeQueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian)

    if (newRow < globalDenahRumahSakit.nRow) then
      row traversal [newRow..(globalDenahRumahSakit.nRow - 1)]
      column traversal [0..(globalDenahRumahSakit.nColumn - 1)]
      freeQueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian)

    if (newColumn > globalDenahRumahSakit.nColumn) then
      if (newRow < globalDenahRumahSakit.nRow) then
        tempRow <- newRow
      else
        tempRow <- globalDenahRumahSakit.nRow
      row traversal [0..(tempRow - 1)]
      column traversal [globalDenahRumahSakit.nColumn..(newColumn - 1)]
      createQueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian,
      globalDenahRumahSakit.kapasitasAntrian +
      globalDenahRumahSakit.kapasitasRuang)

      if (newRow > globalDenahRumahSakit.nRow) then
        row traversal [globalDenahRumahSakit.nRow..(newRow - 1)]
        column traversal [0..(newColumn - 1)]
        createQueue(globalDenahRumahSakit.Ruangan[row][column].idAntrian,
        globalDenahRumahSakit.kapasitasAntrian +
        globalDenahRumahSakit.kapasitasRuang)

    globalDenahRumahSakit.nRow <- newRow
    globalDenahRumahSakit.nColumn <- newColumn
    output("Denah rumah sakit berhasil diubah menjadi ", newRow, " baris dan ",
    newColumn, " kolom.")

```

```

procedure pindahDokter()
{I.S. Posisi ruangan dokter yang ada di globalDenahRumahSakit}
{F.S. Memindahkan seorang dokter dari satu ruangan (ruanganLama) ke ruangan lain
(ruanganBaru).
  Operasi ini melibatkan pertukaran data ruangan, termasuk idDokter dan
  antriannya. }

```

KAMUS LOKAL

```

  ruanganLama, ruanganBaru: string
  rowLama, columnLama, rowBaru, columnBaru: integer
  tempRuanganData: DataTypeRuangan <idDokter: integer, kapasitasRuang: integer,
  kapasitasAntrian: integer, idAntrian: Queue>

```

ALGORITMA

```

  input(ruanganLama, ruanganBaru)

```

```

kodeRuanganKonverter(ruanganLama, rowLama, columnLama)
kodeRuanganKonverter(ruanganBaru, rowBaru, columnBaru)

depend_on (globalDenahRumahSakit.Ruangan[rowLama][columnLama].idDokter,
globalDenahRumahSakit.Ruangan[rowBaru][columnBaru].idDokter)
(globalDenahRumahSakit.Ruangan[rowLama][columnLama].idDokter = 0) :
    output("Pemindahan gagal. Ruangan ", karakterDariInteger(rowLama + 65),
    columnLama + 1, " Kosong.")

(globalDenahRumahSakit.Ruangan[rowLama][columnLama].idDokter != 0) and
(globalDenahRumahSakit.Ruangan[rowBaru][columnBaru].idDokter != 0) :
    output("Pemindahan gagal. Ruangan ", karakterDariInteger(rowBaru + 65),
    columnBaru + 1, " Sudah ditempati.")

(globalDenahRumahSakit.Ruangan[rowLama][columnLama].idDokter != 0) and
(globalDenahRumahSakit.Ruangan[rowBaru][columnBaru].idDokter = 0) :
    { Lakukan pemindahan }
    tempRuanganData <- globalDenahRumahSakit.Ruangan[rowBaru][columnBaru]
    globalDenahRumahSakit.Ruangan[rowBaru][columnBaru] <-
    globalDenahRumahSakit.Ruangan[rowLama][columnLama]
    globalDenahRumahSakit.Ruangan[rowLama][columnLama] <- tempRuanganData

    output("Dr. ",
    getAccountName(globalDenahRumahSakit.Ruangan[rowBaru][columnBaru].idDokter,
    DATA_TYPE_DOCTOR), " berhasil dipindahkan dari ruangan ",
    karakterDariInteger(rowLama + 65), columnLama + 1, " ke ruangan ",
    karakterDariInteger(rowBaru + 65), columnBaru + 1, ".")

```

23. B06

```

procedure skipAntrian()
{ Memungkinkan pasien saat ini untuk 'melompati' antrian ke posisi pertama dari
bagian antrian tunggu
  (tepat setelah pasien yang berada di dalam kapasitas ruangan), jika
memungkinkan.
  Memberikan pesan status berhasil atau gagal. }

KAMUS LOKAL
  lokasiRuangan: Point { Menyimpan informasi baris, kolom, dan status/posisi
antrian pasien.
                                .antrian = -1 jika tidak di antrian, -2 jika di dalam
ruangan,
                                angka positif jika di antrian tunggu (posisi ke-). }
  sizeQueue: integer
  pasienYangSkip, pasienSebelumYangSkip, pasienTerakhirDalamRuangan, tempNode:
Address { Address ke LinkedListNode }
  i: integer { Variabel iterasi }
  currentPatientId: integer { ID pasien saat ini }
  roomRow, roomCol: integer { Untuk menyimpan baris dan kolom ruangan dari
lokasiRuangan }
  targetQueue: QueueType { Asumsi QueueType adalah tipe dari idAntrian }

ALGORITMA
  currentPatientId <- globalCurrentPatient.id
  lokasiRuangan <- posisiRuanganAntrianPasien(currentPatientId)

  if (lokasiRuangan.antrian = -1) then
    output("")
    output("Skip antrian gagal! Anda tidak sedang terdaftar dalam antrian
manapun!")
    output("")
  else if (lokasiRuangan.antrian = -2) then
    output("")
    output("Anda sudah berada di dalam ruangan ",
    karakterDariInteger(lokasiRuangan.row + 65), lokasiRuangan.column + 1, " bersama
Dr. ",
    getAccountName(globalDenahRumahSakit.Ruangan[lokasiRuangan.row][lokasiRuangan.col
umn].idDokter, DATA_TYPE_DOCTOR), "! Tidak bisa skip antrian lagi.")
    output("")

```

```

else if (lokasiRuangan.antrian == 1) then { Asumsi posisi antrian 1 adalah yang
paling depan di bagian tunggu }
  output("")
  output("Anda sudah berada di posisi paling depan antrian Dr. ",
getAccountName(globalDenahRumahSakit.Ruangan[lokasiRuangan.row][lokasiRuangan.col
umn].idDokter, DATA_TYPE_DOCTOR), " di ruangan ",
karakterDariInteger(lokasiRuangan.row + 65), lokasiRuangan.column + 1, "! Tidak
bisa skip lagi.")
  output("")
else
  { Pasien berada di antrian tunggu dan bukan di posisi pertama }
  roomRow <- lokasiRuangan.row
  roomCol <- lokasiRuangan.column
  targetQueue <- globalDenahRumahSakit.Ruangan[roomRow][roomCol].idAntrian

  { sizeQueue di sini adalah posisi absolut pasienYangSkip dari front queue }
  sizeQueue <- globalDenahRumahSakit.kapasitasRuangan + lokasiRuangan.antrian

  pasienYangSkip <- targetQueue.front
  pasienSebelumYangSkip <- NIL { Inisialisasi, akan diisi jika pasienYangSkip
bukan front }
  pasienTerakhirDalamRuangan <- NIL

  { Cari pasienYangSkip, pasienSebelumYangSkip, dan pasienTerakhirDalamRuangan
}
  i <- 1
  while (i < sizeQueue) and (pasienYangSkip != NIL) and (pasienYangSkip.id !=
currentPatientId) do
    pasienSebelumYangSkip <- pasienYangSkip
    if (i == globalDenahRumahSakit.kapasitasRuangan) then
      pasienTerakhirDalamRuangan <- pasienYangSkip
    { akhir dari if }
    pasienYangSkip <- pasienYangSkip.next
    i <- i + 1
  { akhir dari while }

  { Lakukan operasi skip jika semua node yang dibutuhkan ditemukan (tidak NIL)
}
  if (pasienYangSkip != NIL) and (pasienSebelumYangSkip != NIL) and
(pasienTerakhirDalamRuangan != NIL) then
    { Logika C:
      tempNode <- pasienYangSkip.next
      pasienYangSkip.next <- pasienTerakhirDalamRuangan.next
      pasienTerakhirDalamRuangan.next <- pasienYangSkip { Ini yang memindahkan
pasienYangSkip }
      pasienSebelumYangSkip.next <- tempNode
    }
    tempNode <- pasienYangSkip.next

    { Lepas pasienYangSkip dari posisi lama }
    pasienSebelumYangSkip.next <- tempNode

    { Sisipkan pasienYangSkip setelah pasienTerakhirDalamRuangan }
    pasienYangSkip.next <- pasienTerakhirDalamRuangan.next
    pasienTerakhirDalamRuangan.next <- pasienYangSkip

    { Perbarui rear jika pasienYangSkip adalah elemen terakhir yang dipindahkan
}
    if (pasienSebelumYangSkip.next == NIL) then { Sebelumnya pasienYangSkip
adalah rear }
      targetQueue.rear <- pasienSebelumYangSkip
    { akhir dari if }
    { Perlu diupdate juga jika tempNode (yang menjadi next dari
pasienSebelumYangSkip) adalah NIL, maka pasienSebelumYangSkip menjadi rear baru
jika PYS bukan yang paling belakang}
    if (tempNode == NIL) and (pasienYangSkip != targetQueue.rear) then { jika
PYS bukan rear, dan elemen setelah PYS (tempNode) tidak ada, berarti PSS jadi
rear }
      targetQueue.rear <- pasienSebelumYangSkip
    { akhir dari if }

    globalDenahRumahSakit.Ruangan[roomRow][roomCol].idAntrian <- targetQueue {
Update queue di denah }

```

```

        output("")
        output("Anda berhasil maju ke depan antrian Dr. ",
getAccountName(globalDenahRumahSakit.Ruangan[roomRow][roomCol].idDokter,
DATA_TYPE_DOCTOR), " di ruangan ", karakterDariInteger(roomRow + 65), roomCol +
1, "!")
        output("Posisi antrian Anda sebelumnya: ", lokasiRuangan.antrian)
        output("Posisi antrian Anda sekarang: 1") { Setelah yang di dalam ruangan }
        output("")
    else
        output("")
        output("Skip antrian gagal karena kesalahan internal atau data tidak
konsisten.")
        output("")
    { akhir dari if }
    { akhir dari if utama }

```

```

procedure cancelAntrian()
{ Membatalkan antrian pasien saat ini dari sebuah ruangan dokter.
  Memberikan pesan status berhasil atau gagal. }

KAMUS LOKAL
    lokasiRuangan: Point
    sizeQueue: integer
    pasienYangCancel, pasienSebelumYangCancel: Address { Address ke LinkedListNode
}
    i: integer
    currentPatientId: integer
    roomRow, roomCol: integer
    targetQueue: QueueType

ALGORITMA
    currentPatientId <- globalCurrentPatient.id
    lokasiRuangan <- posisiRuanganAntrianPasien(currentPatientId)

    if (lokasiRuangan.antrian = -1) then
        output("")
        output("Cancel antrian gagal! Anda tidak sedang terdaftar dalam antrian
manapun!")
        output("")
    else if (lokasiRuangan.antrian = -2) then
        output("")
        output("Anda sudah berada di dalam ruangan ",
karakterDariInteger(lokasiRuangan.row + 65), lokasiRuangan.column + 1, " bersama
Dr. ",
getAccountName(globalDenahRumahSakit.Ruangan[lokasiRuangan.row][lokasiRuangan.col
umn].idDokter, DATA_TYPE_DOCTOR), "! Tidak bisa cancel antrian.")
        output("")
    else
        { Pasien berada di antrian tunggu }
        roomRow <- lokasiRuangan.row
        roomCol <- lokasiRuangan.column
        targetQueue <- globalDenahRumahSakit.Ruangan[roomRow][roomCol].idAntrian

        { sizeQueue di sini adalah posisi absolut pasienYangCancel dari front queue }
        sizeQueue <- globalDenahRumahSakit.kapasitasRuangan + lokasiRuangan.antrian

        pasienYangCancel <- targetQueue.front
        pasienSebelumYangCancel <- NIL

        { Cari pasienYangCancel dan pasienSebelumYangCancel }
        i <- 1
        { Loop berhenti jika i mencapai posisi pasien yang akan dicancel, atau pasien
ditemukan, atau akhir list }
        while (i < sizeQueue) and (pasienYangCancel != NIL) and (pasienYangCancel.id
!= currentPatientId) do
            pasienSebelumYangCancel <- pasienYangCancel
            pasienYangCancel <- pasienYangCancel.next
            i <- i + 1
        { akhir dari while }

        { Lakukan pembatalan jika pasienYangCancel ditemukan }
        if (pasienYangCancel != NIL) and (pasienYangCancel.id = currentPatientId)
then

```



```

        if (pasienSebelumYangCancel = NIL) then { Pasien yang dibatalkan adalah
elemen pertama (front) }
            targetQueue.front <- pasienYangCancel.next
        else
            pasienSebelumYangCancel.next <- pasienYangCancel.next
        { akhir dari if }

        if (targetQueue.rear = pasienYangCancel) then { Jika yang dihapus adalah
rear }
            targetQueue.rear <- pasienSebelumYangCancel
        { akhir dari if }

        if (targetQueue.front = NIL) then { Jika setelah delete, queue menjadi
kosong }
            targetQueue.rear <- NIL
        { akhir dari if }

        dealokasi(pasienYangCancel) { Asumsi dealokasi adalah primitif untuk node }
        targetQueue.size <- targetQueue.size - 1

        globalDenahRumahSakit.Ruangan[roomRow][roomCol].idAntrian <- targetQueue {
Update queue di denah }

        output("")
        output("Anda berhasil keluar dari antrian Dr. ",
getAccountName(globalDenahRumahSakit.Ruangan[roomRow][roomCol].idDokter,
DATA_TYPE_DOCTOR), " di ruangan ", karakterDariInteger(roomRow + 65), roomCol +
1, "!")
        output("")
    else
        output("")
        output("Pembatalan antrian gagal, pasien tidak ditemukan di posisi yang
diharapkan.")
        output("")
    { akhir dari if }
{ akhir dari if utama }

```

SCREENSHOT FITUR

1. F01

```
src > data > 25-05-2025 > user.csv > data
1 id;username;password;role;riwayat_penyakit;suhu_tubuh;tekanan_darah_sistolik;teka
2 10;neronimo;pass1010;dokter;;;;;;;;;
3 11;ciciko;pass1111;dokter;;;;;;;;;
4 12;cacako;pass1212;dokter;;;;;;;;;
5 13;kroket;pass1313;dokter;;;;;;;;;
6 15;risol;pass1515;dokter;;;;;;;;;
7 88;zeru;pass77;manager;;;;;;;;;
8 1;stewart;pass11;pasien;;36.1;92;77;66;93.7;127;52.4;177;193;328
9 7;tuart;paturrt;pasien;;36.1;92;77;66;93.7;127;52.4;177;193;328
10 2;gro;pass22;pasien;COVID-19;36.6;126;85;67;96.5;175;45.7;156;235;212
11 3;kebin;pass33;pasien;;36.4;96;68;94;92.3;162;64.1;158;240;380
12 6;nikeb;pnikeb;pasien;;36.4;96;68;94;92.3;162;64.1;158;240;380
13 4;pop;pass44;pasien;Diabetes Mellitus;36.9;110;85;73;98.7;152;62.8;157;184;390
14 8;minonette;pass88;pasien;;36.7;93;87;63;97.8;136;77;172;227;380
15 9;tobo;pass99;pasien;;36.8;114;74;61;94.6;100;80.6;173;152;386
16 10;ropik;pass110;pasien;;36.2;103;87;65;96.6;102;72.3;184;150;292
17 5;opor;oporkanajala;pasien;;36.2;103;87;65;96.6;102;72.3;184;150;292
18 16;tobokan;pass1234;pasien;;36.8;114;74;61;94.6;100;80.6;173;152;386
19 20;popokan;passpopokan;pasien;;36.9;110;85;73;98.7;152;62.8;157;184;390
20 100;pendatang;pendatangbaru;pasien;;;;;;;;;
21
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
>>> LOGIN
Masukkan username: anakMIa
Masukkan password: ooooo
USERNAME ATAU PASSWORD SALAH. SILAKAN COBA LAGI.

>>> 
```

```
20 100;pendatang;pendatangbaru;pasien;:::;:::;:::;
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

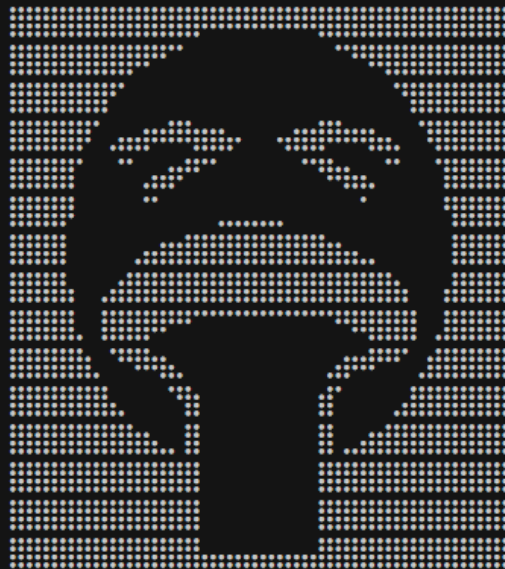
PORTS

```
>>> LOGIN
```

```
Masukkan username: pendatang
```

```
Masukkan password: pendatangbaru
```

```
SELAMAT DATANG PASIEN pendatang!
```



```
Halo pendatang
```

```
Silahkan masukan fungsi yang anda ingin jalankan
```

```
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid
```

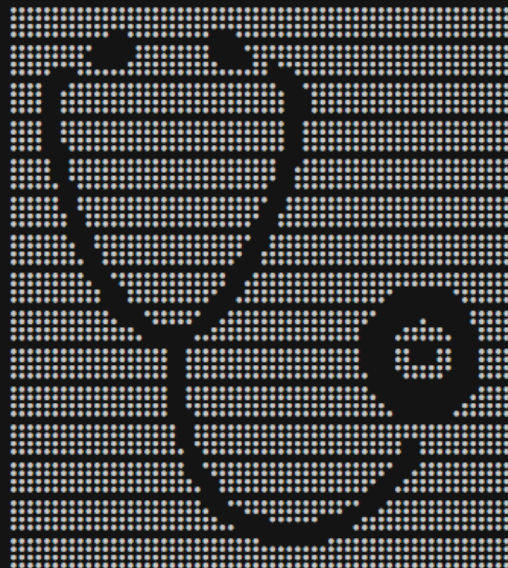
```
>>> 
```

```
>>> LOGIN
```

```
Masukkan username: ciciko
```

```
Masukkan password: pass1111
```

```
SELAMAT DATANG DOKTER ciciko!
```



```
Halo Dokter ciciko. Silahkan masukan fungsi yang anda ingin jalankan.  
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.
```

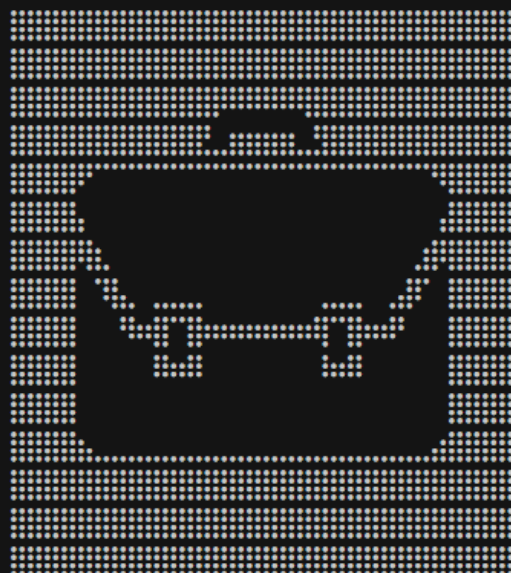
```
>>> █
```

```
>>> LOGIN
```

```
Masukkan username: zeru
```

```
Masukkan password: pass77
```

```
SELAMAT DATANG MANAGER zeru!
```



```
Halo Manager zeru. Silahkan masukan fungsi yang anda ingin jalankan.  
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.
```

```
>>> █
```

2. F02

```
>>> REGISTER
Username: newuserkelllima
Password: newpass11

Pasien newuserkelllima berhasil ditambahkan!
>>> █
```

```
Registrasi gagal! Pasien dengan nama gro sudah terdaftar.
>>> REGISTER
Username: GRo
Password: grosudahadasi

Registrasi gagal! Pasien dengan nama gro sudah terdaftar.
>>> █
```

3. F03

```
>>> LOGOUT
Sampai jumpa
=====

Anda belum masuk ke suatu akun.
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.

>>> █
```

4. F04

```
>>> LUPA_PASSWORD

Masukkan username: newuserkelllima
Masukkan kode unik: newuserkelima
Kode unik salah!
```

```
>>> LUPA_PASSWORD
```

```
Masukkan username: newuserkelllima
```

```
Masukkan kode unik: newuserke3lima
```

```
Halo newuserkelllima, silakan daftarkan ulang password anda!
```

```
Masukkan password baru: newpass22
```

```
Password berhasil diperbarui.
```

```
>>> LUPA_PASSWORD
```

```
Masukkan username: userundef
```

```
Masukkan kode unik: userundef
```

```
Username tidak ditemukan.
```

```
>>> █
```

5. F05

```
=====
```

```
Anda belum masuk ke suatu akun.
```

```
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.
```

```
>>> HELP
```

```
=====
```

```
=====
```

```
Terimakasih telah memanggil fungsi Help
```

```
Berikut merupakan fungsi-fungsi yang dapat anda gunakan
```

```
1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya
```

```
2) LOGIN : Masuki suatu akun
```

```
3) LUPA_PASSWORD : Mengganti atau memperbarui password akun
```

```
4) REGISTER : Membuat akun baru
```

```
5) EXIT : Keluar dari program
```

```
Footnote:
```

```
1) Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
```

```
2) Jangan lupa untuk memasukkan input yang valid
```

```
>>> █
```

```

>>> HELP
=====

=====
Terimakasih telah memanggil fungsi Help
Berikut merupakan fungsi-fungsi yang dapat anda gunakan
1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya
2) LIHAT_DENAH : Memunculkan denah rumah sakit
3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode ruangan)
4) PULANGDOK : Bertanya ke dokter apakah kamu sudah boleh pulang
5) DAFTAR_CHECKUP : Mendaftarkan check-up dengan dokter
6) ANTRIAN : Menunjukkan status antrian pasien
7) SKIP_ANTRIAN : Pasien akan maju ke urutan pertama antrian di luar ruangan
8) CANCEL_ANTRIAN : Pasien akan keluar dari antrian
9) MINUM_OBAT : Meminum obat yang berada di inventory
10) PENAWAR : Meminum penawar untuk memuntahkan obat yang berada di perut
11) LOGOUT : Keluar dari akun yang sedang digunakan
12) EXIT : Keluar dari program

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

```

```

>>> HELP
=====

=====
Terimakasih telah memanggil fungsi Help
Berikut merupakan fungsi-fungsi yang dapat anda gunakan
1) HELP : Memunculkan list fungsi-fungsi yang dapat digunakan beserta penjelasannya
2) LIHAT_DENAH : Memunculkan denah rumah sakit
3) LIHAT_RUANGAN XX : Memunculkan detail ruangan XX (XX: kode ruangan)
4) DIAGNOSIS : Mendiagnosis pasien yang berada di depan antrian
5) NGOBATIN : Mengobati pasien yang berada di depan antrian
6) LOGOUT : Keluar dari akun yang sedang digunakan
7) EXIT : Keluar dari program

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

```

6. F06

```
>>> LIHAT_DENAH
```

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

```
>>> █
```

```
>>> LIHAT_RUANGAN A2
```

```
--- Detail Ruangan A2 ---
```

```
Kapasitas : 3
```

```
Dokter    : ciciko
```

```
Pasien di dalam ruangan :
```

```
1. pop
```

```
2. opor
```

```
3. newuserkelllima
```

```
-----
```

```
>>> █
```

```
>>> LIHAT_RUANGAN B3
```

```
--- Detail Ruangan B3 ---
```

```
Kapasitas : 3
```

```
Dokter    : risol
```

```
Pasien di dalam ruangan :
```

```
Tidak ada pasien di dalam ruangan saat ini.
```

```
-----
```

```
>>> █
```



```
>>> LIHAT_RUANGAN B2

--- Detail Ruangan B2 ---
Kapasitas   : 3
Dokter      : -
Pasien di dalam ruangan :
    Tidak ada pasien di dalam ruangan saat ini.
-----
>>> █
```

```
>>> LIHAT_DENAH
```

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

```
>>> LIHAT_RUANGAN C9
```

```
Tidak ada ruangan dengan kode ruangan C9
```

```
>>> █
```

```

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

Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1

Menampilkan data seluruh user berdasarkan ID terurut ascending...
+-----+-----+-----+-----+
| ID | Nama | Role | Penyakit |
+-----+-----+-----+-----+
| 1 | stewart | Pasien | (Belum diperiksa dokter) |
| 2 | gro | Pasien | COVID-19 |
| 3 | kebin | Pasien | (Belum diperiksa dokter) |
| 4 | pop | Pasien | Diabetes Mellitus |
| 5 | opor | Pasien | (Belum diperiksa dokter) |
| 6 | nikeb | Pasien | (Belum diperiksa dokter) |
| 7 | tuart | Pasien | (Belum diperiksa dokter) |
| 8 | minonette | Pasien | (Belum diperiksa dokter) |
| 9 | tobo | Pasien | (Belum diperiksa dokter) |
| 10 | neronimo | Dokter | - |
| 10 | ropik | Pasien | (Belum diperiksa dokter) |
| 11 | ciciko | Dokter | - |
| 12 | cacako | Dokter | - |
| 13 | kroket | Dokter | - |
| 15 | risol | Dokter | - |
| 16 | tobokan | Pasien | (Belum diperiksa dokter) |
| 20 | popokan | Pasien | (Belum diperiksa dokter) |
| 100 | pendatang | Pasien | (Belum diperiksa dokter) |
+-----+-----+-----+-----+

>>> █

```

```
>>> LIHAT_USER
```

```
Urutkan berdasarkan?
```

```
1. ID
```

```
2. Nama
```

```
>>> Pilihan: 1
```

```
Urutan sort?
```

```
1. ASC (A-Z)
```

```
2. DESC (Z-A)
```

```
>>> Pilihan: 2
```

```
Menampilkan data seluruh user berdasarkan ID terurut descending...
```

ID	Nama	Role	Penyakit
100	pendatang	Pasien	(Belum diperiksa dokter)
20	popokan	Pasien	(Belum diperiksa dokter)
16	tobokan	Pasien	(Belum diperiksa dokter)
15	risol	Dokter	-
13	kroket	Dokter	-
12	cacako	Dokter	-
11	ciciko	Dokter	-
10	neronimo	Dokter	-
10	ropik	Pasien	(Belum diperiksa dokter)
9	tobo	Pasien	(Belum diperiksa dokter)
8	minonette	Pasien	(Belum diperiksa dokter)
7	tuart	Pasien	(Belum diperiksa dokter)
6	nikeb	Pasien	(Belum diperiksa dokter)
5	opor	Pasien	(Belum diperiksa dokter)
4	pop	Pasien	Diabetes Mellitus
3	kebin	Pasien	(Belum diperiksa dokter)
2	gro	Pasien	COVID-19
1	stewart	Pasien	(Belum diperiksa dokter)

```
>>> 
```

```
>>> LIHAT_USER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 3

Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 
```

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

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data seluruh user berdasarkan nama terurut ascending...

ID	Nama	Role	Penyakit
12	cacako	Dokter	-
11	ciciko	Dokter	-
2	gro	Pasien	COVID-19
3	kebin	Pasien	(Belum diperiksa dokter)
13	kroket	Dokter	-
8	minonette	Pasien	(Belum diperiksa dokter)
10	neronimo	Dokter	-
6	nikeb	Pasien	(Belum diperiksa dokter)
5	opor	Pasien	(Belum diperiksa dokter)
100	pendatang	Pasien	(Belum diperiksa dokter)
4	pop	Pasien	Diabetes Mellitus
20	popokan	Pasien	(Belum diperiksa dokter)
15	risol	Dokter	-
10	ropik	Pasien	(Belum diperiksa dokter)
1	stewart	Pasien	(Belum diperiksa dokter)
9	tobo	Pasien	(Belum diperiksa dokter)
16	tobokan	Pasien	(Belum diperiksa dokter)
7	tuart	Pasien	(Belum diperiksa dokter)

```
>>> █
```

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

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2
```

Menampilkan data seluruh user berdasarkan nama terurut descending...

ID	Nama	Role	Penyakit
7	tuart	Pasien	(Belum diperiksa dokter)
16	tobokan	Pasien	(Belum diperiksa dokter)
9	tobo	Pasien	(Belum diperiksa dokter)
1	stewart	Pasien	(Belum diperiksa dokter)
10	ropik	Pasien	(Belum diperiksa dokter)
15	risol	Dokter	-
20	popokan	Pasien	(Belum diperiksa dokter)
4	pop	Pasien	Diabetes Mellitus
100	pendatang	Pasien	(Belum diperiksa dokter)
5	opor	Pasien	(Belum diperiksa dokter)
6	nikeb	Pasien	(Belum diperiksa dokter)
10	neronimo	Dokter	-
8	minonette	Pasien	(Belum diperiksa dokter)
13	kroket	Dokter	-
3	kebin	Pasien	(Belum diperiksa dokter)
2	gro	Pasien	COVID-19
11	ciciko	Dokter	-
12	cacako	Dokter	-

```
>>> █
```

```
>>> LIHAT_PASIEN
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data pasien berdasarkan ID terurut ascending...

+-----+-----+-----+		
ID	Nama	Penyakit
+-----+-----+-----+		
1	stewart	(Belum diperiksa dokter)
2	gro	COVID-19
3	kebin	(Belum diperiksa dokter)
4	pop	Diabetes Mellitus
5	opor	(Belum diperiksa dokter)
6	nikeb	(Belum diperiksa dokter)
7	tuart	(Belum diperiksa dokter)
8	minonette	(Belum diperiksa dokter)
9	tobo	(Belum diperiksa dokter)
10	ropik	(Belum diperiksa dokter)
16	tobokan	(Belum diperiksa dokter)
20	popokan	(Belum diperiksa dokter)
100	pendatang	(Belum diperiksa dokter)
+-----+-----+-----+		

```
>>> █
```

```
>>> LIHAT_PASIEN
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2
```

Menampilkan data pasien berdasarkan ID terurut descending...

ID	Nama	Penyakit
100	pendatang	(Belum diperiksa dokter)
20	popokan	(Belum diperiksa dokter)
16	tobokan	(Belum diperiksa dokter)
10	ropik	(Belum diperiksa dokter)
9	tobo	(Belum diperiksa dokter)
8	minonette	(Belum diperiksa dokter)
7	tuart	(Belum diperiksa dokter)
6	nikeb	(Belum diperiksa dokter)
5	opor	(Belum diperiksa dokter)
4	pop	Diabetes Mellitus
3	kebin	(Belum diperiksa dokter)
2	gro	COVID-19
1	stewart	(Belum diperiksa dokter)

```
>>> █
```



```
>>> LIHAT_PASIEN
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data pasien berdasarkan nama terurut ascending...

ID	Nama	Penyakit
2	gro	COVID-19
3	kebin	(Belum diperiksa dokter)
8	minonette	(Belum diperiksa dokter)
6	nikeb	(Belum diperiksa dokter)
5	opor	(Belum diperiksa dokter)
100	pendatang	(Belum diperiksa dokter)
4	pop	Diabetes Mellitus
20	popokan	(Belum diperiksa dokter)
10	ropik	(Belum diperiksa dokter)
1	stewart	(Belum diperiksa dokter)
9	tobo	(Belum diperiksa dokter)
16	tobokan	(Belum diperiksa dokter)
7	tuart	(Belum diperiksa dokter)

```
>>>
```

```
>>> LIHAT_PASIEN
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2
```

Menampilkan data pasien berdasarkan nama terurut descending...

ID	Nama	Penyakit
7	tuart	(Belum diperiksa dokter)
16	tobokan	(Belum diperiksa dokter)
9	tobo	(Belum diperiksa dokter)
1	stewart	(Belum diperiksa dokter)
10	ropik	(Belum diperiksa dokter)
20	popokan	(Belum diperiksa dokter)
4	pop	Diabetes Mellitus
100	pendatang	(Belum diperiksa dokter)
5	opor	(Belum diperiksa dokter)
6	nikeb	(Belum diperiksa dokter)
8	minonette	(Belum diperiksa dokter)
3	kebin	(Belum diperiksa dokter)
2	gro	COVID-19

```
>>> █
```

```
>>> LIHAT_DOKTER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data dokter berdasarkan ID terurut ascending...

ID	Nama
10	neronimo
11	ciciko
12	cacako
13	kroket
15	risol

```
>>> 
```

```
>>> LIHAT_DOKTER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2
```

```
Menampilkan data dokter berdasarkan ID terurut descending...
```

ID	Nama
15	risol
13	kroket
12	cacako
11	ciciko
10	neronimo

```
>>> █
```

```
>>> LIHAT_DOKTER
Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2
```

```
Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 1
```

Menampilkan data dokter berdasarkan nama terurut ascending...

ID	Nama
12	cacako
11	ciciko
13	kroket
10	neronimo
15	risol

```
>>> 
```

```

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

Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2

Menampilkan data dokter berdasarkan nama terurut descending...
+-----+-----+
| ID   | Nama           |
+-----+-----+
| 15   | risol          |
| 10   | neronimo       |
| 13   | kroket         |
| 11   | ciciko         |
| 12   | cacako         |
+-----+-----+

>>> █

```

8. F08

```

>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

>>> Masukkan nomor ID user: 88

Tidak ditemukan pengguna dengan nomor ID 88!

>>> █

```

```
>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

>>> Masukkan nomor ID user: 9

Menampilkan pengguna dengan nomor ID 9...
+-----+-----+-----+-----+
| ID    | Nama          | Role    | Penyakit                               |
+-----+-----+-----+-----+
| 9     | tobo          | Pasien  | (Belum diperiksa dokter)            |
+-----+-----+-----+-----+

>>> █
```

```
>>> CARI_USER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2

>>> Masukkan nama user: meong

Tidak ditemukan pengguna dengan nama meong!

>>> █
```

```
1. ID
2. Nama
>>> Pilihan: 2
```

```
Menampilkan pengguna dengan nama minonette...
```

ID	Nama	Role	Penyakit
8	minonette	Pasien	(Belum diperiksa dokter)

```
1. ID
2. Nama
3. Penyakit
>>> Pilihan: 3
```

Menampilkan pasien dengan penyakit Diabetes Mellitus...

```
+-----+-----+
| ID    | Nama      | Penyakit |
+-----+-----+
| 4     | pop       | Diabetes Mellitus |
+-----+-----+
```



```
>>> CARI_DOKTER
Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

>>> Masukkan nomor ID dokter: 9

Tidak ditemukan dokter dengan nomor ID 9!

>>> █
```

9. F09

```

>>> LIHAT_SEMUA_ANTRIAN

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

===== A1 =====
Kapabilitas : 3
Dokter      : neronimo
Pasien di dalam ruangan :
  1. gro
  2. kebin
  3. stewart
Pasien di antrian:
  1. tobokan
  2. popokan

===== A2 =====
Kapabilitas : 3
Dokter      : ciciko
Pasien di dalam ruangan :
  1. pop
  2. opor
Pasien di antrian:
  Tidak ada pasien di antrian saat ini.

===== A3 =====
Kapabilitas : 3
Dokter      : cacako
Pasien di dalam ruangan :
  1. nikeb
Pasien di antrian:
  Tidak ada pasien di antrian saat ini.

===== B1 =====
Kapabilitas : 3
Dokter      : kroket
Pasien di dalam ruangan :
  1. minonette
  2. tuart
Pasien di antrian:
  Tidak ada pasien di antrian saat ini.

===== B3 =====
Kapabilitas : 3
Dokter      : risol
Pasien di dalam ruangan :
  Tidak ada pasien di dalam ruangan saat ini.
Pasien di antrian:
  Tidak ada pasien di antrian saat ini.

>>> 

```

10. F10

```
>>> TAMBAH_DOKTER
Username: ciciko
Password: cicikoudahadadidatabasesi

Sudah ada dokter bernama ciciko!
>>> █
```

```
>>> TAMBAH_DOKTER
Username: dokterganesha
Password: hidupkangane

Dokter dokterganesha berhasil ditambahkan!
>>> █
```

```
>>> ASSIGN_DOKTER
Username: meong

Dokter dengan username meong tidak ditemukan.
>>> █
```

```
>>> ASSIGN_DOKTER
Username: dokterganesha

Ruangan: A1

Dokter DokterLain sudah menempati ruangan A1!
Silakan cari ruangan lain untuk dokter dokterganesha.
>>> █
```

```
>>> DIAGNOSIS
CHECKING Influenza
CHECKING Anemia
CHECKING COVID-19
CHECKING Hipertensi
CHECKING Diabetes Mellitus

newuserkelllima terdiagnosa penyakit Diabetes Mellitus!
>>> █
```

```
>>> DIAGNOSIS
newuserkelllima Telah Didiagnosa
>>> █
```

12. F12

```
>>> NGOBATIN
Dokter sedang mengobati pasien newuserkelllima
Pasien memiliki penyakit Diabetes Mellitus
Obat yang harus diberikan:
1. Metformin
2. Lisinopril
3. Remdesivir
4. Vitamin C
>>> █
```

```
>>> NGOBATIN

Pasien sudah diobatin!
>>> █
```

13. F13

```
>>> PULANGDOK

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

```
Dokter sedang memeriksa keadaanmu...
```

```
Masih ada obat yang belum kamu habiskan, minum semuanya dulu yuk!
```

```
>>> █
```

```
>>> PULANGDOK
```

```
Dokter sedang memeriksa keadaanmu...
```

```
Maaf, tapi kamu masih belum bisa pulang!
```

```
Urutan peminuman obat yang diharapkan:
```

```
Metformin -> Lisinopril -> Remdesivir -> Vitamin C
```

```
Urutan obat yang kamu minum:
```

```
Metformin -> Lisinopril -> Vitamin C -> Remdesivir
```

```
Silahkan kunjungi dokter untuk meminta penawar yang sesuai!
```

```
>>> █
```

```
>>> PULANGDOK
```

```
Dokter sedang memeriksa keadaanmu...
```

```
Dokter sedang memeriksa keadaanmu...
```

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

```
>>> █
```

14. F14

```
>>> DAFTAR_CHECKUP
Silahkan masukkan data checkup Anda:
Suhu tubuh (celecius): 36.5
Tekanan darah (sistol/diastol, contoh 120 80): 140 89
Detak jantung (bpm): 99
Saturasi oksigen: 99
Kadar gula darah (mg/dL): 200
Berat badan (kg): 90
Tinggi badan (cm): 150
Kadar kolestrol (mg/dL): 230
Trombosit (ribu/ $\mu$ L): 410

Berikut adalah daftar dokter yang tersedia:
1. Dr. neronimo - Spesialisasi - Ruangan A1 (Antrian: 2 orang)
2. Dr. ciciko - Spesialisasi - Ruangan A2 (Ruangan belum penuh)
3. Dr. cacako - Spesialisasi - Ruangan A3 (Ruangan belum penuh)
4. Dr. kroket - Spesialisasi - Ruangan B1 (Ruangan belum penuh)
5. Dr. risol - Spesialisasi - Ruangan B3 (Ruangan belum penuh)

Pilih dokter (1 - 5): 5

Pendaftaran check-up berhasil!
Anda terdaftar pada antrian Dr. risol di ruangan B3.
Anda dapat langsung masuk ke dalam ruangan.

>>> █
```

```
>>> DAFTAR_CHECKUP
Anda sudah terdaftar dalam antrian check-up!
Silakan selesaikan check-up yang sudah terdaftar terlebih dahulu.

>>> █
```

15. F15

```
>>> ANTRIAN

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

>>> █
```

```
>>> ANTRIAN
```

```
Anda sedang berada di dalam ruangan dokter!
```

```
>>> █
```

16. F16

```
>>> MINUM_OBAT
```

```
Anda belum melakukan diagnosis. Lakukan diagnosis dengan command DIAGNOSIS.
```

```
>>> █
```

```
>>> MINUM_OBAT
```

```
===== DAFTAR OBAT =====
```

```
1. Vitamin C
```

```
Pilih Obat untuk diminum: 2
```

```
Pilihan nomor tidak tersedia!
```

```
Pilih Obat untuk diminum: █
```

```
>>> MINUM_OBAT
```

```
===== DAFTAR OBAT =====
```

```
1. Metformin
```

```
2. Lisinopril
```

```
3. Remdesivir
```

```
4. Vitamin C
```

```
Pilih Obat untuk diminum: 1
```

```
GLEKGLEKGLEK... Metformin berhasil diminum!!!
```

17. F17

```
>>> PENAWAR
Perut kosong, belum ada obat yang diminum.
>>>
```

18. F18

```
>>> EXIT
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n): y
Masukkan nama folder (contoh: data/hari_ini): data/31-05-2025
Membuat folder data/31-05-2025 ...
SAVED USER DATABASE!
SAVED OBAT DATABASE!
SAVED PENYAKIT DATABASE!
SAVED OBAT PENYAKIT DATABASE!
Data berhasil disimpan di folder data/31-05-2025!
Sampai jumpa, Niemons!
yzks1@DESKTOP-SFQ2E9N:/mnt/c/Users/Legion/Documents/GitHub/if1210-tubes-2025-k05-m/src$
```

19. D03

```
yksl@DESKTOP-SFQ2E9N:/mnt/c/Users/Legion/Documents/GitHub/if1210-tubes-2025-k05-m/src$ ./main data/25-05-2025
Loading...
LOADED USER DATABASE!
LOADED OBAT DATABASE!
LOADED PENYAKIT DATABASE!
LOADED OBAT PENYAKIT DATABASE!
Finished Loading!
Selamat datang di rumah sakit Niemons!
=====
```

```
=====
Anda belum masuk ke suatu akun.
Masukan HELP untuk memunculkan list fungsi-fungsi yang valid.
```

```
>>> █
```

20. D04


```
src > data > 31-05-2025 > user.csv > data
1 id;username;password;role;riwayat_penyakit;suhu_tubuh;tekanan_darah_sistolik;tekanan_darah_dias
2 10;neronimo;pass1010;dokter;;;;;;;;;;
3 11;ciciko;pass1111;dokter;;;;;;;;;;
4 12;cacako;pass1212;dokter;;;;;;;;;;
5 13;kroket;pass1313;dokter;;;;;;;;;;
6 15;risol;pass1515;dokter;;;;;;;;;;
7 88;zeru;pass77;manager;;;;;;;;;;
8 1;stewart;pass11;pasien;;36.1;92;77;66;93.7;127;52.4;177;193;328
9 7;tuart;paturrt;pasien;;36.1;92;77;66;93.7;127;52.4;177;193;328
10 2;gro;pass22;pasien;COVID-19;36.6;126;85;67;96.5;175;45.7;156;235;212
11 3;kebin;pass33;pasien;;36.4;96;68;94;92.3;162;64.1;158;240;380
12 6;nikeb;pnikeb;pasien;;36.4;96;68;94;92.3;162;64.1;158;240;380
13 4;pop;pass44;pasien;Diabetes Mellitus;36.9;110;85;73;98.7;152;62.8;157;184;390
14 8;minonette;pass88;pasien;;36.7;93;87;63;97.8;136;77;172;227;380
15 9;tobo;pass99;pasien;;36.8;114;74;61;94.6;100;80.6;173;152;386
16 10;ropik;pass110;pasien;;36.2;103;87;65;96.6;102;72.3;184;150;292
17 5;opor;oporkanajala;pasien;;36.2;103;87;65;96.6;102;72.3;184;150;292
18 16;tobokan;pass1234;pasien;;36.8;114;74;61;94.6;100;80.6;173;152;386
19 20;popokan;passpopokan;pasien;;36.9;110;85;73;98.7;152;62.8;157;184;390
20 100;pendatang;pendatangbaru;pasien;;;;;;;;;;
21 101;newuserkelllima;newpass22;pasien;Diabetes Mellitus;36.5;140;89;99;99;200;90;150;230;410
22
```

```
21 101;newuserkelllima;newpass22;pasien;;;;;;;;;;
22
```

```
src > data > 25-05-2025 > config.txt
```

```
1 2 3
2 3 4
3 10 2 3 1 16 20
4 11 4 5
5 12 6
6 13 8 7
7 0
8 15 0
9 2
10 2 3
11 4 3 2
12 1
13 4 4 5
14
```

```
src > data > 31-05-2025 > ≡ config.txt
1    2 3
2    3 4
3    10 2 3 1 16 20
4    11 4 5
5    12 6
6    13 8 7
7    0 ✨ ✨
8    15 101
9    3
10   2 3
11   4 3 2
12   101 3 2 4 5
13   1
14   4 4 5
15
```

21. B02

```
>>> PINDAH_DOKTER
A1 B1
Pemindahan gagal. Ruangan B1 Sudah ditempati.
```

```
Pemindahan gagal. Ruangan A23 Kosong.
>>> PINDAH_DOKTER A1 B2
Dr. neronimo berhasil dipindahkan dari ruangan A1 ke ruangan B2.
>>> █
```

```
>>> UBAH_DENAH
1 1
Tidak dapat mengubah ukuran denah. Ruangan A2 masih ditempati oleh Dr. ciciko. Silakan pindahkan dokter terlebih dahulu.
>>> █
```

```
>>> LIHAT_DENAH
```

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

```
>>> UBAH_DENAH
```

```
4 10
```

```
Denah rumah sakit berhasil diubah menjadi 4 baris dan 10 kolom.
```

```
>>> LIHAT_DENAH
```

```
      1      2      3      4      5      6      7      8      9     10
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
A | A1  | A2  | A3  | A4  | A5  | A6  | A7  | A8  | A9  | A10 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
B | B1  | B2  | B3  | B4  | B5  | B6  | B7  | B8  | B9  | B10 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
C | C1  | C2  | C3  | C4  | C5  | C6  | C7  | C8  | C9  | C10 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
D | D1  | D2  | D3  | D4  | D5  | D6  | D7  | D8  | D9  | D10 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

```
>>> █
```

```
>>> ASSIGN_DOKTER
```

```
Username: dokterganesha
```

```
Ruangan: A1
```

```
Dokter dokterganesha berhasil diassign ke ruangan A1!
```

22. B06

```
>>> SKIP_ANTRIAN
```

```
Skip antrian gagal! Anda tidak sedang terdaftar dalam antrian manapun!
```

```
>>> █
```

```
>>> CANCEL_ANTRIAN
```

```
Cancel antrian gagal! Anda tidak sedang terdaftar dalam antrian manapun!
```

```
>>> █
```

```
>>> SKIP_ANTRIAN
```

```
Anda sudah berada di dalam ruangan A2 bersama Dr. ciciko! Tidak bisa skip antrian lagi.
```

LAMPIRAN

Link Repository: <https://github.com/Labpro-22/if1210-tubes-2025-k05-m>