

**LAPORAN MILESTONE 2 TUGAS BESAR**  
**ALGORITMA DAN PEMROGRAMAN (IF1210)**



**KELOMPOK K04-I**

Anggota Kelompok :

Abel Gani	18224016
Muhammad Zulfa Fauzan N.	18224064
Almer Zain Farisseno	18224070
Anisa Aulia Alhaqi	18224080
Endda Tsa Azzahra Syaifur	18224094

**INSTITUT TEKNOLOGI BANDUNG**  
**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA - KOMPUTASI**  
**TAHUN 2025**

## **HALAMAN PERNYATAAN**

*“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.”*

### **Kelompok K4-I**

Abel Gani	18224016
Muhammad Zulfa Fauzan N.	18224064
Almer Zain Farisseno	18224070
Anisa Aulia Alhaqi	18224080
Endda Tsa Azzahra Syaifur	18224094

## DAFTAR ISI

<b>HALAMAN PERNYATAAN.....</b>	<b>1</b>
<b>DAFTAR ISI.....</b>	<b>2</b>
<b>DAFTAR TABEL.....</b>	<b>4</b>
<b>DESKRIPSI PERSOALAN.....</b>	<b>5</b>
<b>RENCANA IMPLEMENTASI.....</b>	<b>6</b>
<b>PEMBAGIAN KERJA ANGGOTA.....</b>	<b>9</b>
<b>HASIL RANCANGAN, IMPLEMENTASI, TESTING.....</b>	<b>11</b>
<b>DESAIN COMMAND PROGRAM.....</b>	<b>13</b>
<b>DESAIN KAMUS DATA.....</b>	<b>18</b>
<b>FLOWCHART PROGRAM.....</b>	<b>27</b>
F01 - Login.....	27
F02 - Register.....	28
F03 - Logout.....	29
F04 - Lupa Password.....	30
F05 - Menu & Help.....	31
F06 - Denah Rumah Sakit.....	32
F07 - Lihat User.....	33
F08 - Cari User.....	34
F09 - Lihat Antrian.....	35
F10 - Tambah Dokter.....	36
F11 - Diagnosis.....	37
F12 - Ngobatin.....	38
F13 - Aku boleh pulang, dok?.....	39
F14 - Daftar Check-Up.....	40
F15 - Antrian Saya.....	41
F16 - Minum Obat.....	42
F17 - Minum Penawar.....	43
F18 - Exit.....	44
<b>SPESIFIKASI PROGRAM.....</b>	<b>44</b>
F01 - Login.....	44
F02 - Register Pasien.....	48
F03 - Logout.....	53
F04 - Lupa Password.....	55
F05 - Menu & Help.....	58
F06 - Denah Rumah Sakit.....	65

F07 - Lihat User.....	73
F08 - Cari User.....	81
F09 - Lihat Antrian.....	109
F10 - Tambah Dokter.....	117
F11 - Diagnosis.....	123
F12 - Ngobatin.....	131
F13 - Aku boleh pulang ga, dok?.....	131
F14 - Daftar Check-Up.....	140
F15 - Antrian Saya!.....	156
F16 - Minum Obat.....	165
F17 - Minum Penawar.....	169
F18 - Exit.....	171
B03 - Aura!.....	173
B07 - Search Suggestion.....	174
<b>PROGRAM UTAMA.....</b>	<b>176</b>
Main: hospitalSystem.c.....	176
<b>TANGKAPAN LAYAR.....</b>	<b>203</b>
F01 - Login.....	203
F02 - Register Pasien.....	204
F03 - Logout.....	205
F04 - Lupa Password.....	205
F05 - Menu & Help.....	206
F06 - Denah Rumah Sakit.....	209
F07 - Lihat User.....	211
F08 - Cari User.....	217
F09 - Lihat Antrian.....	222
F10 - Tambah Dokter.....	223
F11 - Diagnosis.....	224
F12 - Ngobatin.....	225
F13 - Aku boleh pulang ga, dok?.....	225
F14 - Daftar Check-Up.....	226
F15 - Antrian Saya!.....	228
F16 - Minum Obat.....	228
F17 - Minum Penawar.....	229
F18 - Exit.....	229
B07 - Search Suggestion.....	230
<b>LAMPIRAN.....</b>	<b>231</b>

## **DAFTAR TABEL**

Tabel 1. Rencana Implementasi ADT.....	6
Tabel 2. Pembagian Kerja Anggota.....	9
Tabel 3. Checklist Rancangan, Implementasi, dan Testing Primitif.....	11

## **DESKRIPSI PERSOALAN**

Dalam tugas besar ini, kami diharuskan membuat sebuah sistem rumah sakit yang terorganisir. Setiap fungsi (F-XX) merupakan fungsionalitas program yang implementasinya memanfaatkan materi dari mata kuliah Algoritma dan Pemrograman yang telah diberikan. Pemrograman modularitas yang memanfaatkan fungsi dan prosedur, algoritma *search*, *sort*, *filter*, hingga penerapan ADT dapat dikombinasikan untuk membuat sebuah fungsi yang berjalan sesuai kebutuhan.

Pada F01, F02, F03, dan F04 kebutuhan program berkaitan dengan akses akun yang terdiri dari *username*, *password*, dan *role*. Masing-masing *role* yang terdiri dari pasien, manager, dan dokter akan memiliki eksplorasi yang berbeda-beda pada setiap fungsi (F-XX). Fungsi register hanya dapat menambahkan pasien, sementara login, logout, dan lupa password dapat diakses seluruh *role*. F05 untuk menu dan help membantu setiap *role* untuk mendapat panduan dari beberapa probabilitas kejadian. Sementara F18 sebagai fungsi exit berguna untuk menyelesaikan keberjalanan sistem dan keluar program.

F06 berkaitan dengan denah rumah sakit yang menggunakan ADT List untuk menyusun ruangan yang terdiri dari kapasitas ruangan, dokter yang bertugas, dan pasien yang dirawat. F07, F08, dan F09 hanya dapat diakses oleh seorang manajer yang mampu menampilkan seluruh data dokter atau pasien dan dapat mencarinya. F10 kembali memberikan akses kepada manajer untuk menambahkan dokter dengan memanfaatkan ADT Set ke ruangan yang tersedia.

Selanjutnya F11 dan F12 memberi akses kepada dokter untuk mendiagnosis penyakit dan memberi obat yang sesuai dalam rangka mengobati pasien. Sementara F13 hingga F17 memberikan akses kepada pasien berkaitan dengan obat dan antrian yang dijalani pasien. Untuk F15 berkaitan dengan antrian menggunakan ADT Queue yang memanfaatkan prinsip *First In, First Out* (FIFO). Lalu F16 dan F17 memanfaatkan ADT Stack dengan prinsip *Last In, First Out* (LIFO).

## RENCANA IMPLEMENTASI

Tabel 1. Rencana Implementasi ADT

Implementasi ADT	Fitur	Deskripsi Implementasi	Alasan Implementasi
Prosedur	F01 - Login	Digunakan untuk mengakses akun, dengan username serta password yang sesuai.	Prosedur digunakan untuk pengecekan kesesuaian antara password dan username.
Prosedur, ADT List	F02 - Register	Digunakan untuk mendaftarkan akun pasien baru, yang kemudian akan di store di ADT List.	Prosedur digunakan untuk menginput dan memindahkan data hasil input ke ADT List. ADT List digunakan untuk memudahkan pengolahan list akun dalam jumlah banyak.
Prosedur	F03 - Logout	Digunakan untuk keluar dari akun, tetapi hanya berhasil apabila akun telah ter-login.	Prosedur digunakan untuk memvalidasi status_login, kemudian menyatakan status logout yang berhasil atau tidak.
Prosedur	F04 - Lupa Password	Digunakan untuk mengubah password yang di simpan di ADT List.	Fungsi digunakan untuk generate RLE username untuk kode unik, prosedur digunakan untuk validasi kode unik dan lanjut ke penggantian apabila benar.
Prosedur	F05 - Menu & Help	Digunakan untuk memberi bantuan pada dokter, pasien, dan manager	Hanya mengeluarkan pesan yang terhubung pada fungsi lain, seperti logout atau diagnosis
ADT List	F06 - Denah Rumah Sakit	Digunakan untuk melihat denah ruangan di rumah sakit beserta detail dari ruangan	Denah berbentuk memanjang dan terurut
Sort	F07 - Lihat User	Digunakan untuk melihat data seluruh pengguna, baik dokter maupun pasien	Perlu melakukan pengurutan data pasien, baik <i>ascending</i> maupun <i>descending</i>
Binary Search	F08 - Cari User	Digunakan untuk mencari data pengguna (dokter,	Melakukan pencarian data dengan jumlah yang besar

		pasien) secara spesifik berdasarkan ID atau Nama	
ADT Queue, Prosedur	F09 - Lihat Antrian	Digunakan untuk mencari data denah beserta data masing masing informasi dalam ruangan, mencakup dokter, pasien dalam ruangan, dan pasien dalam antrian.	Prosedur digunakan untuk mengolah kumpulan data data yang ada didalam ADT ruangan.
ADT Sederhana, ADT List, Prosedur	F10 - Tambah Dokter	Prosedur yang ada akan menginisialisasi ADT sederhana dokter, kemudian di tambahkan ke ADT List kumpulan dokter.	ADT sederhana dan ADT List dapat diproses dengan mudah. Prosedur digunakan untuk mengolah data input.
Fungsi	F11 - Diagnosis	Fungsi digunakan untuk mengolah data pasien berdasarkan template kasus yang ada, sehingga dapat terjadi diagnosis otomatis.	Fungsi dapat mengembalikan nilai apakah diagnosisnya berhasil dilakukan atau tidak
ADT List, Prosedur	F12 - Ngobatin	Prosedur bertugas mengotomatisasi pengobatan berdasarkan pencocokan id obat dengan id penyakit	Karena ada korelasi antara id obat dan id penyakit dalam pengobatan otomatis yang dijalankan dalam fungsi ini.
ADT List	F13 - Aku boleh pulang ga, dok?	Digunakan untuk membandingkan dan validasi urutan obat yang benar	Karena hanya mencocokkan urutan obat tanpa perlu mengeluarkan
ADT Map, Queue linked list	F14 - Daftar Check-Up	Digunakan untuk memilih dokter lalu <i>assign</i> antrian	ADT Map memiliki algoritma key dan value untuk persoalan dokter dan antrian, sementara ADT Queue untuk antrian yang sistemnya FIFO
ADT Queue	F15 - Antrian Saya!	Digunakan untuk mendaftarkan antrian pasien	Karena pasien pertama akan dilayani dulu (FIFO)
ADT Stack	F16 - Minum Obat	Digunakan untuk menyimpan obat secara urut dan obat terakhir berada paling atas	Karena obat pertama berada di paling bawah, dan obat akhir berada paling atas (LIFO)
ADT Stack	F17 - Minum Penawar	Digunakan untuk	Supaya obat yang terakhir

		menawarkan atau mengeluarkan obat yang terakhir kali diminum	diminum ada di urutan awal dan mudah untuk dikeluarkan
Prosedur	F18 - Exit	Keluar dari program	Mengakhiri program
Prosedur, Fungsi	B02 - Denah Dinamis	Mengubah ukuran denah rumah sakit	Prosedur digunakan karena tidak secara langsung mengembalikan nilai data spesifik yang diolah sebagai hasil utamanya. Fungsi digunakan untuk mengelola pergerakan entitas dokter antar ruangan dalam sistem.
Prosedur	B03 - Aura!	Pendataan aura milik dokter	Tidak memerlukan ADT dalam implementasi
ADT Sederhana, Prosedur, Fungsi	B04 - Banarich!!!	Memberikan fitur gacha kepada pasien dengan hadiah berupa saldo, sementara prosedur digunakan untuk melihat saldo keuangan rumah sakit serta saldo dompet dokter atau pasien yang sedang aktif.	Fungsi pada fitur gacha digunakan untuk mengembalikan nilai true/false, sementara prosedur memanfaatkan ADT sederhana dan hanya menampilkan informasi berupa saldo tanpa mengembalikan nilai
Prosedur	B05 - Dead or Alive?!	Pendataan untuk pengelolaan sisa nyawa pasien setelah minum obat	Tidak memerlukan ADT dalam implementasi
Queue, Fungsi	B06 - Mainin Antrian	Digunakan untuk melakukan skip ataupun cancel antrian	Karena berbasis antrian dan FIFO
Prosedur	B07 - Search Suggestion	Memberikan rekomendasi username yang memiliki substring dari nama yang dicari saat username tidak ada pada data	Prosedur digunakan untuk melakukan pengecekan terhadap ketersediaan substring pada username

## PEMBAGIAN KERJA ANGGOTA

Tabel 2. Pembagian Kerja Anggota

<b>Fitur</b>	<b>Implementasi *)</b>	<b>NIM Desainer **)</b>	<b>NIM Coder **)</b>	<b>NIM Tester **)</b>
F01 - Login	procedure loginAkun	18224094	18224094	18224016, 18224064, 18224070, 18224080, 18224094
F02 - Register	procedure registerAkun, ADT List	18224094	18224094	18224016, 18224064, 18224070, 18224080, 18224094
F03 - Logout	Prosedur	18224016	18224016	18224016, 18224064, 18224070, 18224080, 18224094
F04 - Lupa Password	Fungsi dan Prosedur	18224016	18224016	18224016, 18224064, 18224070, 18224080, 18224094
F05 - Menu & Help	Prosedur	18224070	18224070	18224016, 18224064, 18224070, 18224080, 18224094
F06 - Denah Rumah Sakit	ADT List	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
F07 - Lihat User	Sort	18224080	18224080	18224016, 18224064, 18224070, 18224080, 18224094
F08 - Cari User	Binary Search	18224080	18224080	18224016, 18224064, 18224070, 18224080, 18224094
F09 - Lihat Antrian	ADT Queue	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
F10 - Tambah Dokter	Prosedur	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
F11 - Diagnosis	ADT List, Prosedur	18224016	18224016	18224016, 18224064, 18224070, 18224080, 18224094
F12 - Ngobatin	ADT List, Prosedur	18224016	18224016	18224016, 18224064, 18224070, 18224080, 18224094

F13 - Aku boleh pulang ga, dok?	ADT List	18224070	18224070	18224016, 18224064, 18224070, 18224080, 18224094
F14 - Daftar Check-Up	ADT Map, Queue linked list	18224070	18224070	18224016, 18224064, 18224070, 18224080, 18224094
F15 - Antrian Saya!	ADT Queue	18224080	18224080	18224016, 18224064, 18224070, 18224080, 18224094
F16 - Minum Obat	ADT Stack	18224094	18224094	18224016, 18224064, 18224070, 18224080, 18224094
F17 - Minum Penawar	ADT Stack	18224094	18224094	18224016, 18224064, 18224070, 18224080, 18224094
F18 - Exit	Prosedur	18224070	18224070	18224016, 18224064, 18224070, 18224080, 18224094
B01 - Git Best Practice	-	18224016, 18224064, 18224070, 18224080, 18224094	18224016, 18224064, 18224070, 18224080, 18224094	18224016, 18224064, 18224070, 18224080, 18224094
B02 - Denah Dinamis	Prosedur, Fungsi	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
B03 - Aura!	-	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
B04 - Banarich!!!	ADT Sederhana, Prosedur, Fungsi	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
B05 - Dead or Alive?!	-	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
B06 - Mainin Antrian	Queue, Fungsi	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
B07 - Search Suggestion	Prosedur	18224064, 18224080	18224064, 18224080	18224016, 18224064, 18224070, 18224080, 18224094

## HASIL RANCANGAN, IMPLEMENTASI, TESTING

Tabel 3. Checklist Rancangan, Implementasi, dan Testing 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 Rumah Sakit	v	v	v
F07 - Lihat User	v	v	v
F08 - Cari User	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
B01 - Git Best Practice	v	v	v
B02 - Denah Dinamis	v	v	v
B03 - Aura!	v	v	v

B04 - Banarich!!!	v	v	x
B05 - Dead or Alive?!	v	v	x
B06 - Mainin Antrian	v	v	v
B07 - Search Suggestion	v	v	v

## DESAIN COMMAND PROGRAM

```
>>> Fungsi F01 - Login
function login (input hospital: pointer to Hospital, input/output
session: pointer to Session, input username: String, input password:
String) → boolean
    { I.S. Menerima pointer ke struktur Hospital dan Session, serta
    username dan password sebagai String.
        F.S. Mengembalikan true jika login berhasil, false jika gagal.
            Jika berhasil, session diperbarui dengan userId, username,
            role, dan isLoggedIn menjadi true.
            Jika gagal, pesan error yang sesuai ditampilkan.
    }

>>> Fungsi F02 - Register Pasien
function registerPatient(input hospital: pointer to Hospital,
input/output session: pointer to Session, input inputUsername: String,
input password: String) → boolean
    { I.S. Menerima pointer ke struktur Hospital dan Session, serta
    username dan password untuk registrasi.
        F.S. Mengembalikan true jika registrasi berhasil dan sesi
        diperbarui; false jika gagal.
            Jika gagal, pesan error yang sesuai ditampilkan.
    }

>>> Fungsi F03 - Logout
function logout(input/output session: pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Session.
    F.S. Mengembalikan true jika logout berhasil dan sesi direset; false
jika gagal.
    Jika gagal, pesan error yang sesuai ditampilkan.
}

>>> Fungsi F04 - Lupa Password
procedure forgotPassword(input/output UserList : users)
{memasukkan input password users baru jika ke dalam UserList}
{
    I.S Mengambil informasi username user
    F.S Mengembalikan informasi password baru
}
```

```
>>> Fungsi F05 - Menu Help
procedure displayHelp(input session: pointer to Session, input command:
String)
{ I.S. Menerima pointer ke struktur Session dan string perintah.
  F.S. Menampilkan deskripsi bantuan untuk perintah yang diminta atau
pesan error jika perintah tidak valid.
}
```

```
>>> Fungsi F06 - Denah Rumah Sakit
procedure displayLayout(input hospital: pointer to Hospital, input
session: pointer to Session, input printHeaderFlag: boolean)
{ I.S. Menerima pointer ke struktur Hospital, Session, dan sebuah flag
boolean.
  F.S. Menampilkan denah rumah sakit secara visual jika kondisi akses
terpenuhi.
}
```

```
>>> Fungsi F07 - Lihat User
procedure displayPatients(input hospital: pointer to Hospital, input
session: pointer to Session)
  { I.S. Menerima pointer ke struktur Hospital yang valid dan
  terinisialisasi, dan pointer ke struktur Session yang valid dan
  terinisialisasi.
    F.S. Menampilkan daftar semua pasien dalam sistem
  }

procedure displayUsers(input hospital : pointer to Hospital, input
session : pointer to Session, input viewType : integer)
  { I.S. Menerima pointer ke struktur Hospital yang valid dan
  terinisialisasi (berisi data pengguna dan pasien),pointer ke
  struktur Session yang valid dan terinisialisasi,dan sebuah integer
`viewType`
  F.S. Menampilkan daftar semua pengguna (Manajer, Dokter, Pasien,
  lainnya) dalam sistem
  }
```

```

procedure displayDoctors(input hospital: pointer to Hospital, input
session: pointer to Session)
    { I.S. Menerima pointer ke struktur Hospital dan Session.
      F.S. Menampilkan daftar semua dokter dalam sistem
          jika pengguna yang login adalah Manajer; jika tidak, pesan
          error ditampilkan.
    }

```

>>> Fungsi F08 - Cari User

```

procedure findUser(input hospital : pointer to Hospital, input session :
pointer to Session, input query : string, input byId : boolean)
    { I.S. Menerima pointer ke struktur Hospital dan Session, string
      query, dan boolean byId.
      F.S. Mencari dan menampilkan informasi pengguna berdasarkan ID
      atau username.
    }

```

>>> Fungsi F09 - Lihat Antrian

```

procedure displayQueue(input hospital: pointer to Hospital, input
session: pointer to Session)
    { I.S. Menerima pointer ke struktur Hospital dan Session.
      F.S. Menampilkan status antrian untuk setiap ruangan yang
      memiliki dokter
          dan pasien di dalamnya/antri, atau pesan error jika akses
          ditolak atau denah kosong.
    }

```

>>> Fungsi F10 - Tambah Dokter

```

function addDoctor(input hospital: pointer to Hospital, input session:
pointer to Session, input inputUsername: String, input password: String,
input specialization: String) → boolean
    { I.S. Menerima pointer ke struktur Hospital, Session, username,
      password, dan spesialisasi dokter.
      F.S. Mengembalikan true jika dokter berhasil ditambahkan; false
      jika gagal.
      Jika gagal, pesan error yang sesuai ditampilkan.
    }

```

```
}
```

>>> Fungsi F11 - Diagnosis

```
function diagnosePatient(input hospital: pointer to Hospital, input
session: pointer to Session) → boolean

{ I.S. Menerima pointer ke struktur Hospital dan Session.

  F.S. Mengembalikan true jika diagnosis berhasil dan status
pasien diperbarui; false jika gagal.

  Jika gagal, pesan error yang sesuai ditampilkan.

}
```

>>> Fungsi F12 - Ngobatin

```
function treatPatient(input hospital: pointer to Hospital, input
session: pointer to Session) → boolean

{ I.S. Menerima pointer ke struktur Hospital dan Session.

  F.S. Mengembalikan true jika pasien berhasil diberi resep obat
berdasarkan penyakit hasil diagnosa sebelumnya; false jika gagal.

  Jika gagal, pesan error yang sesuai ditampilkan.

}
```

>>> Fungsi F13 - Aku boleh pulang ga, dok?

```
function canGoHome(input hospital: pointer to Hospital, input session:
pointer to Session) → boolean

{ I.S. Menerima pointer ke struktur Hospital, Session.

  F.S. Mengembalikan true jika semua obat telah diminum dan dengan
urutan yang tepat; false jika tidak.

  Jika gagal, pesan error yang sesuai ditampilkan.

}
```

>>> Fungsi F14 - Daftar Check-Up

```
function registerCheckup(input hospital: pointer to Hospital, input
session: pointer to Session) → boolean

{ I.S. Menerima pointer ke struktur Hospital, Session.

  F.S. Mengembalikan true jika pasien berhasil diregistrasi ;
false jika tidak.

  Jika gagal, pesan error yang sesuai ditampilkan.

}
```

>>> Fungsi F15 - Antrian Saya!

```

procedure viewPatientQueue(input hospital: pointer to Hospital, input
session: pointer to Session)
    { I.S. Menerima pointer ke struktur Hospital dan Session.
      F.S. Menampilkan status antrian check-up dari pasien yang telah
      login.
      Jika pasien sudah di dalam ruangan, informasi ditampilkan.
      Jika gagal, pesan error ditampilkan.
    }

```

```

function takeMedication (hospital: pointer to Hospital, session: pointer
to Session) → boolean
    { I.S. hospital terdefinisi dan berisi data rumah sakit, pasien,
    obat-obatan, dan antrian.

      F.S. Jika berhasil, obat yang dipilih pasien ditambahkan ke
      daftar obat yang diminum (patient.medicationsTaken),
      dan fungsi mengembalikan true.
      Jika obat yang diminum salah, nyawa pasien berkurang. Jika
      nyawa habis, pasien dihapus dan sesi direset.

      Jika gagal (misalnya, struktur tidak valid, akses ditolak,
      pasien tidak ditemukan, belum diresepkan,
      tidak ada obat diresepkan, atau urutan obat salah), fungsi
      mengembalikan false dan menampilkan pesan error yang sesuai. }

```

>>> Fungsi F17 - Minum Penawar

```

procedure minumPenawar(input: Hospital hospital, Session session)
{ prosedur untuk memuntahkan obat terakhir yang diminum (top of
stack), dan mengembalikannya ke daftar obat yang diresepkan }

```

>>> Fungsi F18 - Exit

```

procedure exitProgram(input hospital: pointer to Hospital, input
session: pointer to Session)
    { I.S. Menerima pointer ke struktur Hospital dan Session.

```

F.S. Program diakhiri setelah membebaskan memori dan menampilkan pesan keluar.

}

>>> Fungsi B02 - Denah Dinamis

```
function changeLayout( hospital:  Hospital,  session:  Session,  newRowCount: ,  newColCount: ) →   
{ I.S. Menerima  ke struktur Hospital, Session, dan ukuran baris serta kolom denah yang baru.  
F.S. Mengembalikan  jika denah berhasil diubah;  jika gagal.  
Jika gagal, pesan error yang sesuai ditampilkan.  
}  
  
function moveDoctor( hospital:  Hospital,  session:  Session,  username: String,  newRoomCode: String) →   
{ I.S. Menerima  ke struktur Hospital, Session, username dokter, dan kode ruangan baru.  
F.S. Mengembalikan  jika dokter (dan pasiennya) berhasil dipindahkan ke ruangan baru;  jika gagal.  
Jika gagal, pesan error yang sesuai ditampilkan.  
}
```

>>> Fungsi B03 - Aura!

```
procedure displayDoctors( hospital:  Hospital,  session:  Session)  
{ I.S. Menerima  ke struktur Hospital dan Session.  
F.S. Menampilkan daftar semua dokter dalam sistem  
jika pengguna yang login adalah Manajer; jika tidak, pesan error ditampilkan.  
}  
}
```

>>> Fungsi B04 - Banarich!!!

```

function gacha(input hospital: pointer to Hospital, input session:
pointer to Session) → boolean

{ I.S. Menerima pointer ke struktur Hospital dan Session.

  F.S. Mengembalikan true jika gacha berhasil dilakukan dan hadiah
diberikan; false jika gagal.

  Jika gagal, pesan error yang sesuai ditampilkan.

}

procedure viewFinancial(input hospital: pointer to Hospital, input
session: pointer to Session)

{ I.S. Menerima pointer ke struktur Hospital dan Session.

  F.S. Menampilkan laporan finansial rumah sakit atau pesan error
jika akses ditolak.

}

procedure viewWallet(input hospital: pointer to Hospital, input session:
pointer to Session)

{ I.S. Menerima pointer ke struktur Hospital dan Session.

  F.S. Menampilkan saldo dompet pengguna (pasien atau dokter) atau
pesan error jika akses ditolak.

}

```

>>> Fungsi B05 - Dead or Alive!

```

procedure deletePatient (input/output hospital : pointer to Hospital,
input patientid : integer)

{ Menghapus pasien dari sistem }

```

>>> Fungsi B06 - Mainin Antrian

```

function skipPatientInQueue(input hospital: pointer to Hospital, input
session: pointer to Session, input roomCode: String) → boolean

{ I.S. Menerima pointer ke struktur Hospital, Session, dan kode ruangan.

```

F.S. Mengembalikan true jika pasien terdepan berhasil dipindahkan ke akhir antrian; false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}

```
>>> Fungsi B07 - Search Suggestion
procedure displaySubstring(input hospital : pointer to Hospital,
input name : string, input found : boolean)
{ I.S. Menerima pointer ke struktur Hospital, string name, dan
boolean found.

F.S. Daftar username yang mengandung substring dari name
ditampilkan.

}
```

## DESAIN KAMUS DATA

```
>>> Fungsi F01 - Login
passwordLength : integer
encryptedPassword : character
successMessage : character
i : integer
```

```
>>> Fungsi F02 - Register Pasien
passwordLength : integer
maxId : integer
newPatientId : integer
newUser : pointer to User
newPatient : pointer to Patient
successMessage : character
i : integer
```

```
>>> Fungsi F03 - Logout
tempUsername : character
successMessage : character
```

```
>>> Fungsi F04 - Lupa Password
userIdx : integer
```

```

passwordLength : integer
expectedRle : character
successMessage : character
i : integer

>>> Fungsi F5 - Menu & Help
errorMessage : character

>>> Fungsi F6 - Denah Rumah Sakit
i, j : integer

>>> Fungsi F7 - Lihat User
{ fungsi displayUsers}
    widths_users : array [1..4] of integer
    headers_users : array [1..4] of string
    user : pointer to User
    idStr_users : character
    roleStr : character
    diseaseStr : character
    row_users : array [1..4] of string
    i, j : integer
    id : integer
{ fungsi displayPatients }
    widths_patients : array [1..3] of integer
    headers_patients : array [1..3] of string
    patient : pointer to Patient
    idStr_patients : character
    diseaseStr_patients : character
    row_patients : array [1..3] of string
{ fungsi displayDoctors }
    widths_doctors : array [1..3] of integer
    headers_doctors : array [1..3] of string
    doctor : pointer to Doctor
    idStr_doctors : character
    auraStr : character
    row_doctors : array [1..3] of string
    aura : integer

>>> Fungsi F08 - Cari User
{ fungsi findUser}

```

```

overallFound : boolean
printedTableHeaders : boolean
widths : array [0..3] of integer
headers : array [0..3] of string
idStr : character
roleStr : character
diseaseStr : character
targetId : integer
keyUser : User
foundUser : pointer to User
user : pointer to User
exactMatchFound : boolean
suggestionsFound : boolean
row : array [0..3] of string
i, j : integer

{ fungsi findPatient}
    overallFound : boolean
    printedTableHeaders : boolean
    widths : array [0..2] of integer
    headers : array [0..2] of string
    idStr : character
    diseaseStr : character
    exactMatchFound : boolean
    suggestionsFound : boolean
    patient : pointer to Patient
    targetId : integer
    keyPatient : Patient
    foundPatient : pointer to Patient
    searchTypeStr : character
    notFoundMsg : character

```

```

row : array [0..2] of string
i, j : integer
{ fungsi findDoctor }
  found : boolean
  widths : array [0..2] of integer
  headers : array [0..2] of string
  targetId : integer
  keyDoctor : Doctor
  foundDoctor : pointer to Doctor
  idStr : character
  auraStr : character
  row : array [0..2] of string
  tempIdConv_doc : character
  k_id_doc : integer
  tempVal_id_doc : integer { long long }
  isNeg_id_doc : boolean
  numDigits_id_doc : integer
  valCopy_id_doc : integer { long long }
  floatValue_aura : real
  precision_aura : integer
  tempFloatStr_aura : character
  intPart_aura : integer { long long }
  intStr_aura : character
  tempIntConv_aura : character
  k_int_aura : integer
  tempVal_int_aura : integer { long long }
  numDigits_int_aura : integer
  valCopy_int_aura : integer { long long }
  fracPart_aura : real
  p_aura : integer

```

```

digit_aura : integer
digitChar_aura : character
doctor : pointer to Doctor
tempIdConv_doc_name : character
k_id_doc_name : integer
tempVal_id_doc_name : integer { long long }
isNeg_id_doc_name : boolean
numDigits_id_doc_name : integer
valCopy_id_doc_name : integer { long long }
floatValue_aura_name : real
precision_aura_name : integer
tempFloatStr_aura_name : character
intPart_aura_name : integer { long long }
intStr_aura_name : character
tempIntConv_aura_name : character
k_int_aura_name : integer
tempVal_int_aura_name : integer { long long }
numDigits_int_aura_name : integer
valCopy_int_aura_name : integer { long long }
fracPart_aura_name : real
p_aura_name : integer
digit_aura_name : integer
digitChar_aura_name : character
i, m : integer

```

```

>>> Fungsi F09 - Lihat Antrian
hasRoomsWithDoctor : boolean
i, j, k, l, m : integer
room : pointer to Room
header : character [100]
capacityStr : character [20]

```

```
capacity : integer
row1_capacity : array [1..2] of string
widths_capacity : array [1..2] of integer
doctorName : string
row1_doctor : array [1..2] of string
widths_doctor : array [1..2] of integer
found : boolean
foundQueue : boolean
currentQueue : pointer to Queue
currentNode : pointer to QueueNode
position : integer
patientFoundInList : boolean
```

```
>> Fungsi F10 - Tambah Dokter
newCapacity : integer
tempUsers : pointer to User
newDoctorCapacity : integer
tempDoctors : pointer to Doctor
maxId : integer
newDoctorId : integer
newUser : pointer to User
newDoctor : pointer to Doctor
successMsg : character
i : integer
```

```
>> Fungsi F11 - Diagnosis
doctorIdx : integer
doctor : pointer to Doctor
doctorRoom : pointer to Room
patientId : integer
patientIdx : integer
patient : pointer to Patient
diseaseStr : character [50]
d : pointer to Disease
```

```
i, j : integer
```

```
>>> Fungsi F12 - Ngobatin
    doctorIdx : integer
    doctor : pointer to Doctor
    doctorRoom : pointer to Room
    patientId : integer
    patientIdx : integer
    patient : pointer to Patient
    diseaseId : integer
    medicationCount : integer
    tempPrescriptions : pointer to MedicationPrescription
    prescribedMedications : pointer to MedicationList
    tempCount : integer
    med : pointer to Medication
    widths : array [0..3] of integer
    headers : array [0..3] of string
    idStr : character [5]
    numberStr : character [10]
    errorMsg : character [50]
    i, j : integer
```

```
>>> Fungsi F13 - Aku Boleh Pulang Ga, Dok?
    patient : pointer to Patient
    statusStr : string
    descriptionStr : string
    canGoHomeStatus : boolean
    prescribedCount : integer
    takenCount : integer
    correctOrder : boolean
    widths : array [1..2] of integer
```

```
headers : array [1..2] of string
row : array [1..2] of string
doctorId : integer
targetQueue : pointer to Queue
current : pointer to QueueNode
prev : pointer to QueueNode
i, j : integer
```

```
>>> Fungsi F14 - Daftar Check-Up
patientIdx : integer
patient : pointer to Patient
availableDoctorCount : integer
availableDoctors : array [1..N] of pointer to Doctor
doctor : pointer to Doctor
queueCount : integer
numberStr : character [10]
queueCountStr : character [10]
auraStr : character [10]
checkupCostStr : character [20]
row_displayDoctors : array [1..7] of string
widths_displayDoctors : array [1..7] of integer
headers_displayDoctors : array [1..7] of string
doctorChoice : integer
promptSelectDoctor : character [100]
availableCountStr : character [10]
queueIdx : integer
newQueue : pointer to Queue
queueToCheck : pointer to Queue
selectedRoom : pointer to Room
newElements : pointer to TreatmentHistory
successMsg : character [100]
```

```
history : pointer to TreatmentHistory
widths2 : array [1..2] of integer
headers2 : array [1..2] of string
row2 : array [1..2] of string
i, j : integer
```

```
>>> Fungsi F15 - Antrian Saya!
patient : pointer to Patient
i, r, c : integer
roomFound : boolean
patientInRoom : boolean
room : pointer to Room
doctor : pointer to Doctor
patientFoundInQueue : boolean
queueIdx : integer
queueToCheck : pointer to Queue
currentNode : pointer to QueueNode
queuePositionStr : character [5]
queueSizeStr : character [5]
finalQueueFormat : character [10]
widths : array [1..2] of integer
row1 : array [1..2] of string
row2 : array [1..2] of string
row3 : array [1..2] of string
```

```
>>> Fungsi F16 - Minum Obat
p : pointer to Patient
obatList : array [0..max_dl-1] of medicationOrder
obatCount : integer
choice : integer
```

```

selectedMedicationID : integer
diseaseID : integer
foundIndex : integer
medName : string

>>> Fungsi F17 - Minum Penawar
    p : pointer to Patient
    poppedMedID : integer
    diseaseID : integer
    doseOrder : integer
    insertPos : integer
    medName : string
    i, j : integer

>>> Fungsi F18 - Exit
    { Tidak ada kamus data yang digunakan }

>>> Fungsi B02 - Denah Dinamis
    i, j, k : integer
    room : pointer to Room
    errorMessage : string
    newLayout : array of array of Room { Menggambarkan Room** }
    code : string
    numberStr : string
    doctorToMove : pointer to Doctor
    targetRow : integer
    targetCol : integer
    targetRoom : pointer to Room
    currentRoom : pointer to Room
    patientId : integer
    patientToMove : pointer to Patient
    queue : pointer to Queue

```

```
>>> Fungsi B03 - Aura!
    widths_doctors : array [1..3] of integer
    headers_doctors : array [1..3] of string
    doctor : pointer to Doctor
    idStr_doctors : character
    auraStr : character
    row_doctors : array [1..3] of string
    aura : integer
```

```
>>> Fungsi B04 - Banarich!!!
    patientIndex : integer
    patient : pointer to Patient
    seed : unsigned long
    rand1 : unsigned long
    rand2 : unsigned long
    rand3 : unsigned long
    reward : real
    rewardMessage : string
    rewardType : integer
    itemChoice : integer
    smallRewards : array [1..3] of integer
    mediumRewards : array [1..3] of integer
    largeRewards : array [1..2] of integer
    rewardStr : string
    beforeStr : string
    afterStr : string
    beforeBalance : real
    afterBalance : real
    balance : real
    balanceStr : string
```

```
widths : array [1..2] of integer
headers : array [1..2] of string
row : array [1..2] of string
userType : string
i : integer
```

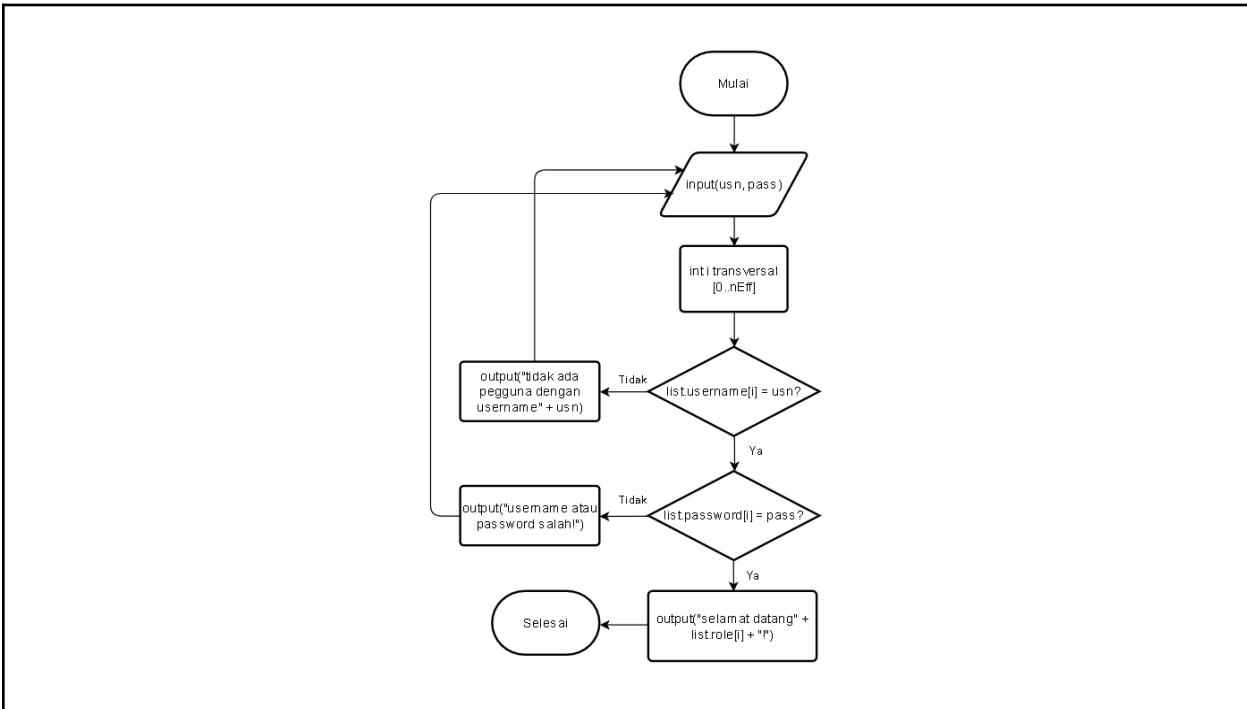
```
>>> Fungsi B05 - Dead or Alive?!
{ Tidak ada kamus data yang digunakan }
```

```
>>> Fungsi B06 - Mainin Antrian
q : pointer to Queue
err : string
patientIdToSkip : integer
successMsg : string
```

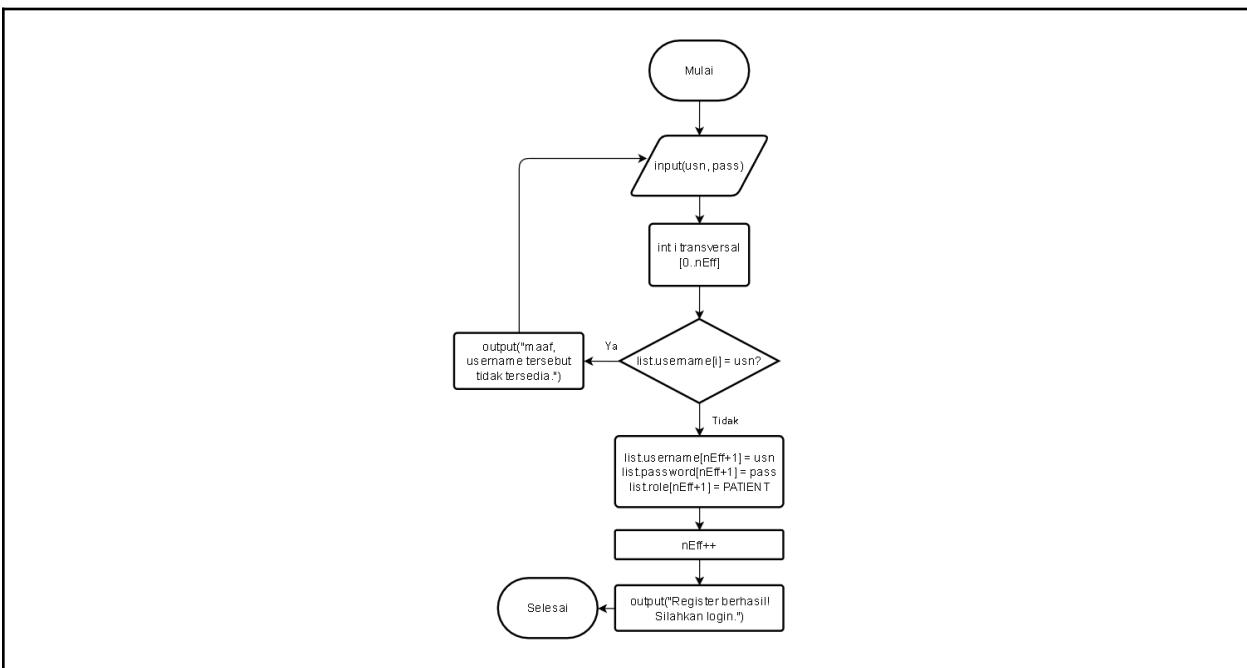
```
>>> Fungsi B07 - Search Suggestion
i, m, k, widths : integer
roleStr, lowerName, lowerUsername : string
headers : array [0..2] of string
```

## FLOWCHART PROGRAM

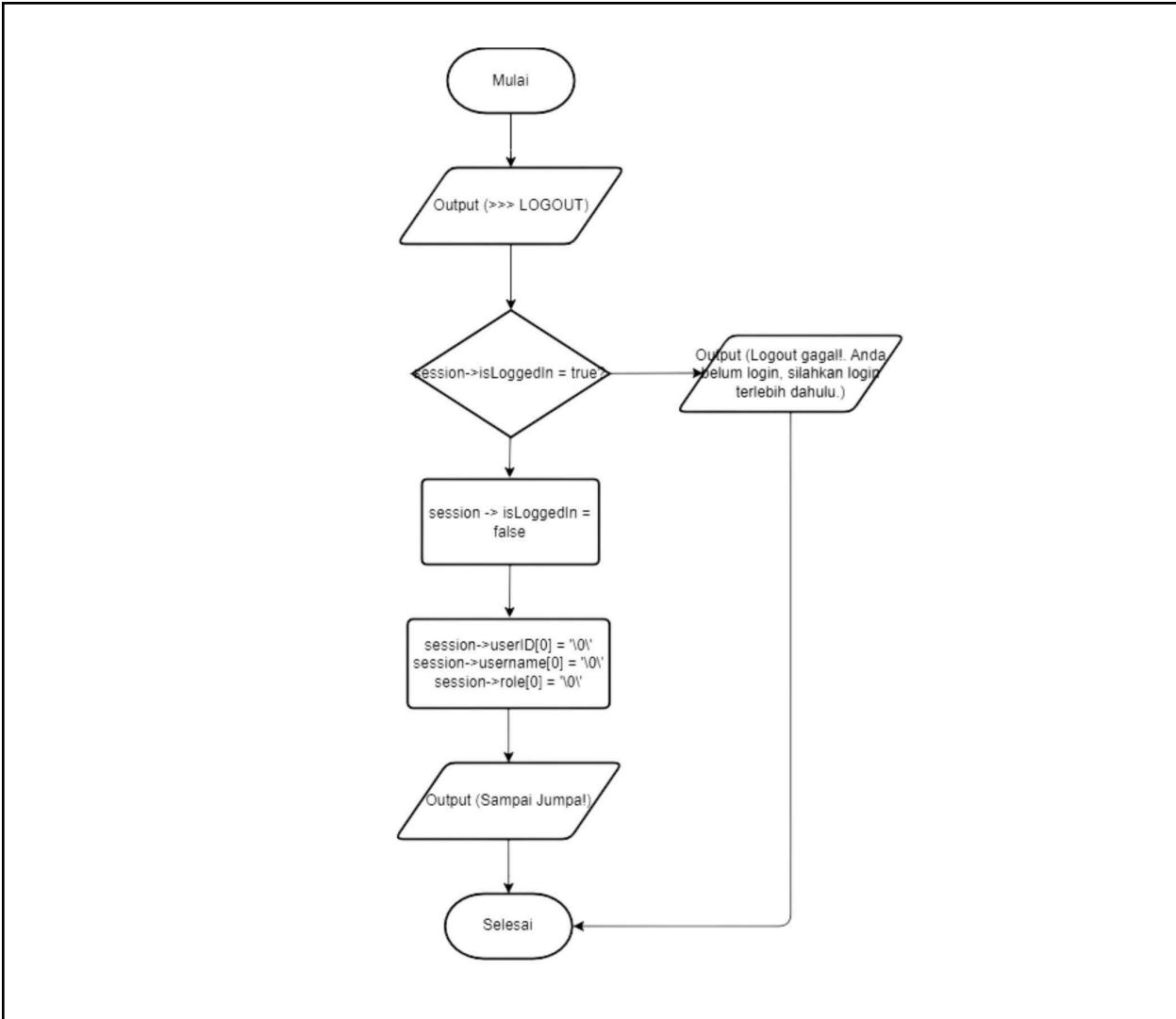
F01 - Login



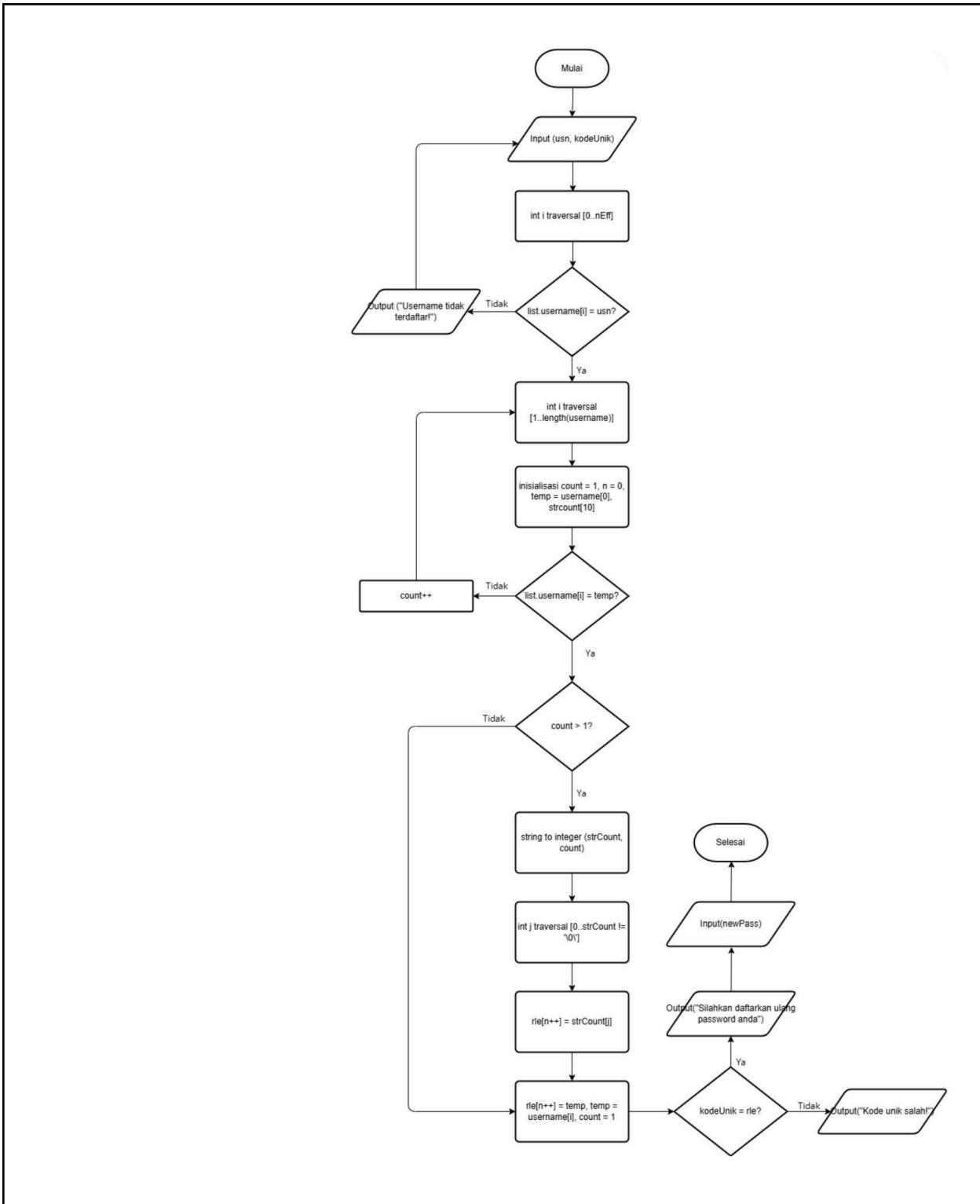
F02 - Register



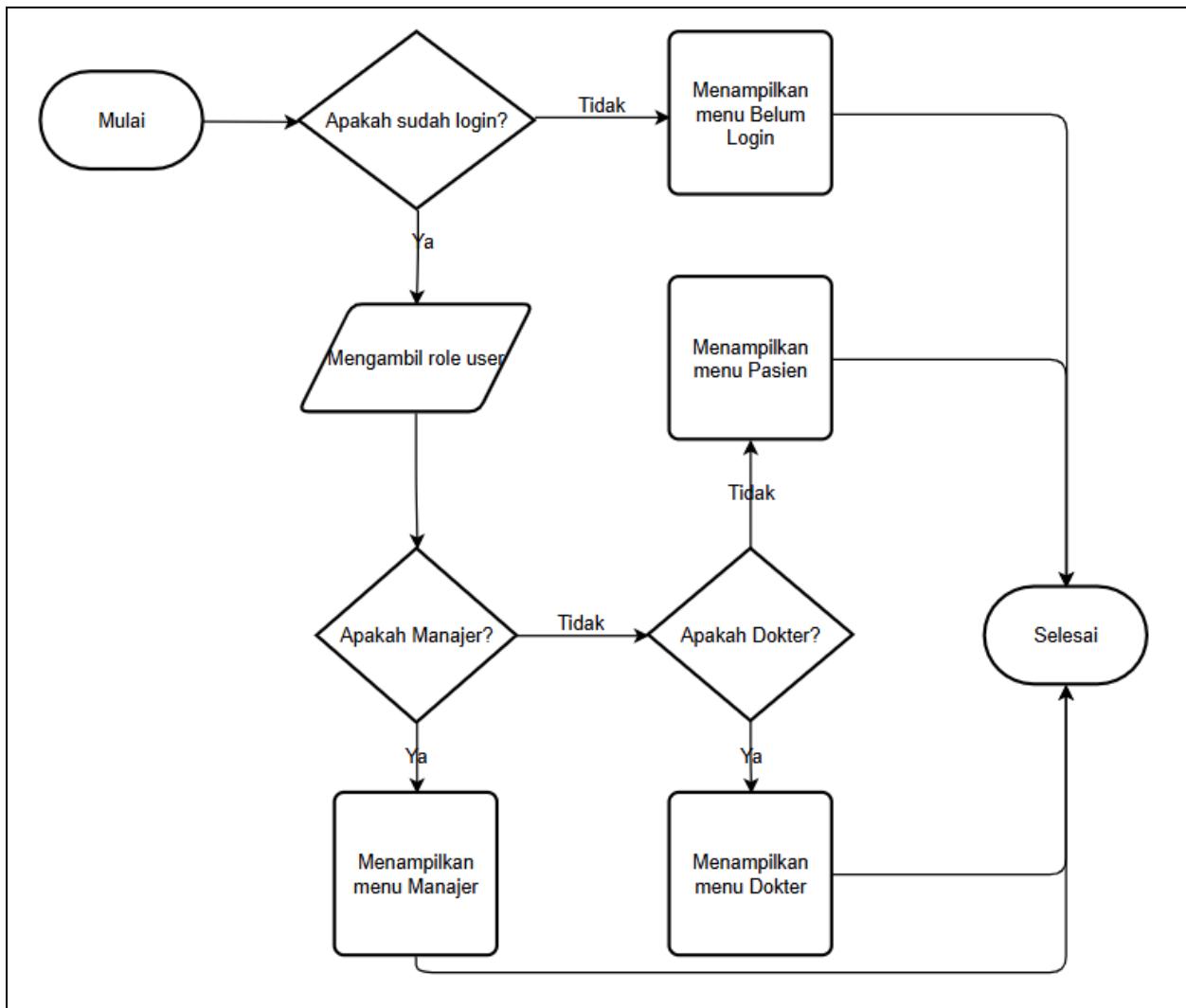
## F03 - Logout



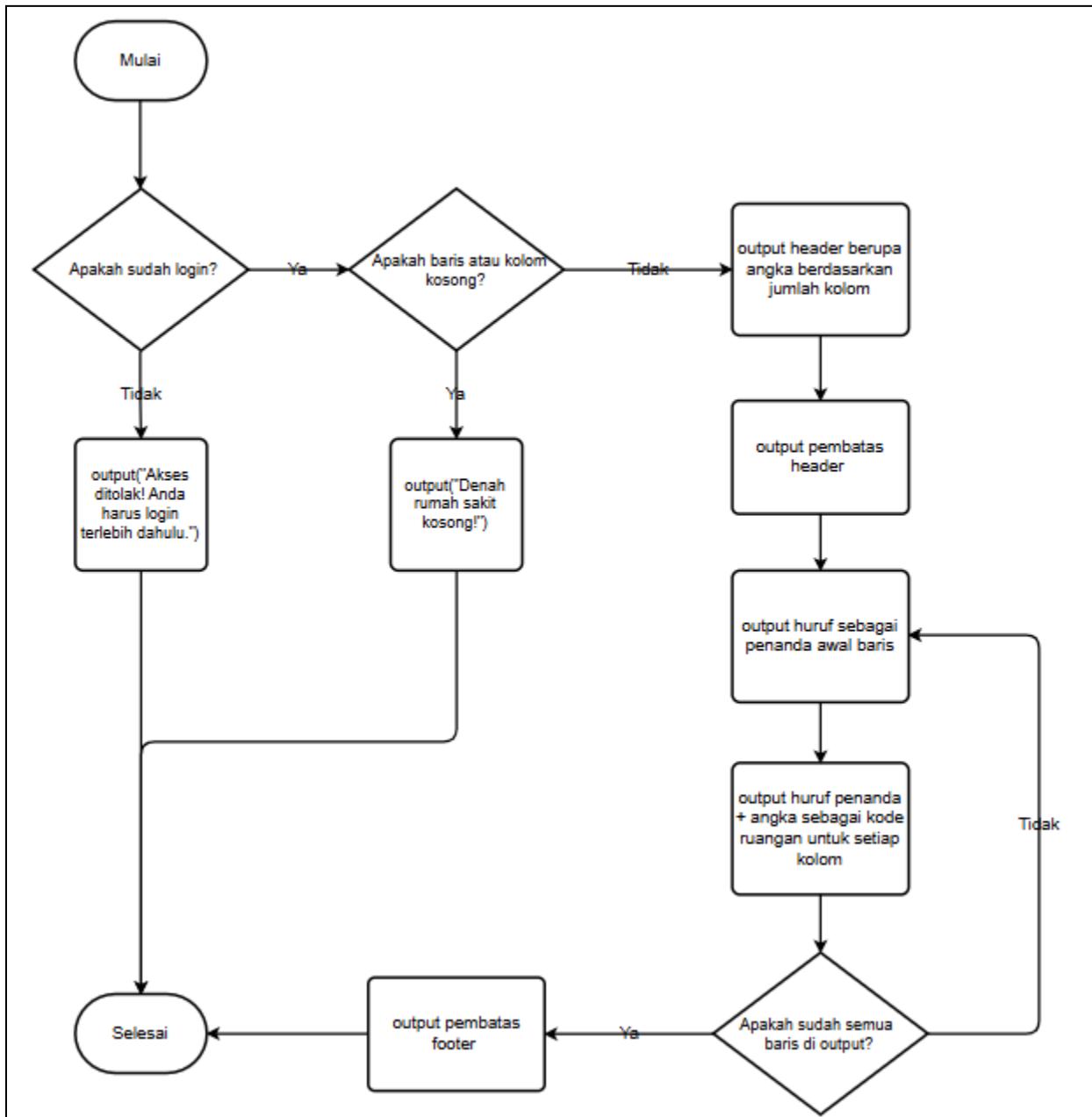
## F04 - Lupa Password



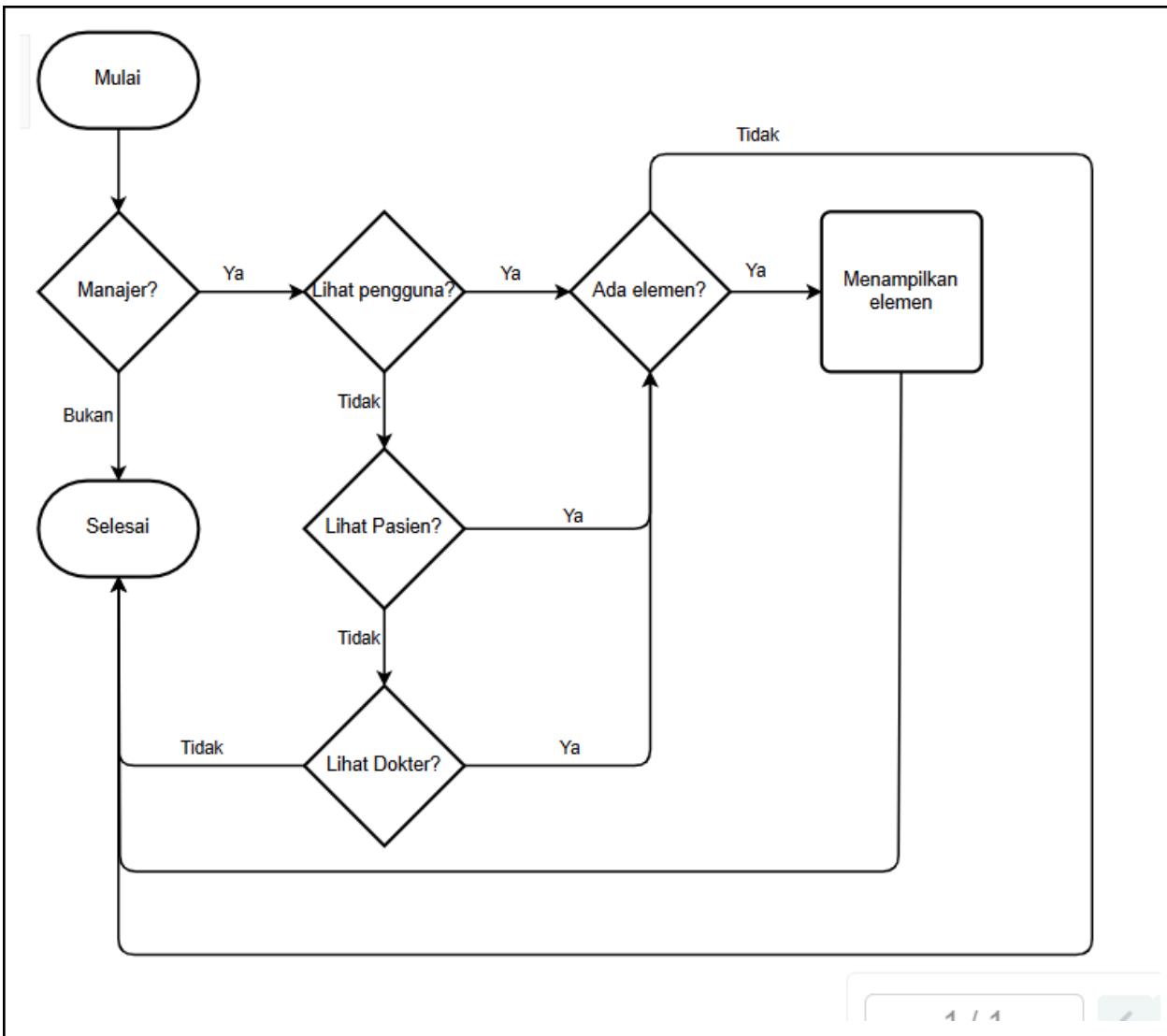
## F05 - Menu & Help



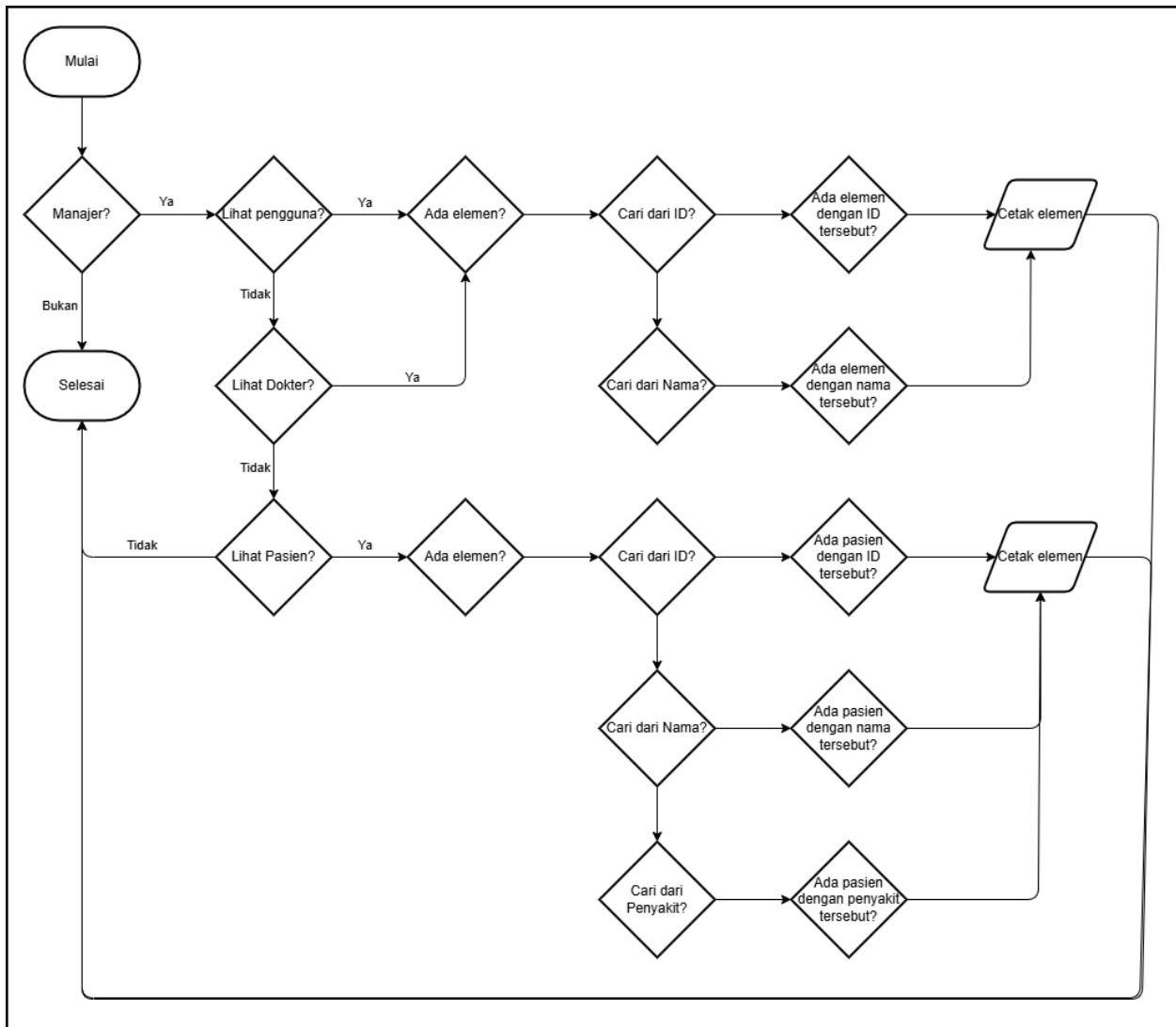
## F06 - Denah Rumah Sakit



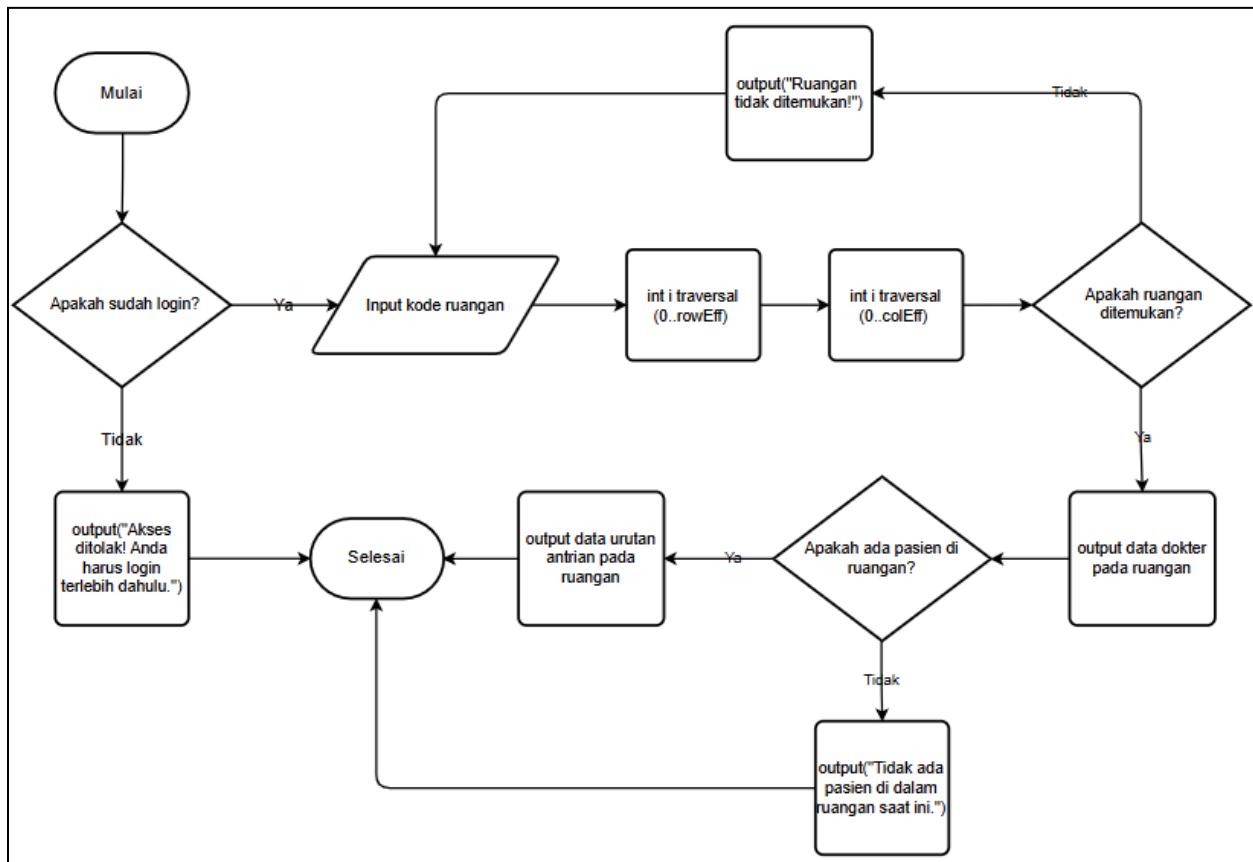
## F07 - Lihat User



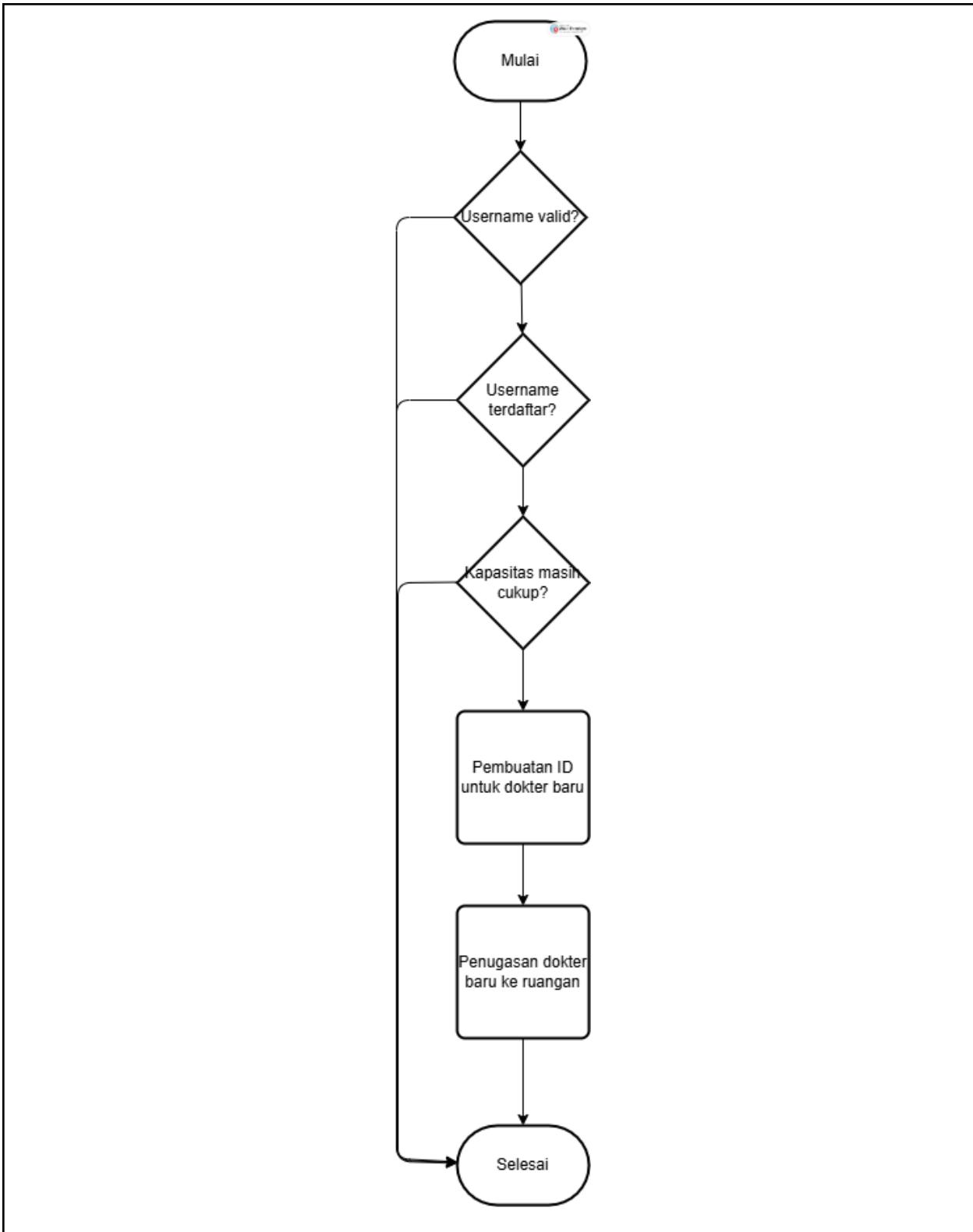
## F08 - Cari User



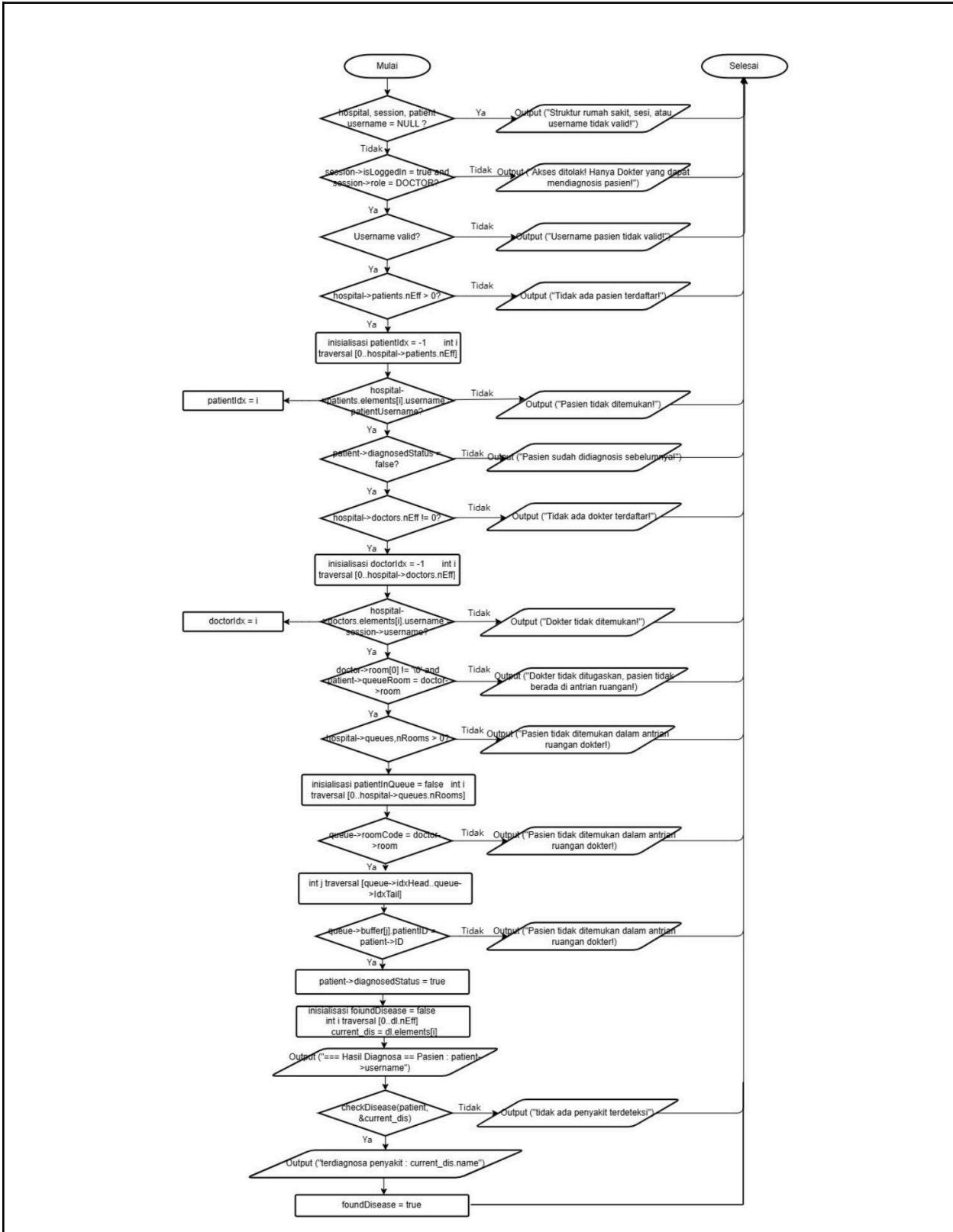
## F09 - Lihat Antrian



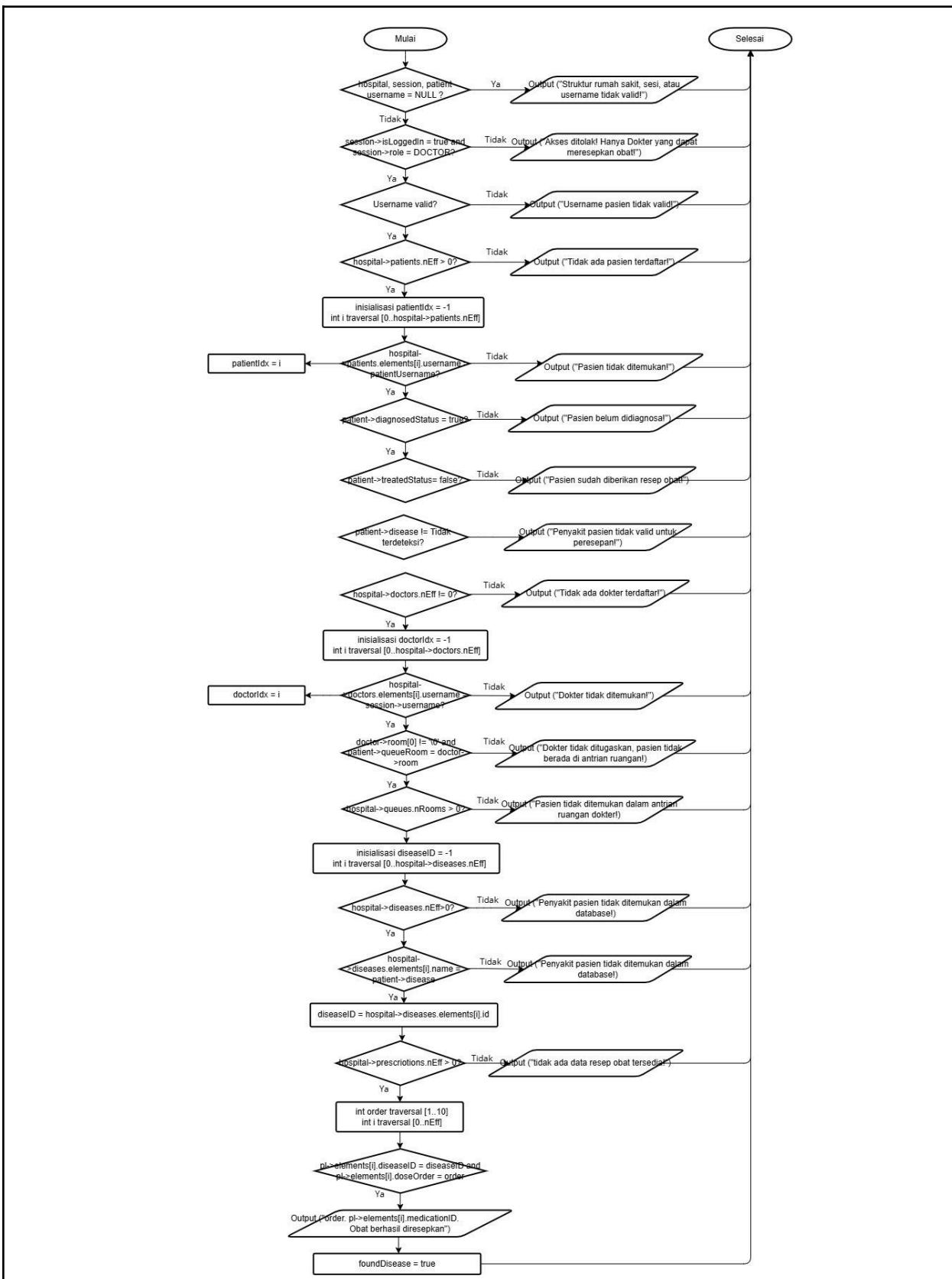
## F10 - Tambah Dokter



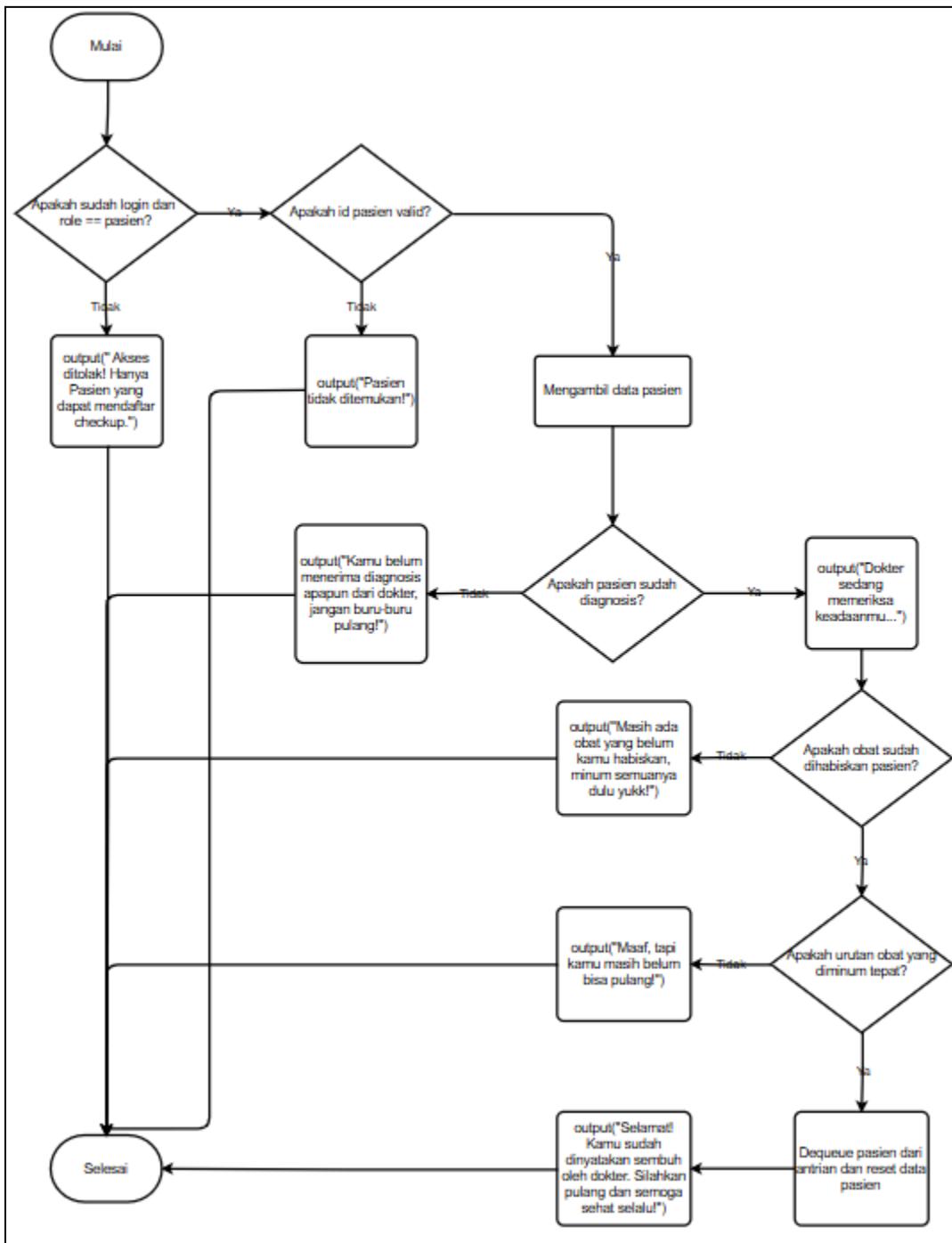
## F11 - Diagnosis



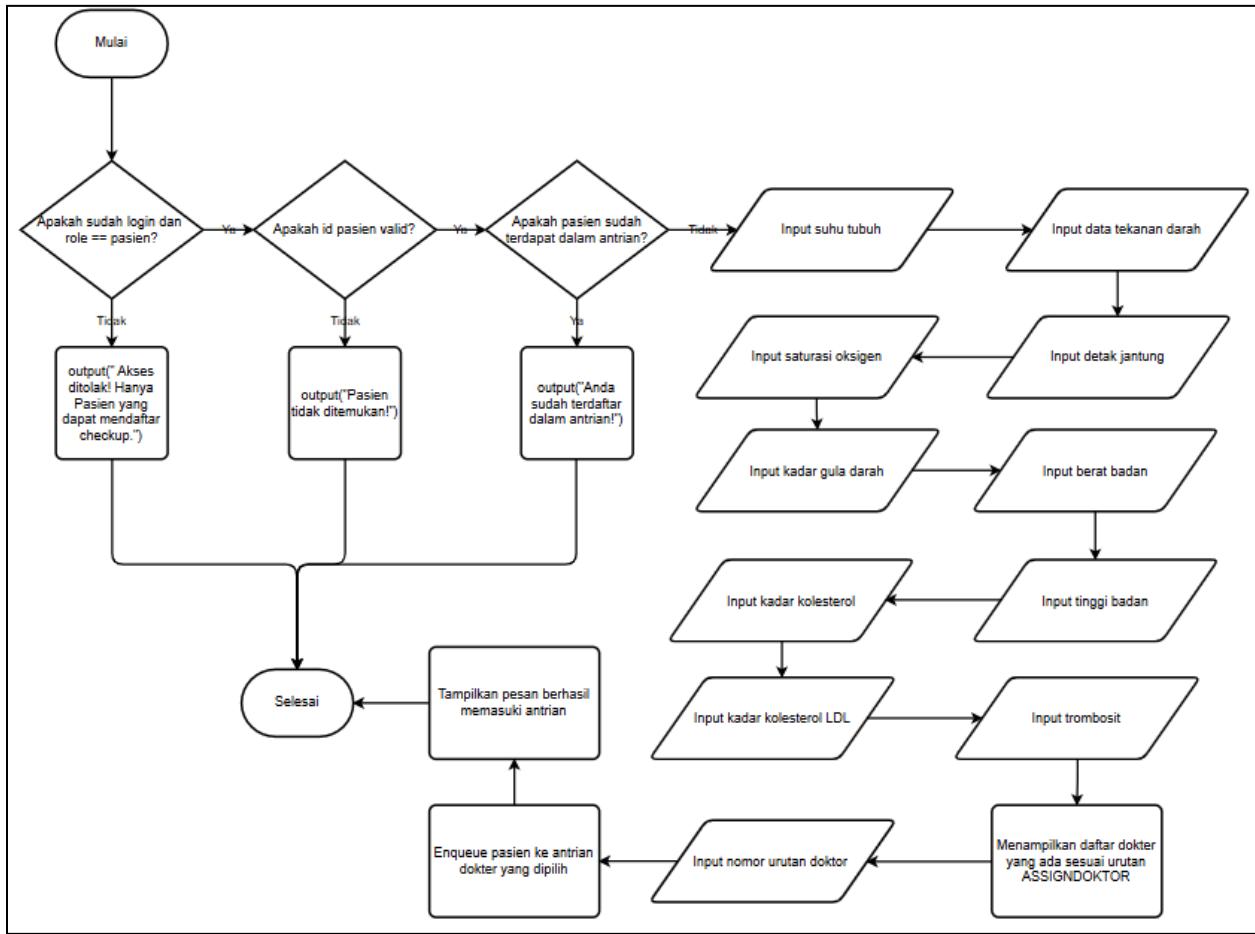
## F12 - Ngobatin



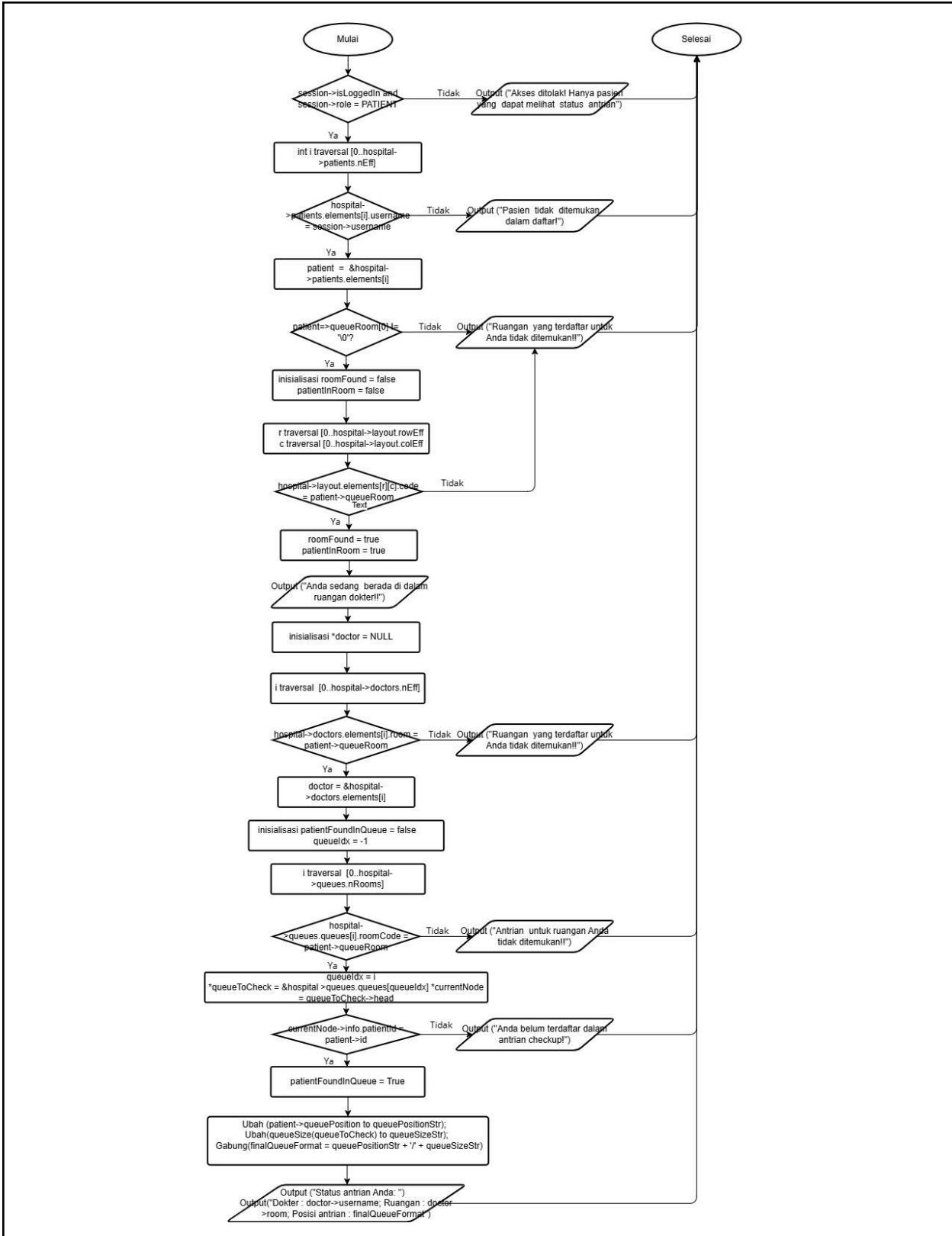
### F13 - Aku boleh pulang, dok?



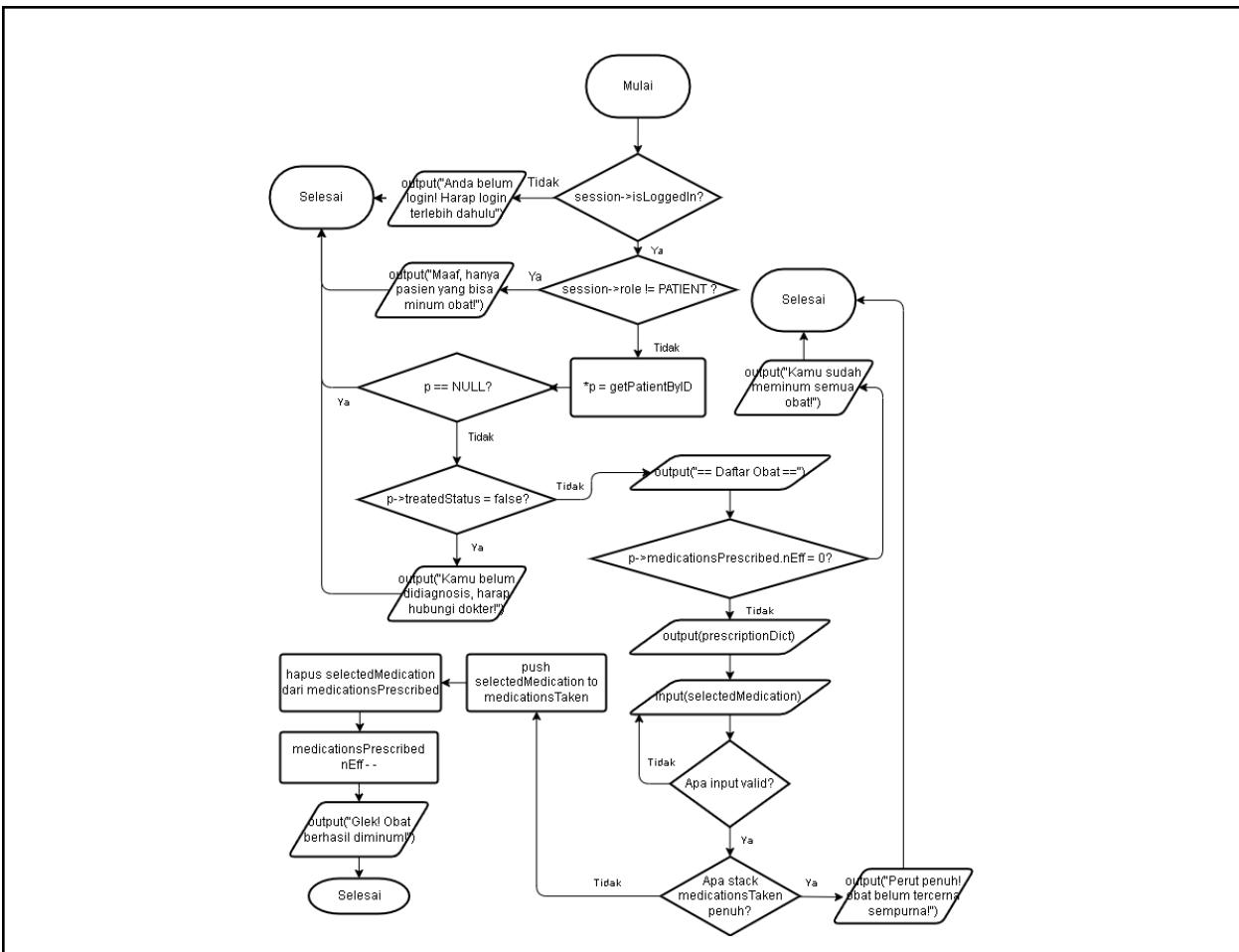
## F14 - Daftar Check-Up



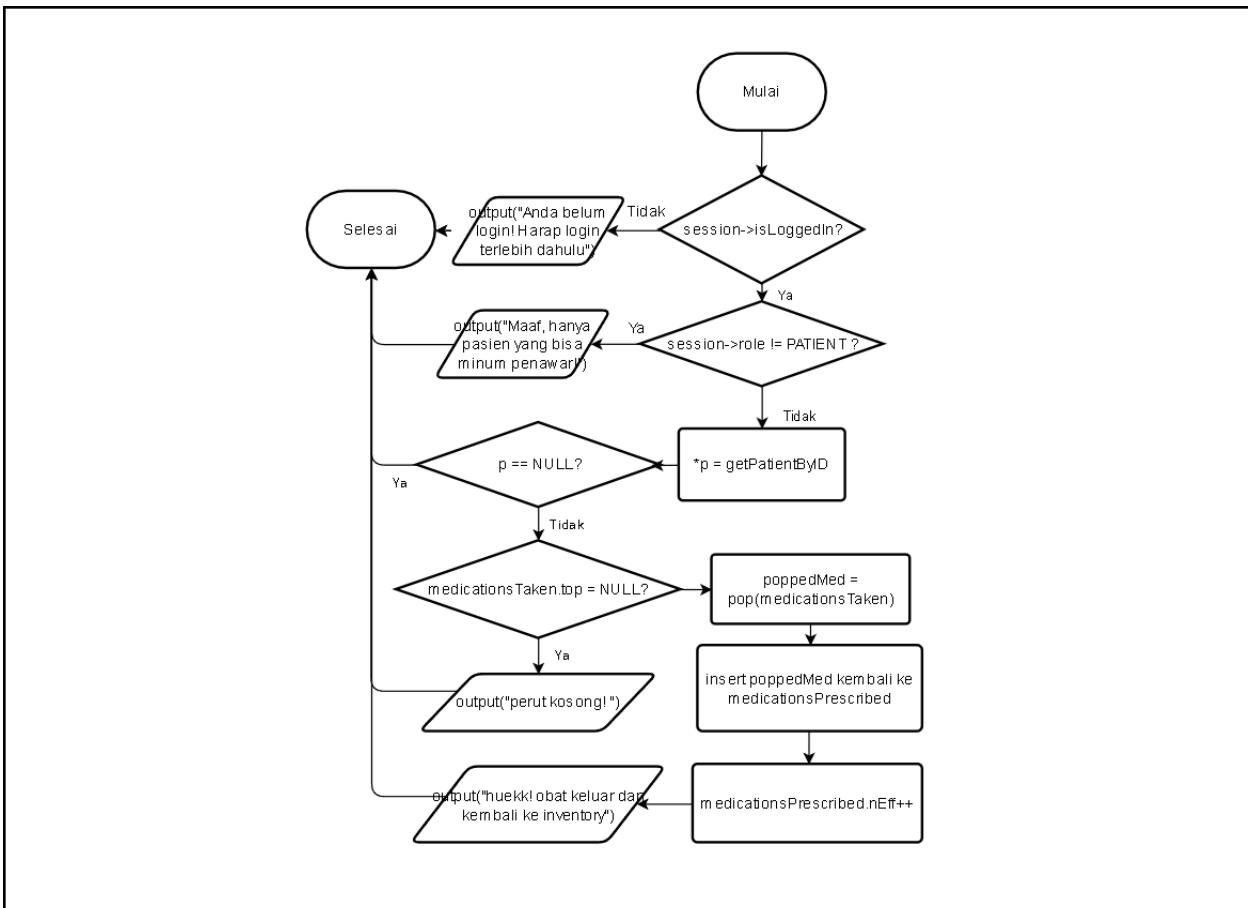
## F15 - Antrian Saya



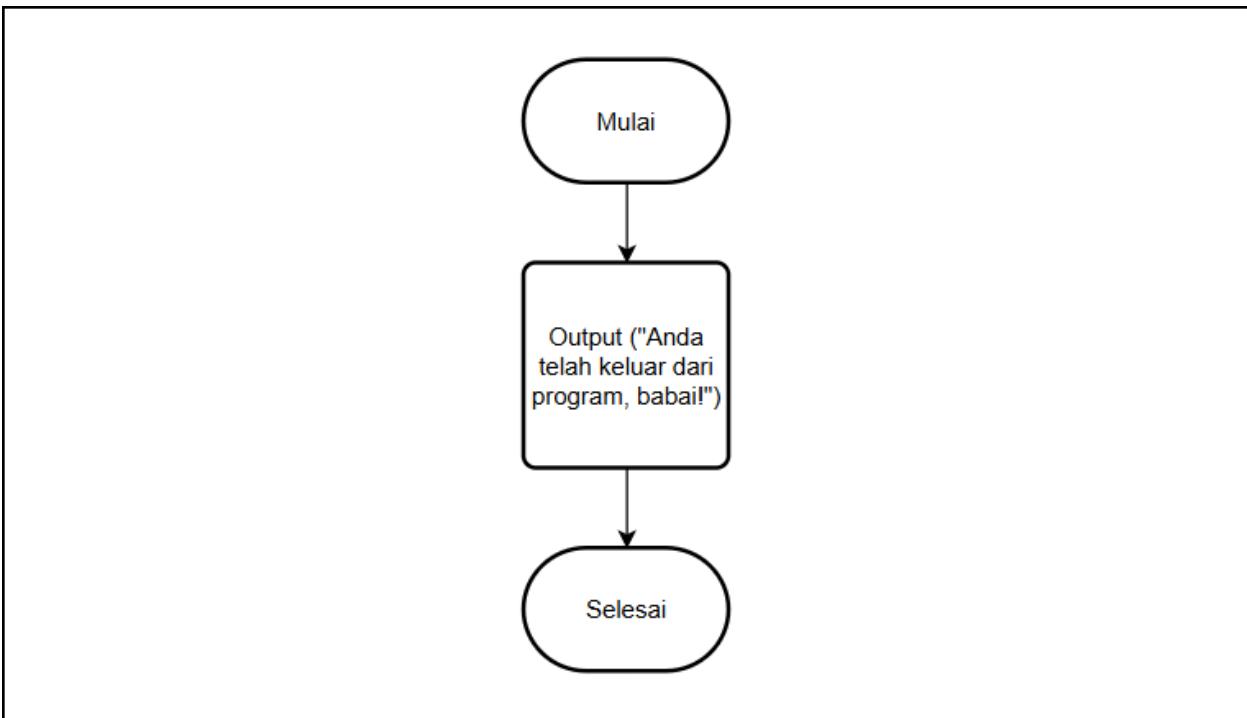
## F16 - Minum Obat



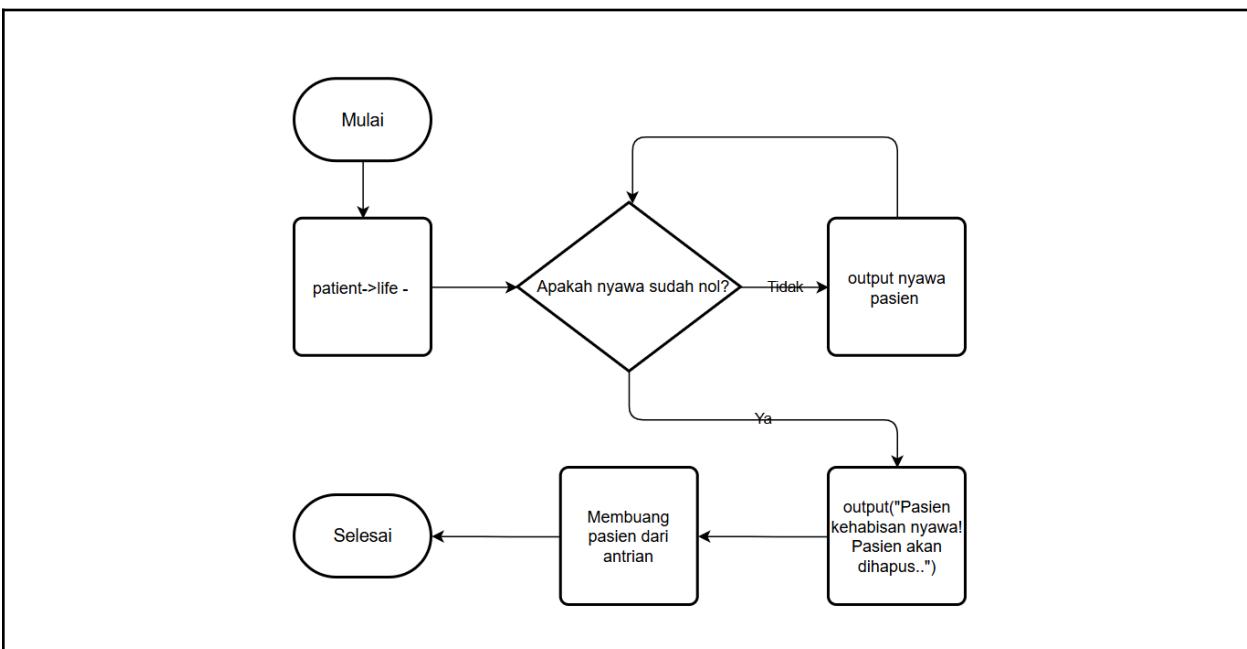
## F17 - Minum Penawar



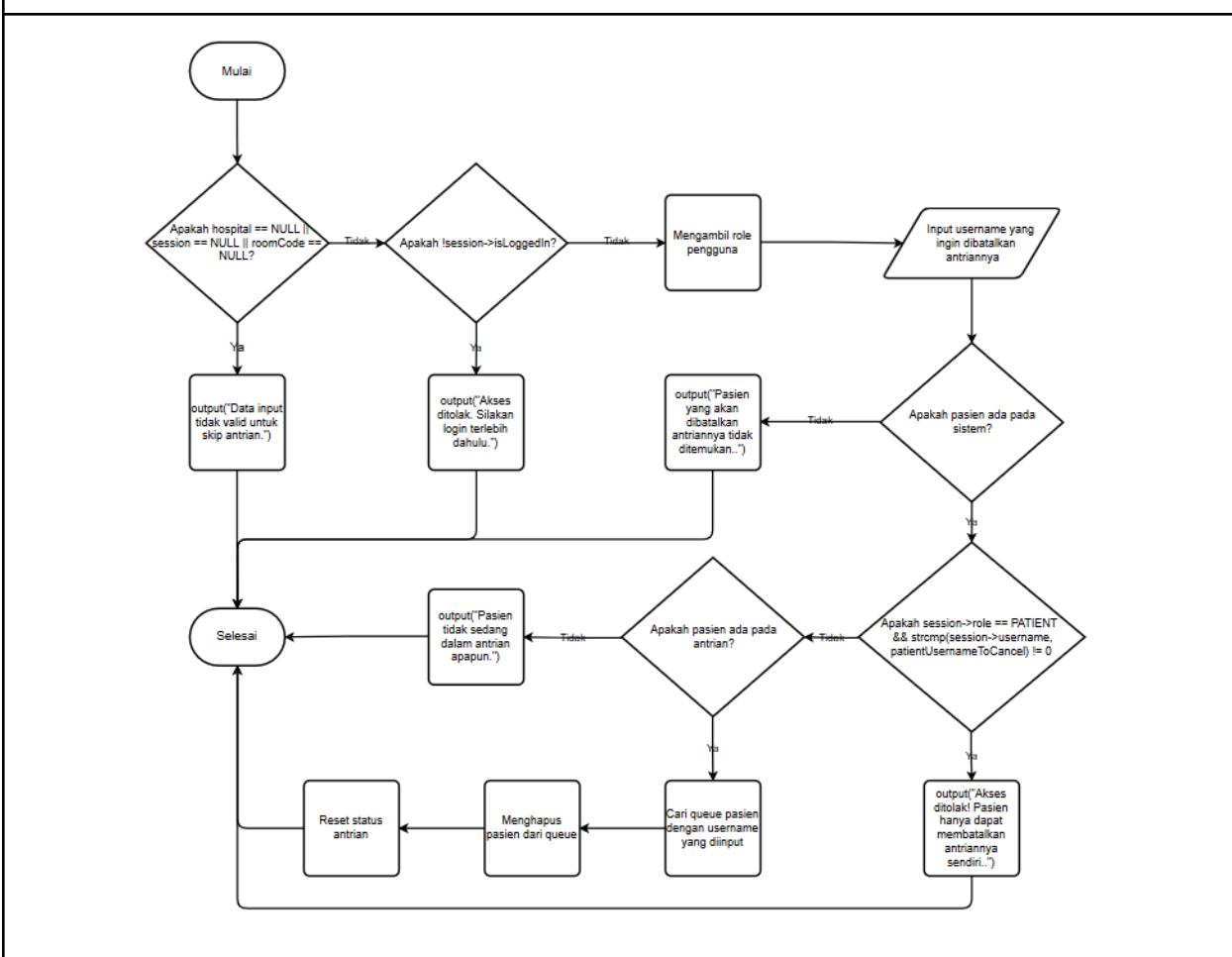
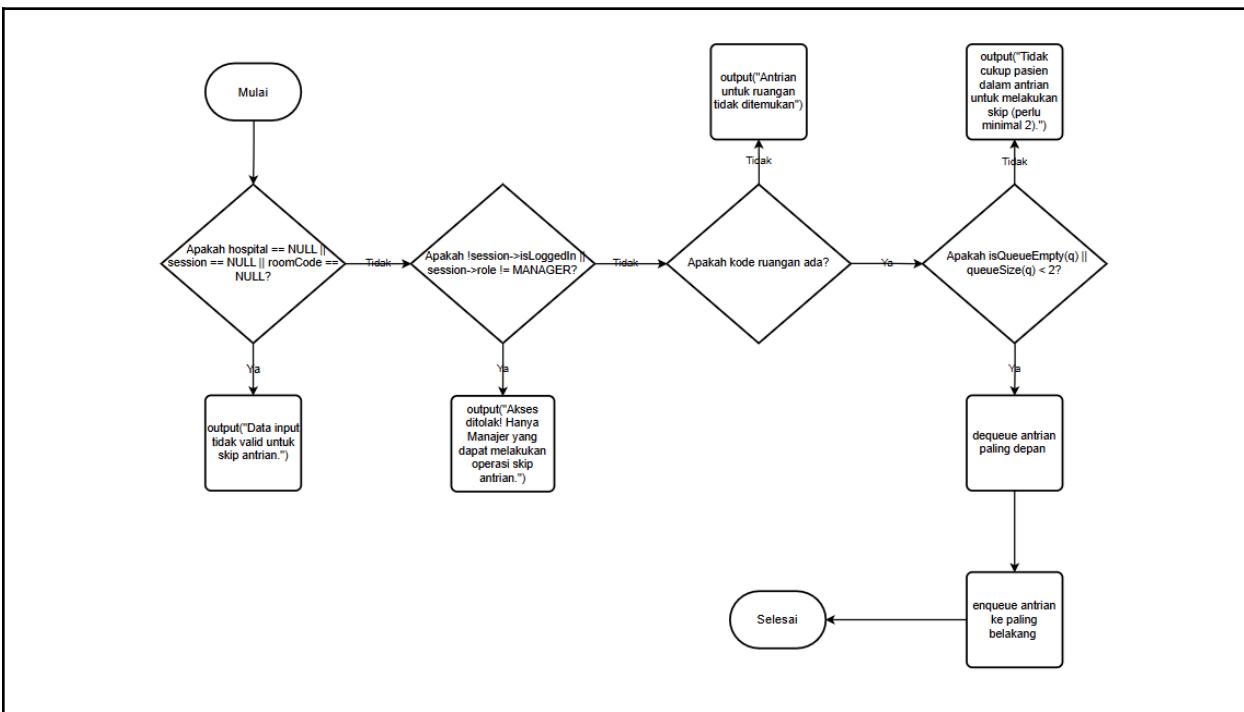
F18 - Exit



B05 - Dead or alive



## B06 - Mainin Antrian



## SPESIFIKASI PROGRAM

F01 - Login

```
{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [true, false] [cite: 1]

type Hospital :
< users : UserList >

type User :
< id : integer;
  username : string;
  password : PasswordData;
  role : Role >

type PasswordData :
< encryptedContent : string >

type UserList :
< elements : array [1..N] of User;
  nEff : integer;
  capacity : integer >

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

{ Definisi function & procedure yang di-import dari file source code lain }

function strlen(input s : string) → integer
{ Mencari panjang string s }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }
```

```

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function isValidUsername(input username : string) → boolean
{ Memvalidasi format username. }

function enigmaEncrypt(input password : string, output encryptedPassword
: string, input bufferSize : integer) → boolean
{ Mengenkripsi password. Mengembalikan true jika berhasil. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function login (input hospital: pointer to Hospital, input/output
session: pointer to Session, input username: String, input password:
String) → boolean
{ I.S. Menerima pointer ke struktur Hospital dan Session, serta
username dan password sebagai String.
    F.S. Mengembalikan true jika login berhasil, false jika gagal.
        Jika berhasil, session diperbarui dengan userId, username,
        role, dan isLoggedIn menjadi true.
        Jika gagal, pesan error yang sesuai ditampilkan.
}

```

#### KAMUS LOKAL

```

passwordLength : integer
encryptedPassword : character
successMessage : character
i : integer

```

#### ALGORITMA

```

depend on (true) :
(hospital = NIL or session = NIL) :
output("Struktur rumah sakit atau sesi tidak valid!")
→ false

```

```

(username = NIL or username[0] = '\0' or password = NIL or
password[0] = '\0') :
    output("Username atau password tidak valid!")
    → false
(not isValidUsername(username)) :
    output("Username tidak valid!")
    → false
(strlen(password) < 6) :
    output("Password harus minimal 6 karakter!")
    → false
(session.isLoggedIn) :
    output("Anda sudah login! Silakan logout terlebih
dahulu.")
    → false
(hospital.users.nEff ≤ 0) :
    output("Tidak ada pengguna terdaftar!")
    → false
(otherwise) :
    i traversal [0...hospital.users.nEff-1]
        if (strcmp(hospital.users.elements[i].username,
username) = 0) then
            passwordLength ← strlen(password)
            depend on (true) :
                (passwordLength ≥ 100) :
                    output("Panjang password terlalu besar
untuk dienkripsi!")
                    → false
                (not enigmaEncrypt(password,
encryptedPassword, 100)) :
                    output("Gagal mengenkripsi password!")
                    → false

(strcmp(hospital.users.elements[i].password.encryptedContent,
encryptedPassword) = 0) :
            session.userId ←
hospital.users.elements[i].id
            strcpy(session.username, username)
            session.role ←
hospital.users.elements[i].role

```

```

session.isLoggedIn ← true

strcpy(successMessage, "Login berhasil
sebagai ")
strcat(successMessage, username)
strcat(successMessage, "!")
strcat(successMessage, " (Peran: ")

depend on (session.role) :
    (MANAGER) :
        strcat(successMessage, "Manajer")
    (DOCTOR) :
        strcat(successMessage, "Dokter")
    (OTHERS) :
        strcat(successMessage, "Pasien")
strcat(successMessage, ")")
output(successMessage)
→ true
(otherwise) :
    output("Password salah!")
    → false

output("Username tidak ditemukan!")
→ false

```

## F02 - Register Pasien

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
    username : string;
    role : Role;
    isLoggedIn : boolean >

type Hospital :
< users : UserList;

```

```

patients : PatientList >

type UserList :
< elements : array [1..N] of User;
  nEff : integer;
  capacity : integer >

type User :
< id : integer;
  username : string;
  password : PasswordData;
  role : Role >

type PasswordData :
< encryptedContent : string >

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer;
  capacity : integer >

type Patient :
< id : integer;
  username : string;
  bananaRich : real;
  life : integer;
  diagnosedStatus : boolean;
  treatedStatus : boolean;
  medicationsPrescribed : MedicationList;
  medicationsTaken : MedicationStack;
  queueRoom : string;
  queuePosition : integer >

type MedicationList :
< medicationId : pointer to integer;
  nEff : integer;
  capacity : integer >

type MedicationStack :

```

```

< medicationId : pointer to integer;
  capacity : integer;
  top : integer >

{ Definisi function & procedure yang di-import dari file source code
lain }

function strlen(input s : string) → integer
{ Mencari panjang string s }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function enigmaEncrypt(input password : string, output encryptedPassword
: string, input bufferSize : integer) → boolean
{ Mengenkripsi password. Mengembalikan true jika berhasil. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function isValidUsername(input username : string) → boolean
{ Memvalidasi format username. }

function isUsernameTaken(input hospital : pointer to Hospital, input
username : string) → boolean
{ Memeriksa apakah username sudah terdaftar. }

function safeMalloc(input size : integer) → pointer to void
{ Mengalokasikan memori dengan aman. }

{ Prosedur Utama }

function registerPatient(input hospital: pointer to Hospital,
input/output session: pointer to Session, input inputUsername: String,
input password: String) → boolean

```

```

{ I.S. Menerima pointer ke struktur Hospital dan Session, serta
username dan password untuk registrasi.

F.S. Mengembalikan true jika registrasi berhasil dan sesi
diperbarui; false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}

```

#### KAMUS LOKAL

```

passwordLength : integer
maxId : integer
newPatientId : integer
newUser : pointer to User
newPatient : pointer to Patient
successMessage : character
i : integer

```

#### ALGORITMA

```

depend on (true) :
    (hospital = NIL or session = NIL) :
        printError("Struktur rumah sakit atau sesi tidak valid!")
        → false
    (inputUsername = NIL or inputUsername[0] = '\0' or password =
    NIL or password[0] = '\0') :
        printError("Username (setelah trim) atau password tidak
valid!")
        → false
    (session.isLoggedIn) :
        printError("Anda sudah login! Silakan logout terlebih
dahulu.")
        → false
    (not isValidUsername(inputUsername)) :
        printError("Username tidak valid! Hanya boleh berisi
huruf, angka, spasi, atau underscore.")
        → false
    (isUsernameTaken(hospital, inputUsername)) :
        printError("Registrasi gagal! Pasien dengan nama tersebut
sudah terdaftar.")
        → false
    (strlen(password) < 6) :
        printError("Password harus minimal 6 karakter!")

```

```

→ false
(hospital.users.nEff ≥ hospital.users.capacity or
hospital.patients.nEff ≥ hospital.patients.capacity) :
    printError("Kapasitas pengguna atau pasien penuh!")
→ false
(strlen(password) ≥ 100) :
    printError("Panjang password terlalu besar untuk
dienkripsi!")
→ false
(otherwise) :
    maxId ← 0
    i traversal [0...hospital.users.nEff-1]
        if (hospital.users.elements[i].id > maxId) then
            maxId ← hospital.users.elements[i].id
    newPatientId ← maxId + 1

    newUser ← &hospital.users.elements[hospital.users.nEff]
    newUser.id ← newPatientId
    strcpy(newUser.username, inputUsername)

    if (not enigmaEncrypt(password,
newUser.password.encryptedContent, 100)) then
        printError("Gagal mengenkripsi password!")
    → false

    newUser.role ← PATIENT

    newPatient ←
&hospital.patients.elements[hospital.patients.nEff]
    newPatient.id ← newPatientId
    strcpy(newPatient.username, inputUsername)
    newPatient.bananaRich ← 100.0
    newPatient.life ← 3
    newPatient.diagnosedStatus ← false
    newPatient.treatedStatus ← false

    newPatient.medicationsPrescribed.medicationId ← (pointer
to integer) safeMalloc(10 * sizeof(integer))

```

```

if (newPatient.medicationsPrescribed.medicationId = NIL)
then
    printError("Gagal alokasi memori untuk resep pasien!")
    → false
    newPatient.medicationsPrescribed.capacity ← 10
    newPatient.medicationsPrescribed.nEff ← 0

    newPatient.medicationsTaken.medicationId ← (pointer to
integer) safeMalloc(10 * sizeof(integer))
    if (newPatient.medicationsTaken.medicationId = NIL) then
        printError("Gagal alokasi memori untuk obat yang
diminum pasien!")

dealokasi(newPatient.medicationsPrescribed.medicationId)
    newPatient.medicationsPrescribed.medicationId ← NIL
    → false
    newPatient.medicationsTaken.capacity ← 10
    newPatient.medicationsTaken.top ← -1
    newPatient.queueRoom[0] ← '\0'
    newPatient.queuePosition ← 0

hospital.users.nEff ← hospital.users.nEff + 1
hospital.patients.nEff ← hospital.patients.nEff + 1

strcpy(successMessage, "Pasien dengan nama ")
strcat(successMessage, inputUsername)
strcat(successMessage, " berhasil terdaftar!")
printSuccess(successMessage)

session.userId ← newPatientId
strcpy(session.username, inputUsername)
session.role ← PATIENT
session.isLoggedIn ← true

output("Fitur ini terintegrasi dengan fitur login, kamu
tidak perlu login lagi.")
output("Anda langsung masuk sebagai pasien!")

→ true

```

## F03 - Logout

```
{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
    < userId : integer;
        username : string;
        role : enumerasi;
        isLoggedIn : boolean >

{ Definisi function & procedure yang di-import dari file source code
lain }
function strcpy(output destination : string, input source : string)
    { Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
    { Menggabungkan string source ke akhir string destination. }

procedure printError(input message : string)
    { Mencetak pesan error. }

procedure printSuccess(input message : string)
    { Mencetak pesan sukses. }

{ Prosedur utama }
function logout(input/output session: pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Session.
  F.S. Mengembalikan true jika logout berhasil dan sesi direset; false
jika gagal.
  Jika gagal, pesan error yang sesuai ditampilkan.
}

KAMUS LOKAL
    tempUsername : character
    successMessage : character

ALGORITMA
    depend on (true) :
        (session = NIL) :
            printError("Sesi tidak valid!")
```

```

→ false
(not session.isLoggedIn) :
    printError("Anda belum login! Silakan login terlebih dahulu.")
→ false
(otherwise) :
    strcpy(tempUsername, session.username)

    session.userId ← -1
    session.username[0] ← '\0'
    session.role ← -1
    session.isLoggedIn ← false

    successMessage[0] ← '\0'
    strcat(successMessage, "Logout berhasil! Sampai jumpa, ")
    strcat(successMessage, tempUsername)
    strcat(successMessage, "!")

    printSuccess(successMessage)
→ true

```

#### F04 - Lupa Password

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Hospital :
< users : UserList >

type UserList :
< elements : array [1..N] of User;
  nEff : integer;
  capacity : integer >

type User :
< id : integer;
  username : string;
  password : PasswordData;
  role : Role >

type PasswordData :

```

```

< encryptedContent : string >

{ Definisi function & procedure yang di-import dari file source code
lain }

function strlen(input s : string) → integer
    { Mencari panjang string s }

function strcmp(input s1 : string, input s2 : string) → integer
    { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function strcat(output destination : string, input source : string)
    { Menggabungkan string source ke akhir string destination. }

procedure printError(input message : string)
    { Mencetak pesan error. }

procedure printSuccess(input message : string)
    { Mencetak pesan sukses. }

function isValidUsername(input username : string) → boolean
    { Memvalidasi format username. }

function generateRleCode(input username : string, output rleCode : string,
input bufferSize : integer) → boolean
    { Menghasilkan kode RLE dari username. }

function enigmaEncrypt(input password : string, output encryptedPassword : string,
input bufferSize : integer) → boolean
    { Mengenkripsi password. }

function forgotPassword(input hospital: pointer to Hospital, input
username: String, input rleCode: String, input newPassword: String) → boolean
    { I.S. Menerima pointer ke struktur Hospital, username, kode RLE,
    dan password baru.
    F.S. Mengembalikan true jika password berhasil direset; false jika
    gagal.
    Jika gagal, pesan error yang sesuai ditampilkan.
    }

```

```

{ Prosedur utama }
procedure forgotPassword(input/output UserList : users)
{memasukkan input password users baru jika ke dalam UserList}
{
    I.S Mengambil informasi username user
    F.S Mengembalikan informasi password baru
}

```

#### KAMUS LOKAL

```

userIdx : integer
passwordLength : integer
expectedRle : character
successMessage : character
i : integer

```

#### ALGORITMA

```

depend on (true) :
(hospital = NIL) :
    printError("Struktur rumah sakit tidak valid!")
    → false
(username = NIL or username[0] = '\0' or rleCode = NIL or
rleCode[0] = '\0' or newPassword = NIL or newPassword[0] = '\0') :
    printError("Input tidak valid!")
    → false
(not isValidUsername(username)) :
    printError("Username tidak valid!")
    → false
(strlen(newPassword) < 6) :
    printError("Password baru harus minimal 6 karakter!")
    → false
(hospital.users.nEff ≤ 0) :
    printError("Tidak ada pengguna terdaftar!")
    → false
(otherwise) :
    userIdx ← -1
    i traversal [0...hospital.users.nEff-1]
        if (strcmp(hospital.users.elements[i].username,
username) = 0) then
            userIdx ← i
            stop

```

```

if (userIdx = -1) then
    printError("Username tidak terdaftar!")
    → false
else
    passwordLength ← strlen(newPassword)
    if (passwordLength ≥ 100) then
        printError("Panjang password terlalu besar untuk
dienkripsi!")
        → false
    else
        if (not generateRleCode(username, expectedRle,
100)) then
            printError("Gagal menghasilkan kode RLE!")
            → false
        else
            if (strcmp(rleCode, expectedRle) ≠ 0) then
                printError("Kode RLE salah!")
                → false
            else
                if (not enigmaEncrypt(newPassword,
hospital.users.elements[userIdx].password.encryptedContent, 100))
then
                    printError("Gagal mengenkripsi
password baru!")
                    → false
                else
                    successMessage[0] ← '\0'
                    strcat(successMessage, "Password untuk
")
                    strcat(successMessage, username)
                    strcat(successMessage, " berhasil
diperbarui!")
                    printSuccess(successMessage)
                    → true

```

{ Fungsi Helper }

```

function generateRleCode(input username: string, output output:
character, input maxLen: integer) → boolean
    { I.S. Menerima username (string), buffer output (array of character), dan panjang maksimum buffer.
        F.S. Mengembalikan true jika kode RLE berhasil dihasilkan dan disimpan di output; false jika gagal.
        Jika gagal, pesan error yang sesuai ditampilkan.
    }

```

## F05 - Menu & Help

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
    < userId : integer;
        username : string;
        role : Role;
        isLoggedIn : boolean >

{ Definisi function & procedure yang di-import dari file source code lain }
function strcmp(input s1 : string, input s2 : string) → integer
    { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function strcpy(output destination : string, input source : string)
    { Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
    { Menggabungkan string source ke akhir string destination. }

procedure printHeader(input title : string)
    { Mencetak header dengan judul tertentu. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
    { Mencetak batas tabel. }

```

```

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak satu baris tabel. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure displayHelp(input session: pointer to Session, input command:
String)
{ I.S. Menerima pointer ke struktur Session dan string perintah.
F.S. Menampilkan deskripsi bantuan untuk perintah yang diminta atau
pesan error jika perintah tidak valid.
}

KAMUS LOKAL
errorMessage : character

ALGORITMA
    if (session = NIL) then
        printError("Sesi tidak valid!")
        return
    else if (command = NIL or command[0] = '\0') then
        printError("Perintah tidak valid!")
        return
    printHeader("Bantuan Perintah")

    if (not session.isLoggedIn) then
        depend on (command) :
            ("LOGIN") :
                output("LOGIN: Masuk ke sistem dengan username dan
password.")
                stop
            ("REGISTER_PATIENT") :
                output("REGISTER_PATIENT: Daftar sebagai pasien baru
dengan username dan password.")
                stop
            ("LUPA_PASSWORD") :
                output("LUPA_PASSWORD: Atur ulang kata sandi
menggunakan username dan kode verifikasi.")
                stop

```

```

("MENU") :
    output("MENU: Tampilkan daftar perintah yang
tersedia.")
    stop
("HELP") :
    output("HELP: Tampilkan bantuan untuk perintah
tertentu.")
    stop
("EXIT") :
    output("EXIT: Keluar dari sistem.")
    stop
(otherwise) :
    strcpy(errorMessage, "Perintah ")
    strcat(errorMessage, command)
    strcat(errorMessage, " tidak dikenal!")
    printError(errorMessage)
    stop
else
    depend on (session.role) :
        (MANAGER) :
            depend on (command) :
                ("LOGOUT") :
                    output("LOGOUT: Keluar dari sesi saat ini.")
                    stop
                ("MENU") :
                    output("MENU: Tampilkan daftar perintah yang
tersedia.")
                    stop
                ("HELP") :
                    output("HELP: Tampilkan bantuan untuk perintah
tertentu.")
                    stop
                ("LIHAT_DENAH") :
                    output("LIHAT_DENAH: Tampilkan denah rumah
sakit atau detail ruangan.")
                    stop
                ("LIHAT_USER") :
                    output("LIHAT_USER: Tampilkan daftar pengguna
(pasien atau dokter).")

```

```

stop
("CARI_USER") :
output("CARI_USER: Cari pengguna berdasarkan
ID atau username.")

stop
("LIHAT_ANTRIAN") :
output("LIHAT_ANTRIAN: Tampilkan antrian
pasien di setiap ruangan.")

stop
("TAMBAH_DOKTER") :
output("TAMBAH_DOKTER: Tambah dokter baru atau
tetapkan dokter ke ruangan.")

stop
("UBAH_DENAH") :
output("UBAH_DENAH: Ubah ukuran denah rumah
sakit.")

stop
("PINDAH_DOKTER") :
output("PINDAH_DOKTER: Pindahkan dokter ke
ruangan lain.")

stop
("LIHAT_FINANSIAL") :
output("LIHAT_FINANSIAL: Tampilkan laporan
keuangan rumah sakit.")

stop
("EXIT") :
output("EXIT: Keluar dari sistem.")

stop
(otherwise) :
strcpy(errorMessage, "Perintah ")
strcat(errorMessage, command)
strcat(errorMessage, " tidak dikenal!")
printError(errorMessage)

stop

(DOCTOR) :
depend on (command) :
("LOGOUT") :
output("LOGOUT: Keluar dari sesi saat ini.")

stop

```

```

("MENU") :
    output("MENU: Tampilkan daftar perintah yang
tersedia.")
    stop
("HELP") :
    output("HELP: Tampilkan bantuan untuk perintah
tertentu.")
    stop
("LIHAT_DENAH") :
    output("LIHAT_DENAH: Tampilkan denah rumah
sakit atau detail ruangan.")
    stop
("DIAGNOSIS") :
    output("DIAGNOSIS: Lakukan diagnosis otomatis
untuk pasien.")
    stop
("TREAT") :
    output("TREAT: Berikan resep obat otomatis
berdasarkan penyakit pasien.")
    stop
("LIHAT_DOMPET") :
    output("LIHAT_DOMPET: Tampilkan saldo
BananaRich dokter.")
    stop
("EXIT") :
    output("EXIT: Keluar dari sistem.")
    stop
(otherwise) :
    strcpy(errorMessage, "Perintah ")
    strcat(errorMessage, command)
    strcat(errorMessage, " tidak dikenal!")
    printError(errorMessage)
    stop
(PATIENT) :
    depend on (command) :
("LOGOUT") :
    output("LOGOUT: Keluar dari sesi saat ini.")
    stop
("MENU") :

```

```

        output("MENU: Tampilkan daftar perintah yang
tersedia.")

        stop
        ("HELP") :
        output("HELP: Tampilkan bantuan untuk perintah
tertentu.")

        stop
        ("LIHAT_DENAH") :
        output("LIHAT_DENAH: Tampilkan denah rumah
sakit atau detail ruangan.")

        stop
        ("PULANG_DOK") :
        output("PULANG_DOK: Periksa apakah pasien
boleh pulang berdasarkan status pengobatan.")

        stop
        ("DAFTAR_CHECKUP") :
        output("DAFTAR_CHECKUP: Daftar untuk
pemeriksaan dengan dokter tertentu.")

        stop
        ("ANTRIAN_SAYA") :
        output("ANTRIAN_SAYA: Tampilkan status antrian
pasien saat ini.")

        stop
        ("MINUM_OBAT") :
        output("MINUM_OBAT: Minum obat sesuai urutan
yang diresepkan.")

        stop
        ("MINUM_PENAWAR") :
        output("MINUM_PENAWAR: Minum penawar untuk
membatalkan obat terakhir.")

        stop
        ("LIHAT_DOMPET") :
        output("LIHAT_DOMPET: Tampilkan saldo
BananaRich pasien.")

        stop
        ("GACHA") :
        output("GACHA: Gunakan BananaRich untuk
bermain gacha.")

        stop

```

```

("LEWATI_ANTRIAN") :
    output("LEWATI_ANTRIAN: Lewati antrian untuk
pemeriksaan.")

    stop

("BATALKAN_ANTRIAN") :
    output("BATALKAN_ANTRIAN: Batalkan antrian
pemeriksaan saat ini.")

    stop

("EXIT") :
    output("EXIT: Keluar dari sistem.")

    stop

(otherwise) :
    strcpy(errorMessage, "Perintah ")
    strcat(errorMessage, command)
    strcat(errorMessage, " tidak dikenal!")
    printError(errorMessage)

    stop

```

{ Fitur Helper }

```

procedure displayMenu(input session: pointer to Session)
{ Spesifikasi prosedur }
{ I.S. Menerima pointer ke struktur Session.
  F.S. Menampilkan daftar perintah yang tersedia berdasarkan status
  login dan peran pengguna.
}

```

```

procedure printFootnote()
{ Spesifikasi prosedur }
{ I.S. Tidak ada.
  F.S. Mencetak catatan kaki ke konsol.
}

```

F06 - Denah Rumah Sakit

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

```

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< layout : Layout;
  doctors : DoctorList;
  patients : PatientList >

type Layout :
< elements : array [1..rowMax][1..colMax] of Room;
  rowEff : integer;
  colEff : integer >

type Room :
< code : string;
  capacity : integer;
  doctorId : integer;
  patientInRoom : PatientInRoomList >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer >

type Doctor :
< id : integer;
  username : string >

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer >

type Patient :
< id : integer;
  username : string >

```

```

type PatientInRoomList :
  < patientId : array [1..N] of integer;
    nEff : integer >

{ Definisi function & procedure yang di-import dari file source code
lain }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

function isValidRoomCode(input hospital : pointer to Hospital, input
roomCode : string) → boolean
{ Memvalidasi format kode ruangan dan keberadaannya di denah. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak baris tabel. }

{ Prosedur utama }
procedure displayLayout(input hospital: pointer to Hospital, input
session: pointer to Session, input printHeaderFlag: boolean)
{ I.S. Menerima pointer ke struktur Hospital, Session, dan sebuah flag
boolean.
}

```

F.S. Menampilkan denah rumah sakit secara visual jika kondisi akses terpenuhi.

}

#### KAMUS LOKAL

i, j : integer

#### ALGORITMA

```
depend_on (true) :
    (hospital = NIL or session = NIL) :
        printError("Struktur rumah sakit atau sesi tidak valid!")
        return
    (not session.isLoggedIn or (session.role ≠ MANAGER and
    session.role ≠ DOCTOR and session.role ≠ PATIENT)) :
        printError("Akses ditolak! Hanya Manajer, Dokter, atau
    Pasien yang dapat melihat denah.")
        return
    (hospital.layout.rowEff ≤ 0 or hospital.layout.colEff ≤ 0) :
        printError("Denah rumah sakit kosong!")
        return
    (otherwise) :
        if (printHeaderFlag) then
            printHeader("Denah Rumah Sakit")

            output("Γ")
            j traversal [0...hospital.layout.colEff-1]
                output("———|")
            output("|\n")

            i traversal [0...hospital.layout.rowEff-1]
                output("| ")
                j traversal [0...hospital.layout.colEff-1]
                    -- COLOR_YELLOW dan COLOR_RESET diasumsikan
                    sebagai makro untuk pewarnaan
                    output(" ", hospital.layout.elements[i][j].code, " "
                |")
                output("|\n")

            if (i ≠ hospital.layout.rowEff - 1) then
                output("|")
```

```

j traversal [0...hospital.layout.colEff-1]
    output("—+")
    output("|\n")

output("L")
j traversal [0...hospital.layout.colEff-1]
    output("—+")
    output("J|\n")


procedure displayRoomDetails(input hospital: pointer to Hospital,
input session: pointer to Session, input roomCode: String)
{ Spesifikasi prosedur }
{ I.S. Menerima pointer ke struktur Hospital, Session, dan string
kode ruangan.
F.S. Menampilkan detail ruangan tertentu (kapasitas, dokter,
pasien) jika kondisi akses dan validasi terpenuhi.
}

KAMUS LOKAL
room : pointer to Room
capacityStr : character
capacity : integer
row1 : array [1..2] of string
widths1 : array [1..2] of integer
doctorStr : character
found : boolean
row2 : array [1..2] of string
widths2 : array [1..2] of integer
row3 : array [1..2] of string
widths3 : array [1..2] of integer
row4 : array [1..2] of string
idStr : character
i, j : integer
header : character

ALGORITMA
depend on (true) :
(hospital = NIL or session = NIL) :
    printError("Struktur rumah sakit atau sesi tidak valid!")

```

```

        return
    (not session.isLoggedIn or (session.role ≠ MANAGER and
session.role ≠ DOCTOR and session.role ≠ PATIENT)) :
        printError("Akses ditolak! Hanya Manajer, Dokter, atau
Pasien yang dapat melihat detail ruangan.")
        return
    (roomCode = NIL or roomCode[0] = '\0') :
        printError("Kode ruangan tidak valid!")
        return
    (not isValidRoomCode(hospital, roomCode)) :
        printError("Kode ruangan tidak valid! Contoh: A1")
        return
    (hospital.layout.rowEff ≤ 0 or hospital.layout.colEff ≤ 0) :
        printError("Denah rumah sakit kosong!")
        return
    (otherwise) :
        room ← NIL
        i traversal [0...hospital.layout.rowEff-1]
            j traversal [0...hospital.layout.colEff-1]
                if (strcmp(hospital.layout.elements[i][j].code,
roomCode) = 0) then
                    room ← &hospital.layout.elements[i][j]
                    stop
                if (room ≠ NIL) then
                    stop

                if (room = NIL) then
                    strcpy(errorMessage, "Ruang ")
                    strcat(errorMessage, roomCode)
                    strcat(errorMessage, " tidak ditemukan!")
                    printError(errorMessage)
                    return

                strcpy(header, "Detail Ruangan ")
                strcat(header, roomCode)
                printHeader(header)

            capacity ← room.capacity
            depend on (true) :

```

```

(capacity < 10) :
    capacityStr[0] ← '0' + capacity
    capacityStr[1] ← '\0'
    stop
(capacity < 100) :
    capacityStr[0] ← '0' + (capacity div 10)
    capacityStr[1] ← '0' + (capacity mod 10)
    capacityStr[2] ← '\0'
    stop
(otherwise) :
    capacityStr[0] ← '1'
    capacityStr[1] ← '0'
    capacityStr[2] ← '0'
    capacityStr[3] ← '\0'
    stop

row1[0] ← "Kapasitas"
row1[1] ← capacityStr
widths1[0] ← 15
widths1[1] ← 20
printTableBorder(widths1, 2, 1)
printTableRow(row1, widths1, 2)
printTableBorder(widths1, 2, 2)

if (room.doctorId = -1 or hospital.doctors.nEff ≤ 0) then
    strcpy(doctorStr, "Tidak ada dokter")
else
    found ← false
    i traversal [0...hospital.doctors.nEff-1]
        if (hospital.doctors.elements[i].id =
room.doctorId) then
            strcpy(doctorStr,
hospital.doctors.elements[i].username)
            found ← true
            stop
        if (not found) then
            strcpy(doctorStr, "Dokter tidak ditemukan")

row2[0] ← "Dokter"

```

```

row2[1] ← doctorStr
widths2[0] ← 15
widths2[1] ← 20
printTableRow(row2, widths2, 2)
printTableBorder(widths2, 2, 3)

row3[0] ← "No."
row3[1] ← "Pasien dalam Ruangan"
widths3[0] ← 5
widths3[1] ← 30
printTableBorder(widths3, 2, 1)
printTableRow(row3, widths3, 2)
printTableBorder(widths3, 2, 2)

if (room.patientInRoom.nEff = 0 or hospital.patients.nEff
≤ 0) then
    row4[0] ← "0."
    row4[1] ← "Tidak ada pasien"
    printTableRow(row4, widths3, 2)
else
    i traversal [0...room.patientInRoom.nEff-1]
    found ← false
    j traversal [0...hospital.patients.nEff-1]
    if (hospital.patients.elements[j].id =
room.patientInRoom.patientId[i]) then
        -- Mengingat ini untuk tujuan notasi
algoritmik,
        -- asumsi konversi integer ke string yang
benar sudah terjadi.
        -- Pada implementasi sebenarnya, ini
mungkin memerlukan fungsi sprintf.
        -- Contoh: sprintf(idStr, "%d.", i + 1);
        -- Karena permintaan spesifik untuk
mempertahankan logika asli,
        -- ini akan menghasilkan karakter ASCII
yang salah jika i+1 ≥ 10.
    idStr[0] ← (i + 1) + '0'
    idStr[1] ← '.'
    idStr[2] ← '\0'

```

```

        row4[0] ← idStr
        row4[1] ←
hospital.patients.elements[j].username
        printTableRow(row4, widths3, 2)
        found ← true
stop
if (not found) then
        idStr[0] ← (i + 1) + '0'
        idStr[1] ← '.'
        idStr[2] ← '\0'
        row4[0] ← idStr
        row4[1] ← "Data pasien tidak ditemukan"
        printTableRow(row4, widths3, 2)
printTableBorder(widths3, 2, 3)

```

{ Fungsi Helper }

```

procedure displayLayout(input hospital: Hospital, input session:
Session)
{ Menampilkan denah rumah sakit dalam bentuk grid matrix. }

```

F07 - Lihat User

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
    username : string;
    role : Role;
    isLoggedIn : boolean >

type Hospital :
< users : UserList;
    patients : PatientList;
    doctors : DoctorList >

```

```

type UserList :
< elements : array [1..N] of User;
  nEff : integer;
  capacity : integer >

type User :
< id : integer;
  username : string;
  role : Role >

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer;
  capacity : integer >

type Patient :
< id : integer;
  username : string;
  disease : string >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer;
  capacity : integer >

type Doctor :
< id : integer;
  username : string;
  aura : integer >

```

```

{ Definisi function & procedure yang di-import dari file source code
lain }

function strlen(input s : string) → integer
{ Mencari panjang string s. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

procedure printError(input message : string)
{ Mencetak pesan error. }

```

```

procedure printHeader(input title : string)
{ Mencetak judul header. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak baris tabel. }

function integerToString(input num : integer, output str : string, input
size : integer) → boolean
{ Mengonversi integer ke string. Mengembalikan true jika berhasil. }

```

{ Prosedur utama }

```

procedure displayUsers(input/output hospital : pointer to Hospital,
input session : pointer to Session, input viewType : integer)
{ Menampilkan daftar pengguna berdasarkan viewType, hanya bisa
diakses MANAGER }

```

#### KAMUS LOKAL

```

widths_users : array [1..4] of integer
headers_users : array [1..4] of string
user : pointer to User
idStr_users : character
roleStr : character
diseaseStr : character
row_users : array [1..4] of string
i, j : integer
id : integer

```

#### ALGORITMA

```

depend on (true) :
(hospital = NIL or session = NIL) :
printError("Struktur rumah sakit atau sesi tidak valid!")

```

```

        return
    (not session.isLoggedIn or session.role ≠ MANAGER) :
        printError("Akses ditolak! Hanya Manajer yang dapat
melihat daftar pengguna.")
        return
    (hospital.users.nEff = 0) :
        output("Tidak ada pengguna terdaftar.")
        return
    (otherwise) :
        printHeader("Daftar Pengguna")
        widths_users[0] ← 5
        widths_users[1] ← 20
        widths_users[2] ← 10
        widths_users[3] ← 20
        headers_users[0] ← "ID"
        headers_users[1] ← "Username"
        headers_users[2] ← "Role"
        headers_users[3] ← "Penyakit"
        printTableBorder(widths_users, 4, 1)
        printTableRow(headers_users, widths_users, 4)
        printTableBorder(widths_users, 4, 2)

        i traversal [0...hospital.users.nEff-1]
            user ← &hospital.users.elements[i]
            diseaseStr[0] ← '-'
            diseaseStr[1] ← '\0'

            id ← user.id
            depend on (true) :
                (id < 10) :
                    idStr_users[0] ← '0' + id
                    idStr_users[1] ← '\0'
                    stop
                (id < 100) :
                    idStr_users[0] ← '0' + (id div 10)
                    idStr_users[1] ← '0' + (id mod 10)
                    idStr_users[2] ← '\0'
                    stop
                (otherwise) :

```

```

        idStr_users[0] ← '1'
        idStr_users[1] ← '0'
        idStr_users[2] ← '0'
        idStr_users[3] ← '\0'
        stop

depend on (user.role) :
    (MANAGER) :
        strcpy(roleStr, "Manajer")
        stop
    (DOCTOR) :
        strcpy(roleStr, "Dokter")
        stop
    (PATIENT) :
        strcpy(roleStr, "Pasien")
        stop
    (otherwise) :
        strcpy(roleStr, "Tidak diketahui")
        stop

if (user.role = PATIENT and hospital.patients.nEff >
0) then
    j traversal [0...hospital.patients.nEff-1]
    if (hospital.patients.elements[j].id =
user.id) then
        if
        (strlen(hospital.patients.elements[j].disease) > 0 and
        hospital.patients.elements[j].disease[0] ≠ '\0') then
            strcpy(diseaseStr,
        hospital.patients.elements[j].disease)
            stop

        row_users[0] ← idStr_users
        row_users[1] ← user.username
        row_users[2] ← roleStr
        row_users[3] ← diseaseStr
        printTableRow(row_users, widths_users, 4)
        printTableBorder(widths_users, 4, 3)

```

```

procedure displayPatients(input hospital: pointer to Hospital, input
session: pointer to Session)
    { I.S. Menerima pointer ke struktur Hospital dan Session.
      F.S. Menampilkan daftar semua pasien dalam sistem
          jika pengguna yang login adalah Manajer; jika tidak, pesan
          error ditampilkan.
    }

```

#### KAMUS LOKAL

```

widths_patients : array [1..3] of integer
headers_patients : array [1..3] of string
patient : pointer to Patient
idStr_patients : character
diseaseStr_patients : character
row_patients : array [1..3] of string

```

#### ALGORITMA

```

depend on (true) :
    (hospital = NIL or session = NIL) :
        printError("Struktur rumah sakit atau sesi tidak valid!")
        return
    (not session.isLoggedIn or session.role ≠ MANAGER) :
        printError("Akses ditolak! Hanya Manajer yang dapat
melihat daftar pasien.")
        return
    (hospital.patients.nEff = 0) :
        output("Tidak ada pasien terdaftar.")
        return
    (otherwise) :
        printHeader("Daftar Pasien")
        widths_patients[0] ← 5
        widths_patients[1] ← 20
        widths_patients[2] ← 20
        headers_patients[0] ← "ID"
        headers_patients[1] ← "Username"
        headers_patients[2] ← "Penyakit"
        printTableBorder(widths_patients, 3, 1)
        printTableRow(headers_patients, widths_patients, 3)
        printTableBorder(widths_patients, 3, 2)

```

```

i traversal [0...hospital.patients.nEff-1]
    patient ← &hospital.patients.elements[i]
    diseaseStr_patients[0] ← '-'
    diseaseStr_patients[1] ← '\0'

    id ← patient.id
    depend on (true) :
        (id < 10) :
            idStr_patients[0] ← '0' + id
            idStr_patients[1] ← '\0'
            stop
        (id < 100) :
            idStr_patients[0] ← '0' + (id div 10)
            idStr_patients[1] ← '0' + (id mod 10)
            idStr_patients[2] ← '\0'
            stop
        (otherwise) :
            idStr_patients[0] ← '1'
            idStr_patients[1] ← '0'
            idStr_patients[2] ← '0'
            idStr_patients[3] ← '\0'
            stop

        if (strlen(patient.disease) > 0 and patient.disease[0]
        ≠ '\0') then
            strcpy(diseaseStr_patients, patient.disease)

            row_patients[0] ← idStr_patients
            row_patients[1] ← patient.username
            row_patients[2] ← diseaseStr_patients
            printTableRow(row_patients, widths_patients, 3)
            printTableBorder(widths_patients, 3, 3)

```

```

procedure displayDoctors(input hospital: pointer to Hospital, input
session: pointer to Session)
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Menampilkan daftar semua dokter dalam sistem

```

```
        jika pengguna yang login adalah Manajer; jika tidak, pesan  
        error ditampilkan.  
    }
```

#### KAMUS LOKAL

```
widths_doctors : array [1..3] of integer  
headers_doctors : array [1..3] of string  
doctor : pointer to Doctor  
idStr_doctors : character  
auraStr : character  
row_doctors : array [1..3] of string  
aura : integer
```

#### ALGORITMA

```
depend on (true) :  
    (hospital = NIL or session = NIL) :  
        printError("Struktur rumah sakit atau sesi tidak valid!")  
        return  
    (not session.isLoggedIn or session.role ≠ MANAGER) :  
        printError("Akses ditolak! Hanya Manajer yang dapat  
        melihat daftar dokter.")  
        return  
    (hospital.doctors.nEff = 0) :  
        output("Tidak ada dokter terdaftar.")  
        return  
    (otherwise) :  
        printHeader("Daftar Dokter")  
        widths_doctors[0] ← 5  
        widths_doctors[1] ← 20  
        widths_doctors[2] ← 10  
        headers_doctors[0] ← "ID"  
        headers_doctors[1] ← "Username"  
        headers_doctors[2] ← "Aura"  
        printTableBorder(widths_doctors, 3, 1)  
        printTableRow(headers_doctors, widths_doctors, 3)  
        printTableBorder(widths_doctors, 3, 2)  
  
        i traversal [0...hospital.doctors.nEff-1]  
            doctor ← &hospital.doctors.elements[i]
```

```

        id ← doctor.id
        if (not integerToString(id, idStr_doctors, 10)) then
-- sizeof(idStr) = 10
        printError("Gagal mengonversi ID ke string!")
        return

        aura ← doctor.aura
        if (not integerToString(aura, auraStr, 10)) then --
sizeof(auraStr) = 10
        printError("Gagal mengonversi aura ke string!")
        return

        row_doctors[0] ← idStr_doctors
        row_doctors[1] ← doctor.username
        row_doctors[2] ← auraStr
        printTableRow(row_doctors, widths_doctors, 3)
        printTableBorder(widths_doctors, 3, 3)
    
```

**Notes** : Fungsi displayDoctors terhubung dengan fitur B03 - Aura!

#### F08 - Cari User

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< layout : Layout;
  doctors : DoctorList;
  patients : PatientList;

```

```

users : UserList > { Tambahan: Struktur UserList }

type UserList :
< elements : array [1..N] of User;
  nEff : integer >

type User :
< id : integer;
  username : string;
  role : Role >

type Layout :
< elements : array [1..rowMax][1..colMax] of Room;
  rowEff : integer;
  colEff : integer >

type Room :
< code : string;
  capacity : integer;
  doctorId : integer;
  patientInRoom : PatientInRoomList >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer >

type Doctor :
< id : integer;
  username : string;
  aura : real >

```

```

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer >

type Patient :
< id : integer;
  username : string;
  disease : string >

type PatientInRoomList :
< patientId : array [1..N] of integer;
  nEff : integer >

{ Definisi function & procedure yang di-import dari file source code
lain }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function strlen(input s : string) → integer
{ Mengembalikan panjang string s. }

procedure printError(input message : string)
{ Mencetak pesan error. }

```

```

procedure printHeader(input title : string)
{ Mencetak judul header. }

function isValidRoomCode(input hospital : pointer to Hospital, input
roomCode : string) → boolean
{ Memvalidasi format kode ruangan dan keberadaannya di denah. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak baris tabel. }

function stringToInt(input s: string) → integer
{ Mengkonversi string ke integer. Mengembalikan -1 jika konversi
gagal. }

function integerToString(input num : integer, output str : string, input
size : integer) → boolean
{ Mengkonversi integer ke string. Mengembalikan true jika berhasil,
false jika tidak. }

function floatToString(input num : real, input precision : integer,
output str : string, input size : integer) → boolean
{ Mengkonversi float ke string dengan presisi tertentu.
Mengembalikan true jika berhasil, false jika tidak. }

```

```

procedure qsort(input array : array of any, input num : integer, input
size : integer, input compare : pointer to function)
    { Mengurutkan array menggunakan algoritma Quicksort. }

{ Fungsi pembantu statis }

function localCharToLower(input c : character) → character
{ Spesifikasi fungsi }
{ I.S. Menerima sebuah karakter.
  F.S. Mengembalikan karakter tersebut dalam huruf kecil jika itu adalah
huruf kapital, jika tidak, mengembalikan karakter asli.
}

```

{ Fungsi Utama }

```

procedure findUser(input hospital : pointer to Hospital, input session :
pointer to Session, input query : string, input byId : boolean)
    { I.S. Menerima pointer ke struktur Hospital dan Session, string
query, dan boolean byId.
  F.S. Mencari dan menampilkan informasi pengguna berdasarkan ID
atau username.
}

```

#### KAMUS LOKAL

```

overallFound : boolean
printedTableHeaders : boolean
widths : array [0..3] of integer
headers : array [0..3] of string
idStr : character
roleStr : character
diseaseStr : character

```

```

targetId : integer
keyUser : User
foundUser : pointer to User
user : pointer to User
exactMatchFound : boolean
suggestionsFound : boolean
row : array [0..3] of string
i, j : integer

```

#### ALGORITMA

```

if (hospital = NIL or session = NIL or query = NIL) then
    printError("Struktur rumah sakit, sesi, atau kueri tidak
valid!")
    → { void }

if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat mencari
pengguna.")
    → { void }

printHeader("Hasil Pencarian Pengguna")
if (hospital.users.nEff = 0) then
    output("Tidak ada pengguna terdaftar.\n")
    → { void }

overallFound ← false
printedTableHeaders ← false
widths[0] ← 5; widths[1] ← 20; widths[2] ← 10; widths[3] ← 20
headers[0] ← "ID"; headers[1] ← "Username"; headers[2] ←
"Role"; headers[3] ← "Penyakit"
if (byId) then
    targetId ← stringToInt(query)
    if (targetId = -1 and strcmp(query, "-1") ≠ 0 and
strcmp(query, "0") ≠ 0) then

```

```

    if (strcmp(query, "-1") ≠ 0) then
        printError("ID pencarian tidak valid (harus berupa
angka).")

        output("Pengguna dengan ID '", query, "' tidak
ditemukan.\n")

        → { void }

keyUser.id ← targetId

qsort(hospital.users.elements, hospital.users.nEff,
sizeof(User), compareUsersById)

foundUser ← customBinarySearchUsers(&keyUser,
hospital.users.elements, hospital.users.nEff, compareUsersById)

if (foundUser ≠ NIL) then
    if (not printedTableHeaders) then
        printTableBorder(widths, 4, 1)
        printTableRow(headers, widths, 4)
        printTableBorder(widths, 4, 2)
        printedTableHeaders ← true

    overallFound ← true

    if (not integerToString(foundUser.id, idStr,
sizeof(idStr))) then
        strcpy(idStr, "ERR")

    depend on (foundUser.role) :
        (MANAGER) : strcpy(roleStr, "Manajer")
        (DOCTOR) : strcpy(roleStr, "Dokter")
        (PATIENT) : strcpy(roleStr, "Pasien")
        (otherwise) : strcpy(roleStr, "N/A")

    strcpy(diseaseStr, "-")

    if (foundUser.role = PATIENT) then
        j traversal [0...hospital.patients.nEff-1]

```

```

        if (hospital.patients.elements[j].id =
foundUser.id and hospital.patients.elements[j].disease[0] ≠ '\0')
then

            strcpy(diseaseStr,
hospital.patients.elements[j].disease)

            stop { break }

        row[0] ← idStr
        row[1] ← foundUser.username
        row[2] ← roleStr
        row[3] ← diseaseStr
        printTableRow(row, widths, 4)

else { Search by name (two-tier) }

    exactMatchFound ← false
    i traversal [0...hospital.users.nEff-1]
    user ← &hospital.users.elements[i]
    if (localCaseInsensitivestrcmp(user.username, query) = 0)

then

        if (not printedTableHeaders) then
            printTableBorder(widths, 4, 1)
            printTableRow(headers, widths, 4)
            printTableBorder(widths, 4, 2)
            printedTableHeaders ← true
        overallFound ← true
        exactMatchFound ← true
        if (not integerToString(user.id, idStr,
sizeof(idStr))) then
            strcpy(idStr, "ERR")
            depend on (user.role) :
                (MANAGER) : strcpy(roleStr, "Manajer")
                (DOCTOR) : strcpy(roleStr, "Dokter")

```

```

(PATIENT) : strcpy(roleStr, "Pasien")
(otherwise) : strcpy(roleStr, "N/A")
strcpy(diseaseStr, "-")
if (user.role = PATIENT) then
    j traversal [0...hospital.patients.nEff-1]
        if (hospital.patients.elements[j].id = user.id
and hospital.patients.elements[j].disease[0] ≠ '\0') then
            strcpy(diseaseStr,
hospital.patients.elements[j].disease)
            stop { break }

row[0] ← idStr
row[1] ← user.username
row[2] ← roleStr
row[3] ← diseaseStr
printTableRow(row, widths, 4)

if (not exactMatchFound and query[0] ≠ '\0') then
    suggestionsFound ← false
    i traversal [0...hospital.users.nEff-1]
        user ← &hospital.users.elements[i]
        if
            (localContainsCaseInsensitiveSubstring(user.username, query)) then
                if (not printedTableHeaders) then
                    printTableBorder(widths, 4, 1)
                    printTableRow(headers, widths, 4)
                    printTableBorder(widths, 4, 2)
                    printedTableHeaders ← true
                if (not suggestionsFound) then
                    output("Tidak ada hasil pencocokan pasti.

Mungkin maksud Anda:\n")
                    suggestionsFound ← true

```

```

        overallFound ← true
        if (not integerToString(user.id, idStr,
sizeof(idStr))) then
            strcpy(idStr, "ERR")
depend on (user.role) :
    (MANAGER) : strcpy(roleStr, "Manajer")
    (DOCTOR) : strcpy(roleStr, "Dokter")
    (PATIENT) : strcpy(roleStr, "Pasien")
    (otherwise) : strcpy(roleStr, "N/A")
    strcpy(diseaseStr, "-")
    if (user.role = PATIENT) then
        j traversal [0...hospital.patients.nEff-1]
        if (hospital.patients.elements[j].id =
user.id and hospital.patients.elements[j].disease[0] ≠ '\0') then
            strcpy(diseaseStr,
hospital.patients.elements[j].disease)
            stop { break }
        row[0] ← idStr
        row[1] ← user.username
        row[2] ← roleStr
        row[3] ← diseaseStr
        printTableRow(row, widths, 4)
    if (overallFound) then
        printTableBorder(widths, 4, 3)
    else
        if (printedTableHeaders) then
            printTableBorder(widths, 4, 3)
        output("Pengguna dengan ", byId ? "ID" : "nama", " ", query,
"' tidak ditemukan.\n")

```

```

procedure findPatient(input hospital : pointer to Hospital, input
session : pointer to Session, input query : string, input byId :
boolean, input byDisease : boolean)
    { I.S. Menerima pointer ke struktur Hospital dan Session, string
query, dan dua boolean untuk kriteria pencarian.

F.S. Mencari dan menampilkan informasi pasien berdasarkan ID,
username, atau penyakit.

}

```

#### KAMUS LOKAL

```

overallFound : boolean
printedTableHeaders : boolean
widths : array [0..2] of integer
headers : array [0..2] of string
idStr : character
diseaseStr : character
exactMatchFound : boolean
suggestionsFound : boolean
patient : pointer to Patient
targetId : integer
keyPatient : Patient
foundPatient : pointer to Patient
searchTypeStr : character
notFoundMsg : character
row : array [0..2] of string
i, j : integer

```

#### ALGORITMA

```

if (hospital = NIL or session = NIL or query = NIL) then
    printError("Struktur rumah sakit, sesi, atau kueri tidak
valid!")

```

```

→ { void }

if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat mencari
pasien.")

→ { void }

printHeader("Hasil Pencarian Pasien")

if (hospital.patients.nEff = 0) then
    output("Tidak ada pasien terdaftar.\n")
→ { void }

overallFound ← false

printedTableHeaders ← false

widths[0] ← 5; widths[1] ← 20; widths[2] ← 50

headers[0] ← "ID"; headers[1] ← "Username"; headers[2] ←
"Penyakit"

if (byDisease) then
    exactMatchFound ← false
    i traversal [0...hospital.patients.nEff-1]
        patient ← &hospital.patients.elements[i]
        if (localCaseInsensitivestrcmp(patient.disease, query) =
0) then
            if (not printedTableHeaders) then
                printTableBorder(widths, 3, 1)
                printTableRow(headers, widths, 3)
                printTableBorder(widths, 3, 2)
                printedTableHeaders ← true
            overallFound ← true
            exactMatchFound ← true
            if (not integerToString(patient.id, idStr,
sizeof(idStr))) then
                strcpy(idStr, "ERR")

```

```

        strcpy(diseaseStr, patient.disease[0] != '\0' ?
patient.disease : "-")
        row[0] ← idStr
        row[1] ← patient.username
        row[2] ← diseaseStr
        printTableRow(row, widths, 3)
    if (not exactMatchFound and query[0] ≠ '\0') then
        suggestionsFound ← false
    i traversal [0...hospital.patients.nEff-1]
        patient ← &hospital.patients.elements[i]
        if
            (localContainsCaseInsensitiveSubstring(patient.disease, query))
        then
            if (not printedTableHeaders) then
                printTableBorder(widths, 3, 1)
                printTableRow(headers, widths, 3)
                printTableBorder(widths, 3, 2)
                printedTableHeaders ← true
            if (not suggestionsFound) then
                output("Tidak ada hasil pencocokan pasti untuk
penyakit. Mungkin maksud Anda:\n")
                suggestionsFound ← true
            overallFound ← true
            if (not integerToString(patient.id, idStr,
sizeof(idStr))) then
                strcpy(idStr, "ERR")
                strcpy(diseaseStr, patient.disease[0] ≠ '\0' ?
patient.disease : "-")
                row[0] ← idStr
                row[1] ← patient.username

```

```

        row[2] ← diseaseStr
        printTableRow(row, widths, 3)

else if (byId) then
    targetId ← stringToInt(query)
    if (targetId = -1 and strcmp(query, "-1") ≠ 0 and
        strcmp(query, "0") ≠ 0) then
        printError("ID pencarian tidak valid.")
        output("Pasien dengan ID '", query, "' tidak
ditemukan.\n")
    → { void }

    keyPatient.id ← targetId
    qsort(hospital.patients.elements, hospital.patients.nEff,
    sizeof(Patient), comparePatientsById)

    foundPatient ← customBinarySearchPatients(&keyPatient,
    hospital.patients.elements, hospital.patients.nEff,
    comparePatientsById)

    if (foundPatient ≠ NIL) then
        if (not printedTableHeaders) then
            printTableBorder(widths, 3, 1)
            printTableRow(headers, widths, 3)
            printTableBorder(widths, 3, 2)
            printedTableHeaders ← true

        overallFound ← true
        if (not integerToString(foundPatient.id, idStr,
        sizeof(idStr))) then
            strcpy(idStr, "ERR")
            strcpy(diseaseStr, foundPatient.disease[0] ≠ '\0' ?
        foundPatient.disease : "-")

            row[0] ← idStr
            row[1] ← foundPatient.username

```

```

        row[2] ← diseaseStr
        printTableRow(row, widths, 3)
    else { Search by name }
        exactMatchFound ← false
        i traversal [0...hospital.patients.nEff-1]
            patient ← &hospital.patients.elements[i]
            if (localCaseInsensitivestrcmp(patient.username, query) =
0) then
                if (not printedTableHeaders) then
                    printTableBorder(widths, 3, 1)
                    printTableRow(headers, widths, 3)
                    printTableBorder(widths, 3, 2)
                    printedTableHeaders ← true
                overallFound ← true
                exactMatchFound ← true
                if (not integerToString(patient.id, idStr,
sizeof(idStr))) then
                    strcpy(idStr, "ERR")
                    strcpy(diseaseStr, patient.disease[0] ≠ '\0' ?
patient.disease : "-")
                    row[0] ← idStr
                    row[1] ← patient.username
                    row[2] ← diseaseStr
                    printTableRow(row, widths, 3)
                if (not exactMatchFound and query[0] ≠ '\0') then
                    suggestionsFound ← false
                    i traversal [0...hospital.patients.nEff-1]
                        patient ← &hospital.patients.elements[i]

```

```

if
(localContainsCaseInsensitiveSubstring(patient.username, query))
then

    if (not printedTableHeaders) then
        printTableBorder(widths, 3, 1)
        printTableRow(headers, widths, 3)
        printTableBorder(widths, 3, 2)
        printedTableHeaders ← true
    if (not suggestionsFound) then
        output("Tidak ada hasil pencocokan pasti untuk
nama. Mungkin maksud Anda:\n")
        suggestionsFound ← true
    overallFound ← true
    if (not integerToString(patient.id, idStr,
sizeof(idStr))) then
        strcpy(idStr, "ERR")
        strcpy(diseaseStr, patient.disease[0] ≠ '\0' ?
patient.disease : "-")
        row[0] ← idStr
        row[1] ← patient.username
        row[2] ← diseaseStr
        printTableRow(row, widths, 3)
    if (overallFound) then
        printTableBorder(widths, 3, 3)
    else
        if (printedTableHeaders) then
            printTableBorder(widths, 3, 3)
        if (byDisease) then
            strcpy(searchTypeStr, "penyakit")
        else if (byId) then

```

```

        strcpy(searchTypeStr, "ID")
else
        strcpy(searchTypeStr, "nama")
        strcpy(notFoundMsg, "Pasien dengan ")
        strcat(notFoundMsg, searchTypeStr)
        strcat(notFoundMsg, " ''")
        strcat(notFoundMsg, query)
        strcat(notFoundMsg, "' tidak ditemukan.")
output(notFoundMsg, "\n")

```

```

procedure findDoctor(input hospital : pointer to Hospital, input session
: pointer to Session, input query : string, input byId : boolean)
    { Spesifikasi prosedur }
    { I.S. Menerima pointer ke struktur Hospital dan Session, string
query, dan boolean byId.
    F.S. Mencari dan menampilkan informasi dokter berdasarkan ID
atau username.
}

```

#### KAMUS LOKAL

```

found : boolean
widths : array [0..2] of integer
headers : array [0..2] of string
targetId : integer
keyDoctor : Doctor
foundDoctor : pointer to Doctor
idStr : character
auraStr : character
row : array [0..2] of string
tempIdConv_doc : character

```

```

k_id_doc : integer
tempVal_id_doc : integer { long long }
isNeg_id_doc : boolean
numDigits_id_doc : integer
valCopy_id_doc : integer { long long }
floatValue_aura : real
precision_aura : integer
tempFloatStr_aura : character
intPart_aura : integer { long long }
intStr_aura : character
tempIntConv_aura : character
k_int_aura : integer
tempVal_int_aura : integer { long long }
numDigits_int_aura : integer
valCopy_int_aura : integer { long long }
fracPart_aura : real
p_aura : integer
digit_aura : integer
digitChar_aura : character
doctor : pointer to Doctor
tempIdConv_doc_name : character
k_id_doc_name : integer
tempVal_id_doc_name : integer { long long }
isNeg_id_doc_name : boolean
numDigits_id_doc_name : integer
valCopy_id_doc_name : integer { long long }
floatValue_aura_name : real
precision_aura_name : integer
tempFloatStr_aura_name : character
intPart_aura_name : integer { long long }

```

```

intStr_aura_name : character
tempIntConv_aura_name : character
k_int_aura_name : integer
tempVal_int_aura_name : integer { long long }
numDigits_int_aura_name : integer
valCopy_int_aura_name : integer { long long }
fracPart_aura_name : real
p_aura_name : integer
digit_aura_name : integer
digitChar_aura_name : character
i, m : integer

```

#### ALGORITMA

```

if (hospital = NIL or session = NIL or query = NIL) then
    printError("Struktur rumah sakit, sesi, atau kueri tidak
valid!")
    → { void }

if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat mencari
dokter.")
    → { void }

printHeader("Hasil Pencarian Dokter")

if (hospital.doctors.nEff = 0) then
    output("Tidak ada dokter terdaftar.\n")
    → { void }

found ← false
widths[0] ← 5; widths[1] ← 20; widths[2] ← 10
headers[0] ← "ID"; headers[1] ← "Username"; headers[2] ← "Aura"
printTableBorder(widths, 3, 1)
printTableRow(headers, widths, 3)
printTableBorder(widths, 3, 2)

```

```

if (byId) then
    targetId ← stringToInt(query)
    if (targetId = -1 and strcmp(query, "-1") ≠ 0) then
        if (strcmp(query, "-1") ≠ 0) then
            printError("ID pencarian tidak valid (harus berupa
angka).")
            printTableBorder(widths, 3, 3)
            output("Dokter dengan ID '", query, "' tidak
ditemukan.\n")
        → {void}
    keyDoctor.id ← targetId
    qsort(hospital.doctors.elements, hospital.doctors.nEff,
sizeof(Doctor), compareDoctorsById)
    foundDoctor ← customBinarySearchDoctors(&keyDoctor,
hospital.doctors.elements, hospital.doctors.nEff,
compareDoctorsById)

    if (foundDoctor ≠ NIL) then
        found ← true
        { Manual integerToString for foundDoctor.id }
        k_id_doc ← 0
        if (foundDoctor.id = 0) then
            tempIdConv_doc[k_id_doc] ← '0'
            k_id_doc ← k_id_doc + 1
        else
            tempVal_id_doc ← foundDoctor.id
            isNeg_id_doc ← false
            if (tempVal_id_doc < 0) then
                isNeg_id_doc ← true
                tempVal_id_doc ← -tempVal_id_doc
            numDigits_id_doc ← 0

```

```

valCopy_id_doc ← tempVal_id_doc
if (valCopy_id_doc = 0 and foundDoctor.id ≠ 0) then
    numDigits_id_doc ← 0
else if (valCopy_id_doc = 0) then
    numDigits_id_doc ← 1
else
    while (valCopy_id_doc > 0) do
        valCopy_id_doc ← valCopy_id_doc div 10
        numDigits_id_doc ← numDigits_id_doc + 1
    k_id_doc ← numDigits_id_doc
    if (tempVal_id_doc = 0 and foundDoctor.id ≠ 0) then
        -- do nothing
    else if (tempVal_id_doc = 0) then
        tempIdConv_doc[0] ← '0'
    else
        while (tempVal_id_doc > 0) do
            k_id_doc ← k_id_doc - 1
            tempIdConv_doc[k_id_doc] ← (tempVal_id_doc
mod 10) + '0'
            tempVal_id_doc ← tempVal_id_doc div 10
        k_id_doc ← numDigits_id_doc
        if (isNeg_id_doc) then
            m traversal [k_id_doc...0]
            tempIdConv_doc[m] ← tempIdConv_doc[m-1]
        tempIdConv_doc[0] ← '-'
        k_id_doc ← k_id_doc + 1
    tempIdConv_doc[k_id_doc] ← '\0'
    if (k_id_doc < sizeof(idStr)) then
        strcpy(idStr, tempIdConv_doc)
    else

```

```

        strcpy(idStr, "BIG")
    { Manual floatToString for foundDoctor.aura (precision 1)
}

    floatValue_aura ← foundDoctor.aura
    precision_aura ← 1
    auraStr[0] ← '\0'
    tempFloatStr_aura[0] ← '\0'
    if (floatValue_aura < 0.0) then
        floatValue_aura ← -floatValue_aura
        strcpy(tempFloatStr_aura, "-")
    intPart_aura ← (integer)floatValue_aura
    { Manual int-to-string for intPart_aura }
    k_int_aura ← 0
    if (intPart_aura = 0) then
        tempIntConv_aura[k_int_aura] ← '0'
        k_int_aura ← k_int_aura + 1
    else
        tempVal_int_aura ← intPart_aura
        numDigits_int_aura ← 0
        valCopy_int_aura ← tempVal_int_aura
        if (valCopy_int_aura = 0) then
            numDigits_int_aura ← 1
        else
            while (valCopy_int_aura > 0) do
                valCopy_int_aura ← valCopy_int_aura div 10
                numDigits_int_aura ← numDigits_int_aura + 1
            k_int_aura ← numDigits_int_aura
            if (tempVal_int_aura = 0) then
                tempIntConv_aura[0] ← '0'
            else

```

```

        while (tempVal_int_aura > 0) do
            k_int_aura ← k_int_aura - 1
            tempIntConv_aura[k_int_aura] ←
                (tempVal_int_aura mod 10) + '0'
                tempVal_int_aura ← tempVal_int_aura div 10
            k_int_aura ← numDigits_int_aura
            tempIntConv_aura[k_int_aura] ← '\0'
            strcpy(intStr_aura, tempIntConv_aura)
            strcat(tempFloatStr_aura, intStr_aura)
            if (precision_aura > 0) then
                strcat(tempFloatStr_aura, ".")
                fracPart_aura ← floatValue_aura - (real)intPart_aura
                p_aura traversal [0...precision_aura-1]
                fracPart_aura ← fracPart_aura * 10
                digit_aura ← (integer)fracPart_aura
                digitChar_aura[0] ← (character)(digit_aura + '0')
                digitChar_aura[1] ← '\0'
                strcat(tempFloatStr_aura, digitChar_aura)
                fracPart_aura ← fracPart_aura - digit_aura
            if (strlen(tempFloatStr_aura) < sizeof(auraStr)) then
                strcpy(auraStr, tempFloatStr_aura)
            else
                strcpy(auraStr, "BIG_F")
                row[0] ← idStr
                row[1] ← foundDoctor.username
                row[2] ← auraStr
                printTableRow(row, widths, 3)
            else { Search by name }
                i traversal [0...hospital.doctors.nEff-1]
                doctor ← &hospital.doctors.elements[i]

```

```

if (containsSubstring(doctor.username, query)) then
    found ← true
    { Manual integerToString for doctor.id }

    k_id_doc_name ← 0
    if (doctor.id = 0) then
        tempIdConv_doc_name[k_id_doc_name] ← '0'
        k_id_doc_name ← k_id_doc_name + 1
    else
        tempVal_id_doc_name ← doctor.id
        isNeg_id_doc_name ← false
        if (tempVal_id_doc_name < 0) then
            isNeg_id_doc_name ← true
            tempVal_id_doc_name ← -tempVal_id_doc_name
        numDigits_id_doc_name ← 0
        valCopy_id_doc_name ← tempVal_id_doc_name
        if (valCopy_id_doc_name = 0 and doctor.id ≠ 0)
            then
                numDigits_id_doc_name ← 0
                else if (valCopy_id_doc_name = 0) then
                    numDigits_id_doc_name ← 1
                else
                    while (valCopy_id_doc_name > 0) do
                        valCopy_id_doc_name ← valCopy_id_doc_name


div 10


                        numDigits_id_doc_name ←
                        numDigits_id_doc_name + 1
                        k_id_doc_name ← numDigits_id_doc_name
                        if (tempVal_id_doc_name = 0 and doctor.id ≠ 0)
                            then
                                -- do nothing

```

```

    else if (tempVal_id_doc_name = 0) then
        tempIdConv_doc_name[0] ← '0'
    else
        while (tempVal_id_doc_name > 0) do
            k_id_doc_name ← k_id_doc_name - 1
            tempIdConv_doc_name[k_id_doc_name] ←
                (tempVal_id_doc_name mod 10) + '0'
            tempVal_id_doc_name ← tempVal_id_doc_name
        div 10
        k_id_doc_name ← numDigits_id_doc_name
        if (isNeg_id_doc_name) then
            m traversal [k_id_doc_name...0]
            tempIdConv_doc_name[m] ←
                tempIdConv_doc_name[m-1]
            tempIdConv_doc_name[0] ← '-'
            k_id_doc_name ← k_id_doc_name + 1
            tempIdConv_doc_name[k_id_doc_name] ← '\0'
        if (k_id_doc_name < sizeof(idStr)) then
            strcpy(idStr, tempIdConv_doc_name)
        else
            strcpy(idStr, "BIG")
        { Manual floatToString for doctor.aura (precision 1) }
        floatValue_aura_name ← doctor.aura
        precision_aura_name ← 1
        auraStr[0] ← '\0'
        tempFloatStr_aura_name[0] ← '\0'
        if (floatValue_aura_name < 0.0) then
            floatValue_aura_name ← -floatValue_aura_name
            strcpy(tempFloatStr_aura_name, "-")
        intPart_aura_name ← (integer)floatValue_aura_name

```

```

{ Manual int-to-string for intPart_aura_name }

k_int_aura_name ← 0
if (intPart_aura_name = 0) then
    tempIntConv_aura_name[k_int_aura_name] ← '0'
    k_int_aura_name ← k_int_aura_name + 1
else
    tempVal_int_aura_name ← intPart_aura_name
    numDigits_int_aura_name ← 0
    valCopy_int_aura_name ← tempVal_int_aura_name
    if (valCopy_int_aura_name = 0) then
        numDigits_int_aura_name ← 1
    else
        while (valCopy_int_aura_name > 0) do
            valCopy_int_aura_name ←
                valCopy_int_aura_name div 10
            numDigits_int_aura_name ←
                numDigits_int_aura_name + 1
            k_int_aura_name ← numDigits_int_aura_name
            if (tempVal_int_aura_name = 0) then
                tempIntConv_aura_name[0] ← '0'
            else
                while (tempVal_int_aura_name > 0) do
                    k_int_aura_name ← k_int_aura_name - 1
                    tempIntConv_aura_name[k_int_aura_name] ←
                        (tempVal_int_aura_name mod 10) + '0'
                    tempVal_int_aura_name ←
                        tempVal_int_aura_name div 10
                k_int_aura_name ← numDigits_int_aura_name
                tempIntConv_aura_name[k_int_aura_name] ← '\0'
                strcpy(intStr_aura_name, tempIntConv_aura_name)

```

```

        strcat(tempFloatStr_aura_name, intStr_aura_name)
        if (precision_aura_name > 0) then
            strcat(tempFloatStr_aura_name, ".")
            fracPart_aura_name ← floatValue_aura_name -
            (real)intPart_aura_name
            p_aura_name traversal [0...precision_aura_name-1]
            fracPart_aura_name ← fracPart_aura_name * 10
            digit_aura_name ← (integer)fracPart_aura_name
            digitChar_aura_name[0] ←
            (character)(digit_aura_name + '0')
            digitChar_aura_name[1] ← '\0'
            strcat(tempFloatStr_aura_name,
            digitChar_aura_name)
            fracPart_aura_name ← fracPart_aura_name -
            digit_aura_name
            if (strlen(tempFloatStr_aura_name) < sizeof(auraStr))
            then
                strcpy(auraStr, tempFloatStr_aura_name)
            else
                strcpy(auraStr, "BIG_F")
                row[0] ← idStr
                row[1] ← doctor.username
                row[2] ← auraStr
                printTableRow(row, widths, 3)
            printTableBorder(widths, 3, 3)
            if (not found) then
                output("Dokter dengan ", byId ? "ID" : "nama", " ", query, ""
                tidak ditemukan.\n")

```

{ Fungsi Helper }

```
function localCaseInsensitivestrcmp(input s1 : string, input s2 : string) → integer
{ I.S. Menerima dua string.
  F.S. Mengembalikan 0 jika string sama (tidak peka huruf besar/kecil),
       nilai negatif jika s1 kurang dari s2, nilai positif jika s1 lebih
       dari s2.
}
```

```
function customBinarySearchDoctors(input key : pointer to Doctor, input
base : array of Doctor, input num : integer, input compare : pointer to
function) → pointer to Doctor
{ I.S. Menerima key Doctor yang akan dicari, array Doctor yang
  sudah terurut, jumlah elemen, dan fungsi pembanding.
  F.S. Mengembalikan pointer ke Doctor yang ditemukan, atau NIL
       jika tidak ditemukan.
}
```

```
function customBinarySearchPatients(input key : pointer to Patient,
input base : array of Patient, input num : integer, input compare :
pointer to function) → pointer to Patient
{ I.S. Menerima key Patient yang akan dicari, array Patient yang
  sudah terurut, jumlah elemen, dan fungsi pembanding.
  F.S. Mengembalikan pointer ke Patient yang ditemukan, atau NIL
       jika tidak ditemukan.
}
```

```
function customBinarySearchUsers(input key : pointer to User, input base
: array of User, input num : integer, input compare : pointer to
function) → pointer to User
```

```
{ I.S. Menerima key User yang akan dicari, array User yang sudah terurut, jumlah elemen, dan fungsi pembanding.  
F.S. Mengembalikan pointer ke User yang ditemukan, atau NIL jika tidak ditemukan.  
}
```

```
function compareDoctorsById(input a : pointer to any, input b : pointer to any) → integer  
{ I.S. Menerima dua pointer ke struktur Doctor.  
F.S. Mengembalikan -1 jika id Doctor a kurang dari Doctor b, 1 jika lebih besar, dan 0 jika sama.  
}
```

```
function comparePatientsById(input a : pointer to any, input b : pointer to any) → integer  
{ I.S. Menerima dua pointer ke struktur Patient.  
F.S. Mengembalikan -1 jika id Patient a kurang dari Patient b, 1 jika lebih besar, dan 0 jika sama.  
}
```

```
function compareUsersById(input a : pointer to any, input b : pointer to any) → integer  
{ I.S. Menerima dua pointer ke struktur User.  
F.S. Mengembalikan -1 jika id User a kurang dari User b, 1 jika lebih besar, dan 0 jika sama.  
}
```

**Notes:** Fitur F08 terhubung dengan fungsi B07 - Search Suggestion

F09 - Lihat Antrian

```
{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
```

```

type Role: enumerasi [MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< layout : Layout;
  doctors : DoctorList;
  patients : PatientList;
  queues : QueueList >

type Layout :
< elements : array [1..rowMax][1..colMax] of Room;
  rowEff : integer;
  colEff : integer >

type Room :
< code : string;
  capacity : integer;
  doctorId : integer;
  patientInRoom : PatientInRoomList
>

type PatientInRoomList :
< patientId : array [1..N] of integer;
  nEff : integer >

```

```

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer >

type Doctor :
< id : integer;
  username : string >

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer >

type Patient :
< id : integer;
  username : string >

type QueueList :
< queues : array [1..N] of Queue;
  nRooms : integer;
  capacity : integer >

type Queue :
< roomCode : string;
  head : pointer to QueueNode >

type QueueNode :
< info : QueueInfo;
  next : pointer to QueueNode >

type QueueInfo :

```

```

< patientId : integer >

{ Definisi function & procedure yang di-import dari file source code
lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

procedure displayLayout(input hospital : pointer to Hospital, input
session : pointer to Session, input printHeaderFlag : boolean)
{ Menampilkan denah rumah sakit. }

function integerToString(input num : integer, output str : string, input
size : integer) → boolean
{ Mengonversi integer ke string. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak baris tabel. }

```

```
function isQueueEmpty(input q : pointer to Queue) → boolean
{ Memeriksa apakah antrian kosong. }
```

#### KAMUS LOKAL

```
hasRoomsWithDoctor : boolean
i, j, k, l, m : integer
room : pointer to Room
header : character [100]
capacityStr : character [20]
capacity : integer
row1_capacity : array [1..2] of string
widths_capacity : array [1..2] of integer
doctorName : string
row1_doctor : array [1..2] of string
widths_doctor : array [1..2] of integer
found : boolean
foundQueue : boolean
currentQueue : pointer to Queue
currentNode : pointer to QueueNode
position : integer
patientFoundInList : boolean
```

```
procedure displayQueue(input hospital: pointer to Hospital, input
session: pointer to Session)
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Menampilkan status antrian untuk setiap ruangan yang
memiliki dokter
  dan pasien di dalamnya/antri, atau pesan error jika akses
ditolak atau denah kosong.
}
```

## ALGORITMA

```
depend on (true) :  
    (hospital = NIL or session = NIL) :  
        printError("Struktur rumah sakit atau sesi tidak valid!")  
        return  
    (not session.isLoggedIn or session.role ≠ MANAGER) :  
        printError("Akses ditolak! Hanya Manajer yang dapat  
melihat antrian.")  
        return  
    (hospital.layout.rowEff ≤ 0 or hospital.layout.colEff ≤ 0) :  
        printError("Denah rumah sakit kosong!")  
        return  
    (otherwise) :  
        printHeader("Status Antrian")  
        displayLayout(hospital, session, false)  
  
        hasRoomsWithDoctor ← false  
  
        i traversal [0...hospital.layout.rowEff-1]  
            j traversal [0...hospital.layout.colEff-1]  
                room ← &hospital.layout.elements[i][j]  
                if (room.doctorId ≠ -1) then  
                    hasRoomsWithDoctor ← true  
                    header[0] ← '\0'  
                    strcat(header, "Antrian Ruangan ")  
                    strcat(header, room.code)  
                    output("\n")  
                    printHeader(header)  
  
        capacity ← room.capacity
```

```

        if (not integerToString(capacity, capacityStr,
20)) then
            printError("Kapasitas ruangan tidak
valid!")
            return
            strcat(capacityStr, " orang")

            row1_capacity[0] ← "Kapasitas"
            row1_capacity[1] ← capacityStr
            widths_capacity[0] ← 15
            widths_capacity[1] ← 10

            printTableBorder(widths_capacity, 2, 1)
            printTableRow(row1_capacity, widths_capacity,
2)

            doctorName ← NIL
            k traversal [0...hospital.doctors.nEff-1]
            if (hospital.doctors.elements[k].id =
room.doctorId) then
                doctorName ←
hospital.doctors.elements[k].username
                stop

            if (doctorName = NIL or hospital.doctors.nEff
≤ 0) then
                output(" Tidak ada dokter")
            else
                row1_doctor[0] ← "Dokter"
                row1_doctor[1] ← doctorName

```

```

        widths_doctor[0] ← 15
        widths_doctor[1] ← 10
        printTableBorder(widths_doctor, 2, 2)
        printTableRow(row1_doctor, widths_doctor,
2)
        printTableBorder(widths_doctor, 2, 3)

        output("Pasien di dalam ruangan:")
        if (room.patientInRoom.nEff = 0 or
hospital.patients.nEff ≤ 0) then
            output(" Tidak ada pasien")
        else
            k traversal
            [0...room.patientInRoom.nEff-1]
                found ← false
                l traversal
                [0...hospital.patients.nEff-1]
                    if
(hospital.patients.elements[l].id =
room.patientInRoom.patientId[k]) then
                        output(" ", k + 1, ". ",
hospital.patients.elements[l].username)
                        found ← true
                        stop
                    if (not found) then
                        output(" ", k + 1, ". Pasien
tidak ditemukan")

        output("Pasien di antrian:")
        foundQueue ← false

```

```

k traversal [0...hospital.queues.nRooms-1]
if
(strcmp(hospital.queues.queues[k].roomCode, room.code) = 0) then
    foundQueue ← true
    currentQueue ←
&hospital.queues.queues[k]

if (isQueueEmpty(currentQueue) or
hospital.patients.nEff ≤ 0) then
    output(" Tidak ada pasien di
antrian")

else
    currentNode ← currentQueue.head
    position ← 1
    while (currentNode ≠ NIL) do
        patientFoundInList ← false
        m traversal
[0...hospital.patients.nEff-1]
        if
(hospital.patients.elements[m].id = currentNode.info.patientId)
then
        output(" ", position,
". ", hospital.patients.elements[m].username)
        patientFoundInList ←
true
        stop
        if (not patientFoundInList)
then

```

```

output(" ", position, ".  
Pasien ID ", currentNode.info.patientId, " tidak ditemukan dalam  
daftar pasien utama")  
  
currentNode ←  
currentNode.next  
  
position ← position + 1  
  
stop  
if (not foundQueue) then  
    output(" Tidak ada pasien di antrian")  
  
if (not hasRoomsWithDoctor) then  
    output("Tidak ada ruangan dengan dokter saat ini.")
```

## F10 - Tambah Dokter

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< users : UserList;
  doctors : DoctorList;
  layout : Layout >

type UserList :
< elements : array [1..N] of User;
  nEff : integer;
  capacity : integer >

type User :
```

```

< id : integer;
  username : string;
  password : PasswordData;
  role : Role >

type PasswordData :
  < encryptedContent : string >

type DoctorList :
  < elements : array [1..N] of Doctor;
    nEff : integer;
    capacity : integer >

type Doctor :
  < id : integer;
    username : string;
    specialization : string;
    aura : integer;
    bananaRich : real;
    room : string >

type Layout :
  < elements : array [1..rowMax][1..colMax] of Room;
    rowEff : integer;
    colEff : integer >

type Room :
  < code : string;
    capacity : integer;
    doctorId : integer;
    patientInRoom : PatientInRoomList
  >

type PatientInRoomList :
  < patientId : array [1..N] of integer;
    nEff : integer >

```

{ Definisi function & procedure yang di-import dari file source code lain }

function strlen(input s : string) → integer

```

{ Mencari panjang string s. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function isValidUsername(input username : string) → boolean
{ Memvalidasi format username. }

function isUsernameTaken(input hospital : pointer to Hospital, input
username : string) → boolean
{ Memeriksa apakah username sudah terdaftar. }

function enigmaEncrypt(input password : string, output encryptedPassword
: string, input bufferSize : integer) → boolean
{ Mengenkripsi password. }

function isValidRoomCode(input hospital : pointer to Hospital, input
roomCode : string) → boolean
{ Memvalidasi format kode ruangan dan keberadaannya di denah. }

function realloc(input ptr : pointer to void, input size : integer) →
pointer to void
{ Mengubah ukuran blok memori yang dialokasikan. }

{ Prosedur utama }

function addDoctor(input hospital: pointer to Hospital, input session:
pointer to Session, input inputUsername: String, input password: String,
input specialization: String) → boolean
{ I.S. Menerima pointer ke struktur Hospital, Session, username,
password, dan spesialisasi dokter.
}

```

F.S. Mengembalikan true jika dokter berhasil ditambahkan; false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}

#### KAMUS LOKAL

```
newCapacity : integer
tempUsers : pointer to User
newDoctorCapacity : integer
tempDoctors : pointer to Doctor
maxId : integer
newDoctorId : integer
newUser : pointer to User
newDoctor : pointer to Doctor
successMsg : character
i : integer
```

#### ALGORITMA

```
depend on (true) :
    (hospital = NIL or session = NIL or inputUsername = NIL or
password = NIL or specialization = NIL) :
        printError("Struktur rumah sakit, sesi, atau input tidak
valid!")
        → false
    (not session.isLoggedIn or session.role ≠ MANAGER) :
        printError("Akses ditolak! Hanya Manajer yang dapat
menambah dokter.")
        → false
    (not isValidUsername(inputUsername)) :
        printError("Username tidak valid! Hanya boleh berisi
huruf, angka, spasi, atau underscore.")
        → false
    (isUsernameTaken(hospital, inputUsername)) :
        printError("Username sudah terdaftar!")
        → false
    (strlen(password) < 6) :
        printError("Password harus minimal 6 karakter!")
        → false
    (not isValidUsername(specialization)) :
```

```

printError("Spesialisasi tidak valid! Gunakan huruf,
angka, atau underscore.")

→ false

(otherwise) :
    if (hospital.users.nEff ≥ hospital.users.capacity or
hospital.doctors.nEff ≥ hospital.doctors.capacity) then
        newCapacity ← hospital.users.capacity * 2
        tempUsers ← (pointer to User)
realloc(hospital.users.elements, newCapacity * sizeof(User))
    if (tempUsers = NIL) then
        printError("Gagal mengalokasi memori untuk
ekspansi pengguna!")
    → false
        hospital.users.elements ← tempUsers
        hospital.users.capacity ← newCapacity

        newDoctorCapacity ← hospital.doctors.capacity * 2
        tempDoctors ← (pointer to Doctor)
realloc(hospital.doctors.elements, newDoctorCapacity *
sizeof(Doctor))
    if (tempDoctors = NIL) then
        printError("Gagal mengalokasi memori untuk
ekspansi dokter!")
    → false
        hospital.doctors.elements ← tempDoctors
        hospital.doctors.capacity ← newDoctorCapacity

    maxId ← 0
    i traversal [0...hospital.users.nEff-1]
    if (hospital.users.elements[i].id > maxId) then
        maxId ← hospital.users.elements[i].id
    newDoctorId ← maxId + 1

    newUser ← &hospital.users.elements[hospital.users.nEff]
    newUser.id ← newDoctorId
    strcpy(newUser.username, inputUsername)
    if (not enigmaEncrypt(password,
newUser.password.encryptedContent, 100)) then
        printError("Gagal mengenkripsi password!")

```

```

→ false
newUser.role ← DOCTOR

newDoctor ←
&hospital.doctors.elements[hospital.doctors.nEff]
newDoctor.id ← newDoctorId
strcpy(newDoctor.username, inputUsername)
strcpy(newDoctor.specialization, specialization)
newDoctor.aura ← 0
newDoctor.bananaRich ← 100.0
newDoctor.room[0] ← '\0'

hospital.users.nEff ← hospital.users.nEff + 1
hospital.doctors.nEff ← hospital.doctors.nEff + 1

successMsg[0] ← '\0'
strcat(successMsg, "Dokter ")
strcat(successMsg, inputUsername)
strcat(successMsg, " berhasil ditambahkan!")
printSuccess(successMsg)
→ true

```

{ Fungsi Helper }

```

function assignDoctor(input hospital: pointer to Hospital, input
session: pointer to Session, input username: String, input roomCode:
String) → boolean
{ I.S. Menerima pointer ke struktur Hospital, Session, username
dokter, dan kode ruangan.
F.S. Mengembalikan true jika dokter berhasil ditugaskan ke
ruangan; false jika gagal.
Jika gagal, pesan error yang sesuai ditampilkan.
}

```

## F11 - Diagnosis

```
{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
    < userId : integer;
        username : string;
        role : Role;
        isLoggedIn : boolean >

type Hospital :
    < doctors : DoctorList;
        layout : Layout;
        patients : PatientList;
        diseases : DiseaseList >

type DoctorList :
    < elements : array [1..N] of Doctor;
        nEff : integer >

type Doctor :
    < id : integer;
        username : string;
        room : string >

type Layout :
    < elements : array [1..rowMax][1..colMax] of Room;
        rowEff : integer;
        colEff : integer >
```

```

type Room :
  < code : string;
    patientInRoom : PatientInRoomList >

type PatientInRoomList :
  < patientId : array [1..N] of integer;
    nEff : integer >

type PatientList :
  < elements : array [1..N] of Patient;
    nEff : integer >

type Patient :
  < id : integer;
    username : string;
    queueRoom : string;
    diagnosedStatus : boolean;
    disease : string;
    bodyTemperature : real;
    systolicBloodPressure : integer;
    diastolicBloodPressure : integer;
    heartRate : integer;
    oxygenSaturation : integer;
    bloodSugarLevel : real;
    weight : real;
    height : real;
    cholesterolLevel : real;
    platelets : integer >

type DiseaseList :

```

```

< elements : array [1..N] of Disease;
  nEff : integer >

type Disease :
  < name : string;
    bodyTemperatureMin : real;
    bodyTemperatureMax : real;
    systolicBloodPressureMin : integer;
    systolicBloodPressureMax : integer;
    diastolicBloodPressureMin : integer;
    diastolicBloodPressureMax : integer;
    heartRateMin : integer;
    heartRateMax : integer;
    oxygenSaturationMin : integer;
    oxygenSaturationMax : integer;
    bloodSugarLevelMin : real;
    bloodSugarLevelMax : real;
    weightMin : real;
    weightMax : real;
    heightMin : real;
    heightMax : real;
    cholesterolLevelMin : real;
    cholesterolLevelMax : real;
    plateletsMin : integer;
    plateletsMax : integer >

```

```

{ Definisi function & procedure yang di-import dari file source code
lain }

function strcmp(input s1 : string, input s2 : string) → integer
  { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

```

```

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function diagnosePatient(input hospital: pointer to Hospital, input
session: pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Mengembalikan true jika diagnosis berhasil dan status
pasien diperbarui; false jika gagal.
  Jika gagal, pesan error yang sesuai ditampilkan.
}

```

#### KAMUS LOKAL

```

doctorIdx : integer
doctor : pointer to Doctor
doctorRoom : pointer to Room
patientId : integer
patientIdx : integer
patient : pointer to Patient
diseaseStr : character [50]
d : pointer to Disease
i, j : integer

```

## ALGORITMA

```
depend on (true) :  
    (hospital = NIL or session = NIL) :  
        printError("Struktur rumah sakit atau sesi tidak valid!")  
        → false  
    (not session.isLoggedIn or session.role ≠ DOCTOR) :  
        printError("Akses ditolak! Hanya Dokter yang dapat  
mendiagnosis.")  
        → false  
    (otherwise) :  
        doctorIdx ← -1  
        i traversal [0...hospital.doctors.nEff-1]  
        if (strcmp(hospital.doctors.elements[i].username,  
session.username) = 0) then  
            doctorIdx ← i  
            stop  
  
if (doctorIdx = -1) then  
    printError("Dokter tidak ditemukan dalam daftar!")  
    → false  
  
doctor ← &hospital.doctors.elements[doctorIdx]  
  
if (doctor.room[0] = '\0') then  
    printError("Dokter tidak ditugaskan ke ruangan  
manapun!")  
    → false  
  
doctorRoom ← NIL  
i traversal [0...hospital.layout.rowEff-1]
```

```

j traversal [0...hospital.layout.colEff-1]
    if (strcmp(hospital.layout.elements[i][j].code,
doctor.room) = 0) then
        doctorRoom ← &hospital.layout.elements[i][j]
        stop
    if (doctorRoom ≠ NIL) then
        stop

if (doctorRoom = NIL or doctorRoom.patientInRoom.nEff = 0)
then
    printError("Ruangan dokter ini tidak memiliki pasien
yang mengantri.")
    → false

patientId ← doctorRoom.patientInRoom.patientId[0]
patientIdx ← -1
i traversal [0...hospital.patients.nEff-1]
    if (hospital.patients.elements[i].id = patientId) then
        patientIdx ← i
        stop

if (patientIdx = -1) then
    printError("Pasien tidak ditemukan!")
    → false

patient ← &hospital.patients.elements[patientIdx]

depend on (true) :
    (patient.diagnosedStatus) :
        printError("Pasien sudah didiagnosa sebelumnya!")

```

```

→ false
(strcmp(patient.queueRoom, doctor.room) ≠ 0) :
    printError("Pasien tidak berada di antrian ruangan
dokter!")

→ false
(otherwise) :
    strcpy(diseaseStr, "Tidak terdeteksi")

if (hospital.diseases.nEff > 0) then
    i traversal [0...hospital.diseases.nEff-1]
        d ← &hospital.diseases.elements[i]
        if (patient.bodyTemperature ≥
d.bodyTemperatureMin and patient.bodyTemperature ≤
d.bodyTemperatureMax and
            patient.systolicBloodPressure ≥
d.systolicBloodPressureMin and patient.systolicBloodPressure ≤
d.systolicBloodPressureMax and
            patient.diastolicBloodPressure ≥
d.diastolicBloodPressureMin and patient.diastolicBloodPressure ≤
d.diastolicBloodPressureMax and
            patient.heartRate ≥ d.heartRateMin
and patient.heartRate ≤ d.heartRateMax and
            patient.oxygenSaturation ≥
d.oxygenSaturationMin and patient.oxygenSaturation ≤
d.oxygenSaturationMax and
            patient.bloodSugarLevel ≥
d.bloodSugarLevelMin and patient.bloodSugarLevel ≤
d.bloodSugarLevelMax and
            patient.weight ≥ d.weightMin and
patient.weight ≤ d.weightMax and

```

```

                patient.height ≥ d.heightMin and
patient.height ≤ d.heightMax and
                                patient.cholesterolLevel ≥
d.cholesterolLevelMin and patient.cholesterolLevel ≤
d.cholesterolLevelMax and
                                patient.platelets ≥ d.plateletsMin
and patient.platelets ≤ d.plateletsMax) then

                                strcpy(diseaseStr, d.name)
patient.disease[0] ← '\0'
strcat(patient.disease, diseaseStr)
patient.diagnosedStatus ← true
stop

if (strcmp(diseaseStr, "Tidak terdeteksi") = 0)
then
                                patient.diagnosedStatus ← true
                                printSuccess("Pasien tidak terdiagnosis
penyakit apapun!")
else
                                output(patient.username, " terdiagnosa
penyakit ", diseaseStr, "!")
→ true

```

F12 - Ngobatin

```
{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }
type Role: enumerasi [PATIENT, DOCTOR, MANAGER]
```

```

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< doctors : DoctorList;
  layout : Layout;
  patients : PatientList;
  diseases : DiseaseList >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer >

type Doctor :
< id : integer;
  username : string;
  room : string >

type Layout :
< elements : array [1..rowMax][1..colMax] of Room;
  rowEff : integer;
  colEff : integer >

type Room :
< code : string;
  patientInRoom : PatientInRoomList >

```

```

type PatientInRoomList :
    < patientId : array [1..N] of integer;
        nEff : integer >

type PatientList :
    < elements : array [1..N] of Patient;
        nEff : integer >

type Patient :
    < id : integer;
        username : string;
        queueRoom : string;
        diagnosedStatus : boolean;
        disease : string;
        bodyTemperature : real;
        systolicBloodPressure : integer;
        diastolicBloodPressure : integer;
        heartRate : integer;
        oxygenSaturation : integer;
        bloodSugarLevel : real;
        weight : real;
        height : real;
        cholesterolLevel : real;
        platelets : integer >

type DiseaseList :
    < elements : array [1..N] of Disease;
        nEff : integer >

type Disease :

```

```

< name : string;
  bodyTemperatureMin : real;
  bodyTemperatureMax : real;
  systolicBloodPressureMin : integer;
  systolicBloodPressureMax : integer;
  diastolicBloodPressureMin : integer;
  diastolicBloodPressureMax : integer;
  heartRateMin : integer;
  heartRateMax : integer;
  oxygenSaturationMin : integer;
  oxygenSaturationMax : integer;
  bloodSugarLevelMin : real;
  bloodSugarLevelMax : real;
  weightMin : real;
  weightMax : real;
  heightMin : real;
  heightMax : real;
  cholesterolLevelMin : real;
  cholesterolLevelMax : real;
  plateletsMin : integer;
  plateletsMax : integer >

```

```

{ Definisi function & procedure yang di-import dari file source code
lain }

function strcmp(input s1 : string, input s2 : string) → integer
  { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

procedure printError(input message : string)
  { Mencetak pesan error. }

procedure printSuccess(input message : string)

```

```

{ Mencetak pesan sukses. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function treatPatient(input hospital: pointer to Hospital, input
session: pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Mengembalikan true jika pasien berhasil diberi resep obat
berdasarkan penyakit hasil diagnosa sebelumnya; false jika gagal.
  Jika gagal, pesan error yang sesuai ditampilkan.
}

```

#### KAMUS LOKAL

```

doctorIdx : integer
doctor : pointer to Doctor
doctorRoom : pointer to Room
patientId : integer
patientIdx : integer
patient : pointer to Patient
diseaseId : integer
medicationCount : integer
tempPrescriptions : pointer to MedicationPrescription
prescribedMedications : pointer to MedicationList
tempCount : integer
med : pointer to Medication
widths : array [0..3] of integer
headers : array [0..3] of string
idStr : character [5]
numberStr : character [10]
errorMsg : character [50]
i, j : integer

```

## ALGORITMA

```
depend on (true) :  
    (hospital = NIL or session = NIL) :  
        printError("Struktur rumah sakit atau sesi tidak valid!")  
        → false  
    (not session.isLoggedIn or session.role ≠ DOCTOR) :  
        printError("Akses ditolak! Hanya Dokter yang dapat  
mendiagnosis.")  
        → false  
    (otherwise) :  
        doctorIdx ← -1  
        i traversal [0...hospital.doctors.nEff-1]  
        if (strcmp(hospital.doctors.elements[i].username,  
session.username) = 0) then  
            doctorIdx ← i  
            stop  
  
if (doctorIdx = -1) then  
    printError("Dokter tidak ditemukan dalam daftar!")  
    → false  
  
doctor ← &hospital.doctors.elements[doctorIdx]  
  
if (doctor.room[0] = '\0') then  
    printError("Dokter tidak ditugaskan ke ruangan  
manapun!")  
    → false  
  
doctorRoom ← NIL  
i traversal [0...hospital.layout.rowEff-1]
```

```

j traversal [0...hospital.layout.colEff-1]
    if (strcmp(hospital.layout.elements[i][j].code,
doctor.room) = 0) then
        doctorRoom ← &hospital.layout.elements[i][j]
        stop
    if (doctorRoom ≠ NIL) then
        stop

if (doctorRoom = NIL or doctorRoom.patientInRoom.nEff = 0)
then
    printError("Tidak ada pasien yang berada di dalam
ruangan.")
    → false

patientId ← doctorRoom.patientInRoom.patientId[0]
patientIdx ← -1
i traversal [0...hospital.patients.nEff-1]
    if (hospital.patients.elements[i].id = patientId) then
        patientIdx ← i
        stop

if (patientIdx = -1) then
    printError("Pasien tidak ditemukan!")
    → false
patient ← &hospital.patients.elements[patientIdx]
if (strcmp(patient.disease, "Tidak terdeteksi") = 0) then
    printSuccess("Pasien tidak menderita penyakit apapun!")
    patient.treatedStatus ← true
    → true
else if (not patient.diagnosedStatus) then

```

```

        printError("Pasien belum mendapat diagnosa dari dokter!")

        → false

else if (patient.treatedStatus) then

        printError("Pasien sudah diberikan resep obat!")

        → false

else if (strcmp(patient.queueRoom, doctor.room) ≠ 0) then

        printError("Pasien tidak berada di antrian ruangan

dokter!")

        → false

else if (patient.id ≠ patientId) then

        printError("Pasien ini tidak berada di depan antrian

untuk ruangan dokter ini.")

        → false


diseaseId ← -1

for i traversal [0...hospital.diseases.nEff-1]

    if (strcmp(hospital.diseases.elements[i].name,

patient.disease) = 0) then

        diseaseId ← hospital.diseases.elements[i].id

        stop (true)

    if (diseaseId = -1) then

        printError("Penyakit pasien tidak ditemukan dalam

database!")

        → false

medicationCount ← 0

for i traversal [0...hospital.prescriptions.nEff-1]

    if (hospital.prescriptions.elements[i].diseaseId =

diseaseId) then

        medicationCount ← medicationCount + 1

```

```

tempPrescriptions ← NULL
prescribedMedications.elements ← NULL
prescribedMedications.capacity ← 0
prescribedMedications.nEff ← 0

if (medicationCount > 0) then
    alokasi (tempPrescriptions)
    if (tempPrescriptions = NULL) then
        printError("Gagal mengalokasi memori untuk resep
sementara!")
    → false

tempCount ← 0
for i traversal [0...hospital.prescriptions.nEff-1]
    if (hospital.prescriptions.elements[i].diseaseId
= diseaseId) then
        tempPrescriptions[tempCount] ←
hospital.prescriptions.elements[i]
        tempCount ← tempCount + 1

for i traversal [0...medicationCount - 2]
    for j traversal [0...medicationCount - i - 2]
        if (tempPrescriptions[j].doseOrder >
tempPrescriptions[j + 1].doseOrder) then
            temp ← tempPrescriptions[j]
            tempPrescriptions[j] ←
tempPrescriptions[j + 1]
            tempPrescriptions[j + 1] ← temp

```

```

alokasi (prescribedMedications.elements)
if (prescribedMedications.elements = NULL) then
    dealokasi (tempPrescriptions)
    printError("Gagal mengalokasi memori untuk daftar
obat!")
    → false

for i traversal [0...medicationCount-1]
    prescribedMedications.elements[i].id ←
tempPrescriptions[i].medicationId
    prescribedMedications.nEff ←
prescribedMedications.nEff + 1

    dealokasi (tempPrescriptions)

if (prescribedMedications.nEff > 0) then
    output(COLOR_GREEN, "Dokter sedang mengobati
pasien!\n", COLOR_RESET)
    output("Pasien memiliki penyakit ", COLOR_YELLOW,
patient.disease, COLOR_RESET, "\n")
    output("\nObat yang harus diberikan:\n")

    widths[0] ← 5
    widths[1] ← 30
    header[0] ← "No."
    header[1] ← "Nama Obat"
    printTableBorder(widths, 2, 1)
    printTableRow(header, widths, 2)

```

```

printTableBorder(widths, 2, 2)

for i traversal [0...prescribedMedications.nEff-1]
    med < NULL
    for j traversal [0...hospital.medications.nEff-1]
        if (hospital.medications.elements[j].id =
prescribedMedications.elements[i].id) then
            med < &hospital.medications.elements[j]
            stop (true)
        if (med ≠ NULL) then
            strcpy(numberStr, "")
            integerToString(i + 1, numberStr,
sizeof(numberStr))
            strcat(numberStr, ".")
            row[0] < numberStr
            row[1] < med.name
            printTableRow(row, widths, 2)
        else
            strcpy(errorMsg, "")
            strcat(errorMsg, "Obat tidak ditemukan untuk
ID: ")

integerToString(prescribedMedications.elements[i].id, idStr,
sizeof(idStr))
            strcat(errorMsg, idStr)
            printError(errorMsg)

printTableBorder(widths, 2, 3)
patient.medicationsPrescribed.nEff < 0

```

```

        for i traversal [0...prescribedMedications.nEff-1]

    patient.medicationsPrescribed.medicationId[patient.medicationsPre
scribed.nEff] ← prescribedMedications.elements[i].id
            patient.medicationsPrescribed.nEff ←
    patient.medicationsPrescribed.nEff + 1

            dealokasi (prescribedMedications.elements)
    patient.treatedStatus ← true
            → true

```

F13 - Aku boleh pulang ga, dok?

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< patients : PatientList;
  doctors : DoctorList;
  treatmentHistory : TreatmentHistoryList;
  queues : QueueList >

type PatientList :
< elements : array [1..N] of Patient;

```

```

nEff : integer >

type Patient :
< id : integer;
  diagnosedStatus : boolean;
  treatedStatus : boolean;
  medicationsPrescribed : MedicationList;
  medicationsTaken : MedicationStack;
  queueRoom : string;
  queuePosition : integer >

type MedicationList :
< medicationId : pointer to integer;
  nEff : integer >

type MedicationStack :
< medicationId : pointer to integer;
  top : integer >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer >

type Doctor :
< id : integer;
  aura : integer >

type TreatmentHistoryList :
< elements : array [1..N] of TreatmentHistory;
  nEff : integer >

```

```

type TreatmentHistory :
< patientId : integer;
  doctorId : integer >

type QueueList :
< queues : array [1..N] of Queue;
  capacity : integer >

type Queue :
< roomCode : string;
  head : pointer to QueueNode;
  tail : pointer to QueueNode;
  size : integer >

type QueueNode :
< info : QueueInfo;
  next : pointer to QueueNode;
  prev : pointer to QueueNode >

type QueueInfo :
< patientId : integer >

{ Definisi function & procedure yang di-import dari file source code lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

```

```

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
{ Mencetak baris tabel. }

function isQueueEmpty(input q : pointer to Queue) → boolean
{ Memeriksa apakah antrian kosong. }

procedure free(input ptr : pointer to void)
{ Memberikan memori. }

function canGoHome(input hospital: pointer to Hospital, input session:
pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Mengembalikan true jika pasien boleh pulang; false jika
tidak.
  Menampilkan status dan keterangan mengapa pasien
boleh/tidak boleh pulang.
}

```

#### KAMUS LOKAL

patient : pointer to Patient

```

statusStr : string
descriptionStr : string
canGoHomeStatus : boolean
prescribedCount : integer
takenCount : integer
correctOrder : boolean
widths : array [1..2] of integer
headers : array [1..2] of string
row : array [1..2] of string
doctorId : integer
targetQueue : pointer to Queue
current : pointer to QueueNode
prev : pointer to QueueNode
i, j : integer

```

#### ALGORITMA

```

depend on (true) :
(hospital = NIL or session = NIL) :
    printError("Struktur rumah sakit atau sesi tidak valid!")
    → false
(not session.isLoggedIn or session.role ≠ PATIENT) :
    printError("Akses ditolak! Hanya Pasien yang dapat
memeriksa status pulang.")
    → false
(hospital.patients.nEff = 0) :
    printError("Tidak ada pasien terdaftar!")
    → false
(otherwise) :
    patient ← NIL
    i traversal [0...hospital.patients.nEff-1]

```

```

if (hospital.patients.elements[i].id = session.userId)
then

    patient ← &hospital.patients.elements[i]

    stop

if (patient = NIL) then
    printError("Pasien tidak ditemukan!")
    → false

canGoHomeStatus ← false

depend on (true) :
    (not patient.diagnosedStatus) :
        statusStr ← "Belum Diagnosa"
        descriptionStr ← "Anda belum didiagnosa oleh
dokter."
        stop
    (not patient.treatedStatus) :
        statusStr ← "Belum Diberi Obat"
        descriptionStr ← "Anda belum diberikan resep
obat."
        stop
    (otherwise) :
        prescribedCount ←
patient.medicationsPrescribed.nEff
        takenCount ← patient.medicationsTaken.top + 1

if (takenCount ≠ prescribedCount) then
    statusStr ← "Belum Minum Semua Obat"

```

```

descriptionStr ← "Anda belum mengonsumsi
semua obat yang telah diresepkan."
else
    correctOrder ← true
    i traversal [0...takenCount-1]
    if
        (patient.medicationsTaken.medicationId[i] ≠
        patient.medicationsPrescribed.medicationId[i]) then
            correctOrder ← false
            stop

        if (not correctOrder) then
            statusStr ← "Urutan Obat Salah"
            descriptionStr ← "Anda mengonsumsi obat
dengan urutan yang salah."
        else
            statusStr ← "Boleh Pulang"
            descriptionStr ← "Anda telah selesai
menjalani perawatan."
            canGoHomeStatus ← true
            stop

printHeader("Status Pulang")

widths[0] ← 15
widths[1] ← 40
headers[0] ← "Status"
headers[1] ← "Keterangan"

row[0] ← statusStr

```

```

row[1] ← descriptionStr

printTableBorder(widths, 2, 1)
printTableRow(headers, widths, 2)
printTableRow(row, widths, 2)
printTableBorder(widths, 2, 3)

if (canGoHomeStatus) then
    i traversal [0...hospital.treatmentHistory.nEff-1]
        if
(hospital.treatmentHistory.elements[i].patientId = patient.id)
then
    doctorId ←
hospital.treatmentHistory.elements[i].doctorId
    j traversal [0...hospital.doctors.nEff-1]
        if (hospital.doctors.elements[j].id =
doctorId) then
            hospital.doctors.elements[j].aura ←
hospital.doctors.elements[j].aura + 1
            stop
        stop

if (patient.queueRoom[0] ≠ '\0') then
    targetQueue ← NIL
    i traversal [0...hospital.queues.capacity-1]
        if (hospital.queues.queues[i].roomCode[0] ≠
'\0' and strcmp(hospital.queues.queues[i].roomCode,
patient.queueRoom) = 0) then
            targetQueue ← &hospital.queues.queues[i]
            stop

```

```

        if (targetQueue ≠ NIL and not
isQueueEmpty(targetQueue)) then
    current ← targetQueue.head
    prev ← NIL
    while (current ≠ NIL) do
        if (current.info.patientId = patient.id)
then
        if (prev = NIL) then
            targetQueue.head ← current.next
            if (targetQueue.head ≠ NIL) then
                targetQueue.head.prev ← NIL
            else
                targetQueue.tail ← NIL
            else
                prev.next ← current.next
                if (current.next ≠ NIL) then
                    current.next.prev ← prev
                else
                    targetQueue.tail ← prev
                free(current)
                targetQueue.size ← targetQueue.size -
1
                stop
                prev ← current
                current ← current.next

patient.queueRoom[0] ← '\0'
patient.queuePosition ← 0

```

```

        printSuccess("Selamat, Anda boleh pulang!")
        → true

else
        printError("Anda belum boleh pulang!")
        → false

```

#### F14 - Daftar Check-Up

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah
sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
    username : string;
    role : Role;
    isLoggedIn : boolean >

type Hospital :
< patients : PatientList;
    doctors : DoctorList;
    queues : QueueList;
    layout : Layout;
    finance : FinanceData;
    treatmentHistory : TreatmentHistoryList >

type PatientList :
< elements : array [1..N] of Patient;
    nEff : integer >

```

```

type Patient :
  < id : integer;
    username : string;
    queueRoom : string;
    queuePosition : integer;
    bananaRich : real;
    bodyTemperature : real;
    systolicBloodPressure : integer;
    diastolicBloodPressure : integer;
    heartRate : integer;
    oxygenSaturation : integer;
    bloodSugarLevel : real;
    weight : real;
    height : real;
    cholesterolLevel : real;
    platelets : integer >

type DoctorList :
  < elements : array [1..N] of Doctor;
    nEff : integer >

type Doctor :
  < id : integer;
    username : string;
    specialization : string;
    room : string;
    aura : integer;
    checkupCost : real;
    bananaRich : real >

```

```

type QueueList :
    < queues : array [1..N] of Queue;
        nRooms : integer;
        capacity : integer >

type Queue :
    < roomCode : string;
        head : pointer to QueueNode >

type QueueNode :
    < info : QueueInfo;
        next : pointer to QueueNode >

type QueueInfo :
    < patientId : integer >

type Layout :
    < elements : array [1..rowMax][1..colMax] of Room;
        rowEff : integer;
        colEff : integer >

type Room :
    < code : string;
        capacity : integer;
        patientInRoom : PatientInRoomList >

type PatientInRoomList :
    < patientId : array [1..N] of integer;
        nEff : integer >

```

```

type FinanceData :
    < hospitalBalance : real >

type TreatmentHistoryList :
    < elements : array [1..N] of TreatmentHistory;
        nEff : integer;
        capacity : integer >

type TreatmentHistory :
    < patientId : integer;
        doctorId : integer;
        room : string;
        disease : string;
        treatmentStatus : boolean >

{ Definisi function & procedure yang di-import dari file source
code lain }

function strcmp(input s1 : string, input s2 : string) → integer
    { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika
    sama. }

procedure printError(input message : string)
    { Mencetak pesan error. }

procedure printHeader(input title : string)
    { Mencetak judul header. }

procedure printSuccess(input message : string)
    { Mencetak pesan sukses. }

```

```

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths
: array of integer, input numCols : integer)
{ Mencetak baris tabel. }

function integerToString(input num : integer, output str : string,
input size : integer) → boolean
{ Mengonversi integer ke string. Mengembalikan true jika
berhasil. }

function floatToString(input num : real, output str : string,
input size : integer, input precision : integer) → boolean
{ Mengonversi float ke string. Mengembalikan true jika
berhasil. }

function queueSize(input q : pointer to Queue) → integer
{ Mengembalikan ukuran antrian. }

function readValidInt(output value : pointer to integer, input
prompt : string) → boolean
{ Membaca dan memvalidasi input integer. Mengembalikan true
jika valid. }

function enqueue(input q : pointer to Queue, input patientId :
integer) → boolean
{ Menambahkan pasien ke antrian. Mengembalikan true jika
berhasil. }

```

```

procedure initializeQueue(output q : pointer to Queue, input
roomCode : string)
{ Menginisialisasi struktur antrian baru. }

function realloc(input ptr : pointer to void, input size :
integer) → pointer to void
{ Mengubah ukuran blok memori yang dialokasikan. }

function strcpy(output destination : string, input source :
string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source :
string)
{ Menggabungkan string source ke akhir string destination. }

```

```

function registerCheckup(input hospital: pointer to Hospital, input
session: pointer to Session, input healthData: array of real) → boolean
{ I.S. Menerima pointer ke struktur Hospital, Session, dan array data
kesehatan pasien.

F.S. Mengembalikan true jika pendaftaran check-up berhasil; false jika
gagal.

}

```

#### KAMUS LOKAL

```

patientIdx : integer

patient : pointer to Patient

availableDoctorCount : integer

availableDoctors : array [1..N] of pointer to Doctor

doctor : pointer to Doctor

```

```

queueCount : integer
numberStr : character [10]
queueCountStr : character [10]
auraStr : character [10]
checkupCostStr : character [20]
row_displayDoctors : array [1..7] of string
widths_displayDoctors : array [1..7] of integer
headers_displayDoctors : array [1..7] of string
doctorChoice : integer
promptSelectDoctor : character [100]
availableCountStr : character [10]
queueIdx : integer
newQueue : pointer to Queue
queueToCheck : pointer to Queue
selectedRoom : pointer to Room
newElements : pointer to TreatmentHistory
successMsg : character [100]
history : pointer to TreatmentHistory
widths2 : array [1..2] of integer
headers2 : array [1..2] of string
row2 : array [1..2] of string
i, j : integer

```

#### ALGORITMA

```

depend on (true) :
    (hospital = NIL or session = NIL or healthData = NIL) :
        printError("Struktur rumah sakit, sesi, atau data
kesehatan tidak valid!")
        → false
    (not session.isLoggedIn or session.role ≠ PATIENT) :

```

```

        printError("Akses ditolak! Hanya Pasien yang dapat
mendaftar checkup.")
        → false
    (otherwise) :
        patientIdx ← -1
        i traversal [0...hospital.patients.nEff-1]
            if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then
                patientIdx ← i
                stop

            if (patientIdx = -1) then
                printError("Pasien tidak ditemukan!")
                → false

        patient ← &hospital.patients.elements[patientIdx]

        if (patient.queueRoom[0] ≠ '\0') then
            printError("Anda sudah terdaftar dalam antrian!")
            → false

        availableDoctorCount ← 0
        i traversal [0...hospital.doctors.nEff-1]
            doctor ← &hospital.doctors.elements[i]
            if (doctor.room[0] ≠ '\0') then
                availableDoctors[availableDoctorCount] ← doctor
                availableDoctorCount ← availableDoctorCount + 1

        if (availableDoctorCount = 0) then
            printError("Tidak ada dokter yang tersedia saat ini.")

```

```

→ false

output("Daftar Dokter yang Tersedia:")

widths_displayDoctors[0] ← 5
widths_displayDoctors[1] ← 20
widths_displayDoctors[2] ← 15
widths_displayDoctors[3] ← 10
widths_displayDoctors[4] ← 10
widths_displayDoctors[5] ← 10
widths_displayDoctors[6] ← 15
headers_displayDoctors[0] ← "No"
headers_displayDoctors[1] ← "Dokter"
headers_displayDoctors[2] ← "Spesialisasi"
headers_displayDoctors[3] ← "Ruangan"
headers_displayDoctors[4] ← "Antrian"
headers_displayDoctors[5] ← "Aura"
headers_displayDoctors[6] ← "Biaya Checkup"

printTableBorder(widths_displayDoctors, 7, 1)
printTableRow(headers_displayDoctors,
widths_displayDoctors, 7)
printTableBorder(widths_displayDoctors, 7, 2)

i traversal [0...availableDoctorCount-1]
    doctor ← availableDoctors[i]
    queueCount ← 0
    j traversal [0...hospital.queues.nRooms-1]
        if (strcmp(hospital.queues.queues[j].roomCode,
doctor.room) = 0) then

```

```

queueCount ←
queueSize(&hospital.queues.queues[j])
stop

if (not integerToString(i + 1, numberStr, 10)) then
    printError("Gagal mengonversi nomor ke string!")
    return
    strcat(numberStr, ". ")

if (not integerToString(queueCount, queueCountStr,
10)) then
    printError("Gagal mengonversi jumlah antrian ke
string!")
    return

if (not integerToString(doctor.aura, auraStr, 10))
then
    printError("Gagal mengonversi aura ke string!")
    return

if (not floatToString(doctor.checkupCost,
checkupCostStr, 20, 2)) then
    printError("Gagal mengonversi biaya checkup ke
string!")
    return

row_displayDoctors[0] ← numberStr
row_displayDoctors[1] ← doctor.username
row_displayDoctors[2] ← doctor.specialization
row_displayDoctors[3] ← doctor.room

```

```

        row_displayDoctors[4] ← queueCountStr
        row_displayDoctors[5] ← auraStr
        row_displayDoctors[6] ← checkupCostStr
        printTableRow(row_displayDoctors,
widths_displayDoctors, 7)
        printTableBorder(widths_displayDoctors, 7, 3)

        doctorChoice ← 0
        while (doctorChoice < 1 or doctorChoice >
availableDoctorCount) do
            promptSelectDoctor[0] ← '\0'
            strcat(promptSelectDoctor, "Pilih dokter (1 - ")
            if (not integerToString(availableDoctorCount,
availableCountStr, 10)) then
                printError("Gagal mengonversi jumlah dokter
tersedia ke string!")
            return
            strcat(promptSelectDoctor, availableCountStr)
            strcat(promptSelectDoctor, ": ")

            if (not readValidInt(&doctorChoice,
promptSelectDoctor)) then
                printError("Pilihan tidak valid!, harap masukkan
angka yang sesuai.")
            → false

        doctor ← availableDoctors[doctorChoice - 1]

        if (patient.bananaRich < doctor.checkupCost) then

```

```

        printError("Saldo BananaRich tidak cukup untuk
checkup!")
        → false

        queueIdx ← -1
        i traversal [0...hospital.queues.nRooms-1]
            if (strcmp(hospital.queues.queues[i].roomCode,
doctor.room) = 0) then
                queueIdx ← i
                stop

            if (queueIdx = -1) then
                if (hospital.queues.nRooms ≥
hospital.queues.capacity) then
                    printError("Kapasitas antrian penuh!")
                    → false

                queueIdx ← hospital.queues.nRooms
                hospital.queues.nRooms ← hospital.queues.nRooms + 1
                newQueue ← &hospital.queues.queues[queueIdx]
                initializeQueue(newQueue, doctor.room)

                queueToCheck ← &hospital.queues.queues[queueIdx]

                selectedRoom ← NIL
                i traversal [0...hospital.layout.rowEff-1]
                    j traversal [0...hospital.layout.colEff-1]
                        room ← &hospital.layout.elements[i][j]
                        if (strcmp(room.code, doctor.room) = 0) then
                            selectedRoom ← room

```

```

stop

if (selectedRoom ≠ NIL) then
    stop

    patient.bodyTemperature ← healthData[0]
    patient.systolicBloodPressure ← (integer)healthData[1]
    patient.diastolicBloodPressure ← (integer)healthData[2]
    patient.heartRate ← (integer)healthData[3]
    patient.oxygenSaturation ← (integer)healthData[4]
    patient.bloodSugarLevel ← healthData[5]
    patient.weight ← healthData[6]
    patient.height ← healthData[7]
    patient.cholesterolLevel ← healthData[8]
    patient.platelets ← (integer)healthData[9]

    patient.bananaRich ← patient.bananaRich -
doctor.checkupCost
    hospital.finance.hospitalBalance ←
hospital.finance.hospitalBalance + (0.2 * doctor.checkupCost)
    doctor.bananaRich ← doctor.bananaRich + (0.8 *
doctor.checkupCost)

    if (hospital.treatmentHistory.nEff ≥
hospital.treatmentHistory.capacity) then
        newElements ← (pointer to TreatmentHistory)
realloc(hospital.treatmentHistory.elements,
sizeof(TreatmentHistory) * hospital.treatmentHistory.capacity * 2)
    if (newElements = NIL) then
        printError("Gagal mengalokasikan ulang memori
untuk riwayat pengobatan!")

```

```

        patient.bananaRich ← patient.bananaRich +
doctor.checkupCost

        hospital.finance.hospitalBalance ←
hospital.finance.hospitalBalance - (0.2 * doctor.checkupCost)

        doctor.bananaRich ← doctor.bananaRich - (0.8 *
doctor.checkupCost)

        → false

        hospital.treatmentHistory.capacity ←
hospital.treatmentHistory.capacity * 2

        hospital.treatmentHistory.elements ← newElements

        history ←
&hospital.treatmentHistory.elements[hospital.treatmentHistory.nEff
]

        hospital.treatmentHistory.nEff ←
hospital.treatmentHistory.nEff + 1

        history.patientId ← patient.id
        history.doctorId ← doctor.id
        strcpy(history.room, doctor.room)
        strcpy(history.disease, "Belum Didagnosis")
        history.treatmentStatus ← false

if (selectedRoom ≠ NIL and
selectedRoom.patientInRoom.nEff < selectedRoom.capacity) then
        strcpy(patient.queueRoom, doctor.room)
        patient.queuePosition ← 0

selectedRoom.patientInRoom.patientId[selectedRoom.patientInRoom.nE
ff] ← patient.id

```

```

selectedRoom.patientInRoom.nEff ←
selectedRoom.patientInRoom.nEff + 1

output("\\n")
printHeader("Pendaftaran Checkup")

widths2[0] ← 20
widths2[1] ← 20
headers2[0] ← "Dokter"
headers2[1] ← "Ruangan"
printTableBorder(widths2, 2, 1)
printTableRow(headers2, widths2, 2)
printTableBorder(widths2, 2, 2)
row2[0] ← doctor.username
row2[1] ← doctor.room
printTableRow(row2, widths2, 2)
printTableBorder(widths2, 2, 3)

output("Anda telah dipindahkan langsung ke ruangan
dokter!")

successMsg[0] ← '\\0'
strcat(successMsg, "Pendaftaran checkup untuk ")
strcat(successMsg, session.username)
strcat(successMsg, " pada dr.")
strcat(successMsg, doctor.username)
strcat(successMsg, " di ruangan ")
strcat(successMsg, doctor.room)
strcat(successMsg, " berhasil!")
printSuccess(successMsg)

```

```

→ true

else if (not enqueue(queueToCheck, patient.id)) then
    printError("Gagal menambahkan pasien ke antrian!")

    patient.bananaRich ← patient.bananaRich +
doctor.checkupCost

    hospital.finance.hospitalBalance ←
hospital.finance.hospitalBalance - (0.2 * doctor.checkupCost)

    doctor.bananaRich ← doctor.bananaRich - (0.8 *
doctor.checkupCost)

    hospital.treatmentHistory.nEff ←
hospital.treatmentHistory.nEff - 1

→ false

else
    strcpy(patient.queueRoom, doctor.room)
    patient.queuePosition ← queueSize(queueToCheck)

output("\\n")
printHeader("Pendaftaran Checkup")

widths2[0] ← 20
widths2[1] ← 20
headers2[0] ← "Dokter"
headers2[1] ← "Ruangan"
printTableBorder(widths2, 2, 1)
printTableRow(headers2, widths2, 2)
printTableBorder(widths2, 2, 2)
row2[0] ← doctor.username

```

```

        row2[1] ← doctor.room
        printTableRow(row2, widths2, 2)
        printTableBorder(widths2, 2, 3)

output("Posisi antrian: ", patient.queuePosition)

successMsg[0] ← '\0'
strcat(successMsg, "Pendaftaran checkup untuk ")
strcat(successMsg, session.username)
strcat(successMsg, " pada dr.")
strcat(successMsg, doctor.username)
strcat(successMsg, " di ruangan ")
strcat(successMsg, doctor.room)
strcat(successMsg, " berhasil!")
printSuccess(successMsg)

→ true

```

## F15 - Antrian Saya!

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah
sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
    < userId : integer;
        username : string;
        role : Role;
        isLoggedIn : boolean >

```

```

type Hospital :
    < patients : PatientList;
        layout : Layout;
        doctors : DoctorList;
        queues : QueueList >

type PatientList :
    < elements : array [1..N] of Patient;
        nEff : integer >

type Patient :
    < id : integer;
        username : string;
        queueRoom : string;
        queuePosition : integer >

type Layout :
    < elements : array [1..rowMax][1..colMax] of Room;
        rowEff : integer;
        colEff : integer >

type Room :
    < code : string;
        patientInRoom : PatientInRoomList >

type PatientInRoomList :
    < patientId : array [1..N] of integer;
        nEff : integer >

```

```

type DoctorList :
    < elements : array [1..N] of Doctor;
        nEff : integer >

type Doctor :
    < id : integer;
        username : string;
        room : string >

type QueueList :
    < queues : array [1..N] of Queue;
        nRooms : integer >

type Queue :
    < roomCode : string;
        head : pointer to QueueNode >

type QueueNode :
    < info : QueueInfo;
        next : pointer to QueueNode >

type QueueInfo :
    < patientId : integer >

```

---

```

{ Definisi function & procedure yang di-import dari file source
code lain }

function strcmp(input s1 : string, input s2 : string) →
integer

```

```

{ Membandingkan dua string s1 dan s2. Mengembalikan 0 jika
sama. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

procedure printTableBorder(input widths : array of integer,
input numCols : integer, input type : integer)
{ Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input
widths : array of integer, input numCols : integer)
{ Mencetak baris tabel. }

function integerToString(input num : integer, output str :
string, input size : integer) → boolean
{ Mengonversi integer ke string. Mengembalikan true jika
berhasil. }

function queueSize(input q : pointer to Queue) → integer
{ Mengembalikan ukuran antrian. }

function strcpy(output destination : string, input source :
string)
{ Menyalin string source ke destination. }

```

```
function strcat(output destination : string, input source :  
string)  
{ Menggabungkan string source ke akhir string destination.  
}
```

```
procedure viewMySpecificQueueStatus(input hospital: pointer to  
Hospital, input session: pointer to Session)  
{ I.S. Menerima pointer ke struktur Hospital dan Session.  
F.S. Menampilkan status antrian pasien yang sedang login atau  
pesan error jika tidak ada antrian.  
}
```

#### KAMUS LOKAL

```
patient : pointer to Patient  
i, r, c : integer  
roomFound : boolean  
patientInRoom : boolean  
room : pointer to Room  
doctor : pointer to Doctor  
patientFoundInQueue : boolean  
queueIdx : integer  
queueToCheck : pointer to Queue  
currentNode : pointer to QueueNode  
queuePositionStr : character [5]  
queueSizeStr : character [5]  
finalQueueFormat : character [10]  
widths : array [1..2] of integer  
row1 : array [1..2] of string  
row2 : array [1..2] of string  
row3 : array [1..2] of string
```

## ALGORITMA

```
if (not session.isLoggedIn or session.role ≠ PATIENT) then
    printError("Akses ditolak! Hanya Pasien yang dapat melihat
status antrian.")
    return

else
    patient ← NIL
    i traversal [0...hospital.patients.nEff-1]
        if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then
            patient ← &hospital.patients.elements[i]
            stop

if (patient = NIL) then
    printError("Pasien tidak ditemukan dalam daftar!")
    return

if (patient.queueRoom[0] ≠ '\0') then
    roomFound ← false
    patientInRoom ← false
    r traversal [0...hospital.layout.rowEff-1]
        c traversal [0...hospital.layout.colEff-1]
        room ← &hospital.layout.elements[r][c]
        if (strcmp(room.code, patient.queueRoom) = 0)
            then
                roomFound ← true
                i traversal [0...room.patientInRoom.nEff-1]
                    if (room.patientInRoom.patientId[i] =
patient.id) then
```

```

        patientInRoom ← true
stop

        if (patientInRoom) then
            output("Anda sedang berada di dalam
ruangan dokter! Silakan menyelesaikan pemeriksaan.")
            return
stop

        if (roomFound) then
stop

        if (not roomFound) then
            printError("Ruang yang terdaftar untuk Anda tidak
ditemukan!")
            return

        if (patient.queueRoom[0] ≠ '\0') then
            doctor ← NIL
            i traversal [0...hospital.doctors.nEff-1]
            if (strcmp(hospital.doctors.elements[i].room,
patient.queueRoom) = 0) then
                doctor ← &hospital.doctors.elements[i]
stop

        if (doctor ≠ NIL) then
            patientFoundInQueue ← false
            queueIdx ← -1
            i traversal [0...hospital.queues.nRooms-1]

```

```

        if (strcmp(hospital.queues.queues[i].roomCode,
patient.queueRoom) = 0) then
            queueIdx ← i
            stop

        if (queueIdx = -1) then
            printError("Antrian untuk ruangan Anda tidak
ditemukan atau Anda tidak ada di dalamnya!")
            return

        queueToCheck ← &hospital.queues.queues[queueIdx]
        currentNode ← queueToCheck.head
        while (currentNode ≠ NIL) do
            if (currentNode.info.patientId = patient.id)
then
                patientFoundInQueue ← true
                stop
                currentNode ← currentNode.next

            if (not patientFoundInQueue) then
                printError("Anda belum terdaftar dalam antrian
check-up! Silakan daftar terlebih dahulu dengan command
DAFTAR_CHECKUP.")
                patient.queueRoom[0] ← '\0'
                patient.queuePosition ← 0
                return

            if (not integerToString(patient.queuePosition,
queuePositionStr, 5)) then -- sizeof(queuePositionStr) = 5

```

```

        printError("Gagal mengonversi posisi antrian ke
string!")

        return

    if (not integerToString(queueSize(queueToCheck),
queueSizeStr, 5)) then -- sizeof(queueSizeStr) = 5
        printError("Gagal mengonversi ukuran antrian ke
string!")

        return

    finalQueueFormat[0] ← '\0'
    strcat(finalQueueFormat, queuePositionStr)
    strcat(finalQueueFormat, "/")
    strcat(finalQueueFormat, queueSizeStr)

    printHeader("Status antrian Anda:")

    widths[0] ← 20
    widths[1] ← 20
    row1[0] ← "Dokter"
    row1[1] ← doctor.username
    row2[0] ← "Ruangan"
    row2[1] ← doctor.room
    row3[0] ← "Posisi antrian"
    row3[1] ← finalQueueFormat

    printTableBorder(widths, 2, 1)
    printTableRow(row1, widths, 2)
    printTableBorder(widths, 2, 2)
    printTableRow(row2, widths, 2)

```

```

        printTableBorder(widths, 2, 2)
        printTableRow(row3, widths, 2)
        printTableBorder(widths, 2, 3)

    return

else

    printError("Dokter untuk ruangan antrian Anda tidak
ditemukan!")

return

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

```

#### F16 - Minum Obat

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah
sakit }

type PatientMedicationPrescribedList :
< medicationID : array [1..N] of integer;
    capacity      : integer;
    nEff          : integer >

type PatientMedicationTakenList :
< medicationID : array [1..N] of integer;
    capacity      : integer;
    top           : integer >

type Medication :
< id    : integer;

```

```

        name : string >

type MedicationList :
< elements : array [1..N] of Medication;
  capacity : integer;
  nEff      : integer >

type MedicationPrescription :
< medicationID : integer;
  diseaseID     : integer;
  doseOrder     : integer >

type PrescriptionList :
< elements : array [1..N] of MedicationPrescription;
  capacity : integer;
  nEff      : integer >

type medicationOrder :
< medicationID : integer;
  doseOrder     : integer >

function takeMedication (hospital: pointer to Hospital, session: pointer to Session) → boolean
{ I.S. hospital terdefinisi dan berisi data rumah sakit, pasien, obat-obatan, dan antrian.
  F.S. Jika berhasil, obat yang dipilih pasien ditambahkan ke daftar obat yang diminum (patient.medicationsTaken), dan fungsi mengembalikan true.
  Jika obat yang diminum salah, nyawa pasien berkurang. Jika nyawa habis, pasien dihapus dan sesi direset.
}

```

```
        Jika gagal (misalnya, struktur tidak valid, akses ditolak,  
pasien tidak ditemukan, belum diresepkan,  
tidak ada obat diresepkan, atau urutan obat salah), fungsi  
mengembalikan false dan menampilkan pesan error yang sesuai. }
```

#### KAMUS LOKAL

```
patientIdx: integer  
patient: pointer to Patient  
needsAntidote: boolean  
lastMedicationId: integer  
takenCount: integer  
expectedMedicationId: integer  
isMedicationAvailable: boolean  
foundInTaken: boolean  
widths: array [0...1] of integer  
headers: array [0...1] of string  
validOptions: array [0...patient.medicationsPrescribed.nEff-1]  
of boolean  
displayCount: integer  
medicationId: integer  
medicationName: character [50]  
medicationIdStr: character [10]  
choice: integer  
selectedMedicationId: integer  
row: array [0...1] of string  
room: pointer to Room  
roomFound: boolean  
targetQueue: pointer to Queue  
nextPatientInfo: QueueInfo
```

#### ALGORITMA

```

if (hospital = NULL or session = NULL) then
    printError("Struktur rumah sakit atau sesi tidak valid!")
{Fungsi printError dari fitur print.h}
    → false
else if (not session.isLoggedIn or session.role ≠ PATIENT)
then
    printError("Akses ditolak! Hanya Pasien yang dapat
mengonsumsi obat.") {Fungsi printError dari fitur print.h}
    → false
else
    patientIdx ← -1
    for i traversal [0...hospital.patients.nEff-1]
        if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then {Fungsi strcmp dari fitur C standard
library}
            patientIdx ← i
            stop (true) { break from for loop }
        if (patientIdx = -1) then
            printError("Pasien tidak ditemukan dalam daftar!")
{Fungsi printError dari fitur print.h}
        → false
        patient ← &hospital.patients.elements[patientIdx]
        if (strcmp(patient.disease, "Tidak terdeteksi") = 0) then
{Fungsi strcmp dari fitur C standard library}
            output(COLOR_YELLOW, "Pasien tidak diresepkan obat
karena tidak ada penyakit terdeteksi", COLOR_RESET)
            → false
        else if (not patient.treatedStatus) then
            printError("Anda belum diberikan resep obat!") {Fungsi
printError dari fitur print.h}

```

```

        → false

else if (patient.medicationsPrescribed.nEff = 0) then
    printError("Tidak ada obat yang diresepkan!") {Fungsi
printError dari fitur print.h}

        → false

if (patient.medicationsTaken.top ≥ 0) then
    needsAntidote ← false
    lastMedicationId ←
patient.medicationsTaken.medicationId[patient.medicationsTaken.top
]

    takenCount ← patient.medicationsTaken.top + 1
    if (takenCount ≤ patient.medicationsPrescribed.nEff)
then
    expectedMedicationId ←
patient.medicationsPrescribed.medicationId[takenCount - 1]
    if (lastMedicationId ≠ expectedMedicationId) then
        needsAntidote ← true
    else
        needsAntidote ← true
    if (needsAntidote) then
        printError("Urutan obat salah! Silakan minum
penawar terlebih dahulu.") {Fungsi printError dari fitur print.h}
        → false

isMedicationAvailable ← false
for i traversal [0...patient.medicationsPrescribed.nEff-1]
    foundInTaken ← false
    for j traversal [0...patient.medicationsTaken.top]

```

```

        if (patient.medicationsPrescribed.medicationId[i]
= patient.medicationsTaken.medicationId[j]) then
            foundInTaken ← true
            stop (true) { break from inner for loop }
        if (not foundInTaken) then
            isMedicationAvailable ← true
            stop (true) { break from outer for loop }
        if (not isMedicationAvailable) then
            printError("Semua obat yang diresepkan sudah
dikonsumsi!") {Fungsi printError dari fitur print.h}
            → false

        printHeader("Daftar Obat yang Diresepkan") {Fungsi
printHeader dari fitur print.h}
        output(COLOR_BLUE, "[📋 | Info] - Daftar obat yang harus
diminum sesuai dengan urutan\n", COLOR_RESET)
        widths[0] ← 15
        widths[1] ← 30
        headers[0] ← "Urutan Minum"
        headers[1] ← "Nama Obat"
        printTableBorder(widths, 2, 1) {Fungsi printTableBorder
dari fitur print.h}
        printTableRow(headers, widths, 2) {Fungsi printTableRow
dari fitur print.h}
        printTableBorder(widths, 2, 2) {Fungsi printTableBorder
dari fitur print.h}

        displayCount ← 0
        for i traversal [0...patient.medicationsPrescribed.nEff-1]

```

```

        medicationId <
patient.medicationsPrescribed.medicationId[i]
        alreadyTaken < false
        for j traversal [0...patient.medicationsTaken.top]
            if (patient.medicationsTaken.medicationId[j] =
medicationId) then
                alreadyTaken < true
                stop (true) { break from inner for loop }
            if (not alreadyTaken) then
                validOptions[i] < true
                displayCount < displayCount + 1
                for j traversal [0...hospital.medications.nEff-1]
                    if (hospital.medications.elements[j].id =
medicationId) then
                        strcpy(medicationName,
hospital.medications.elements[j].name) {Fungsi strcpy dari fitur C
standard library}
                        integerToString(i + 1, medicationIdStr,
sizeof(medicationIdStr)) {Fungsi integerToString dari fitur
helper.h}
                        row[0] < medicationIdStr
                        row[1] < medicationName
                        printTableRow(row, widths, 2) {Fungsi
printTableRow dari fitur print.h}
                    stop (true) { break from for loop }
            else
                validOptions[i] < false
            printTableBorder(widths, 2, 3) {Fungsi printTableBorder
dari fitur print.h}
    
```

```

        while (not readValidInt(&choice, ">>> Pilih obat untuk
diminum: ")) {Fungsi readValidInt dari fitur helper.h}
            printError("Input tidak valid! Silakan masukkan nomor
obat yang benar.") {Fungsi printError dari fitur print.h}

        if (choice < 1 or choice >
patient.medicationsPrescribed.nEff) then
            printError("Pilihan nomor untuk obat tidak tersedia!")
{Fungsi printError dari fitur print.h}
        → false

        else if (not validOptions[choice - 1]) then
            printError("Obat tersebut sudah diminum atau tidak
tersedia!") {Fungsi printError dari fitur print.h}
        → false

expectedMedicationId ← -1
for i traversal [0...patient.medicationsPrescribed.nEff-1]
    foundInTaken ← false
    for j traversal [0...patient.medicationsTaken.top]
        if (patient.medicationsPrescribed.medicationId[i]
= patient.medicationsTaken.medicationId[j]) then
            foundInTaken ← true
            stop (true) { break from inner for loop }
        if (not foundInTaken) then
            expectedMedicationId ←
patient.medicationsPrescribed.medicationId[i]
            stop (true) { break from outer for loop }

```

```

        selectedMedicationId ←
patient.medicationsPrescribed.medicationId[choice - 1]

patient.medicationsTaken.medicationId[+patient.medicationsTaken.t
op] ← selectedMedicationId

    if (selectedMedicationId ≠ expectedMedicationId) then
        patient.life--
        output(COLOR_YELLOW, "[ | Info ] - Obat Salah! Nyawa
berkurang satu.\n", COLOR_RESET)
        if (patient.life ≤ 0) then
            printError("Pasien kehabisan nyawa! Pasien akan
dihapus. ) {Fungsi printError dari fitur print.h}
        if (patient.queueRoom[0] ≠ '\0') then
            room ← NULL
            roomFound ← false
            for i traversal [0...hospital.layout.rowEff-1]
                for j traversal
                    [0...hospital.layout.colEff-1]
                    if
                        (strcmp(hospital.layout.elements[i][j].code, patient.queueRoom) =
                        0) then {Fungsi strcmp dari fitur C standard library}
                            room ←
                            &hospital.layout.elements[i][j]
                            roomFound ← true
                            stop (true) { break from inner for
loop }
                    if (roomFound) then
                        stop (true) { break from outer for
loop }
    
```

```

        if (room) then
            for i traversal
                [0...room.patientInRoom.nEff-1]
                    if (room.patientInRoom.patientId[i] =
patient.id) then
                        for j traversal
                            [i...room.patientInRoom.nEff - 2]

room.patientInRoom.patientId[j] ← room.patientInRoom.patientId[j +
1]
room.patientInRoom.nEff--
stop (true) { break from for loop
}

targetQueue ← NULL
for i traversal
[0...hospital.queues.capacity-1]
if
(hospital.queues.queues[i].roomCode[0] ◁ '\0' and
strcmp(hospital.queues.queues[i].roomCode, patient.queueRoom) = 0)
then {Fungsi strcmp dari fitur C standard library}
        targetQueue ←
&hospital.queues.queues[i]
stop (true) { break from for loop
}

if (targetQueue and not
isEmpty(targetQueue)) then {Fungsi isEmpty dari fitur
queue.h}
        if (dequeue(targetQueue,
&nextPatientInfo.patientId)) then {Fungsi dequeue dari fitur
queue.h}

```

```

        if (room.patientInRoom.nEff <
room.patientInRoom.capacity) then

    room.patientInRoom.patientId[room.patientInRoom.nEff] ←
nextPatientInfo.patientId

            room.patientInRoom.nEff++
            deletePatient(hospital, patient.id) {Fungsi
deletePatient dari fitur patient.h}

            output(COLOR_YELLOW, "[● | Info ] - Sesi tidak
valid setelah pasien kehabisan nyawa.\n", COLOR_RESET)

            session.isLoggedIn ← false
            session.userId ← -1
            strcpy(session.username, "") {Fungsi strcpy dari
fitur C standard library}

        → false

else

    output("→ Sisa nyawa: ")
    for i traversal [0...2]
        if (i < patient.life) then
            output("● ")
        else
            output("○ ")
    output("\\n")
    → false

else

    printSuccess("Obat berhasil diminum!") {Fungsi
printSuccess dari fitur print.h}

    → true

```

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit
}

type PatientMedicationPrescribedList :
< medicationID : array [1..N] of integer;
  capacity      : integer;
  nEff          : integer >

type PatientMedicationTakenList :
< medicationID : array [1..N] of integer;
  capacity      : integer;
  top           : integer >

type Medication :
< id    : integer;
  name  : string >

type MedicationList :
< elements : array [1..N] of Medication;
  capacity : integer;
  nEff    : integer >

type MedicationPrescription :
< medicationID : integer;
  diseaseID     : integer;
  doseOrder     : integer >

type PrescriptionList :
< elements : array [1..N] of MedicationPrescription;
  capacity : integer;
  nEff    : integer >

type medicationOrder :
< medicationID : integer;
  doseOrder     : integer >

function takeAntidote (hospital: pointer to Hospital, session: pointer to Session) → boolean

```

KAMUS LOKAL

patientIdx: integer

```

isNeedAntidote: boolean
wrongMedicationId: integer
lastMedicationId: integer
expectedMedicationId: integer
takenCount: integer
patient: pointer to Patient
medicationName: character
lifeStr: character [15]
widths: array [0...2] of integer
headers: array [0...2] of string
row: array [0...2] of string
successMsg: character

```

#### ALGORITMA

```

if (hospital = NULL or session = NULL) then
    printError("Struktur rumah sakit atau sesi tidak valid!")
    → false
else if (not session.isLoggedIn or session.role ≠ PATIENT) then
    printError("Akses ditolak! Hanya Pasien yang dapat mengonsumsi
penawar.")
    → false
else
    patientIdx ← -1
    for i traversal [0...hospital.patients.nEff-1]
        if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then
            patientIdx ← i
            stop (true) { break from for loop }
        if (patientIdx = -1) then
            printError("Pasien tidak ditemukan!")
        → false

```

```

patient <- &hospital.patients.elements[patientIdx]
if (patient.medicationsTaken.top < 0) then
    printError("Anda belum mengonsumsi obat apapun!")
    → false
isNeedAntidote <- false
wrongMedicationId <- -1
lastMedicationId <

patient.medicationsTaken.medicationId[patient.medicationsTaken.top]
expectedMedicationId <- -1
takenCount <- patient.medicationsTaken.top + 1
if (takenCount ≤ patient.medicationsPrescribed.nEff) then
    expectedMedicationId <

patient.medicationsPrescribed.medicationId[takenCount - 1]
if (expectedMedicationId ≠ -1 and lastMedicationId ≠
expectedMedicationId) then
    isNeedAntidote <- true
    wrongMedicationId <- lastMedicationId
else
    for i traversal [0...patient.medicationsTaken.top]
        if (i < patient.medicationsPrescribed.nEff) then
            if (patient.medicationsTaken.medicationId[i] ≠
patient.medicationsPrescribed.medicationId[i]) then
                if (i = patient.medicationsTaken.top) then
                    isNeedAntidote <- true
                    wrongMedicationId <
patient.medicationsTaken.medicationId[i]
                    stop (true)
                if (isNeedAntidote and wrongMedicationId ≠ -1) then
                    strcpy(medicationName, "Tidak dikenal")
                    for i traversal [0...hospital.medications.nEff-1]

```

```

        if (hospital.medications.elements[i].id =
wrongMedicationId) then
            strcpy(medicationName,
hospital.medications.elements[i].name)
            stop (true)
            printHeader("Konsumsi Penawar")
            widths[0] ← 20
            widths[1] ← 20
            widths[2] ← 20
            headers[0] ← "Obat Dibatalkan"
            headers[1] ← "Nyawa Tersisa"
            headers[2] ← "Status"
            printTableBorder(widths, 3, 1)
            printTableRow(headers, widths, 3)
            printTableBorder(widths, 3, 2)
            strcpy(lifeStr, "")
            for i traversal [0...2]
                if (i < patient.life) then
                    strcat(lifeStr, "● ")
                else
                    strcat(lifeStr, "○ ")
            row[0] ← medicationName
            row[1] ← lifeStr
            row[2] ← "Penawar berhasil"
            printTableRow(row, widths, 3)
            printTableBorder(widths, 3, 3)
            strcpy(successMsg, "Penawar berhasil! Obat ")
            strcat(successMsg, medicationName)
            strcat(successMsg, " dibatalkan.")
            printSuccess(successMsg)

```

```

    → true

else

    printError("Anda tidak perlu mengonsumsi penawar! Urutan
obat yang diminum sudah benar.")

    → false

```

F18 - Exit

```

{ Definisi function & procedure yang di-import dari file source code
lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

procedure freeHospital(input hospital : pointer to Hospital)
{ Membebaskan memori yang dialokasikan untuk struktur hospital. }

procedure exitProgram(input hospital: pointer to Hospital, input
session: pointer to Session)
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Program diakhiri setelah membebaskan memori dan menampilkan
  pesan keluar.
}

ALGORITMA
  if (hospital = NIL or session = NIL) then

```

```

        printError("Struktur rumah sakit atau sesi tidak valid!")

        return

    else

        printHeader("Keluar Program")

        printSuccess("Terima kasih telah menggunakan Sistem Manajemen
Rumah Sakit Nimons!\n")

        output("[REDACTED]")
        output("|[REDACTED] Sampai Jumpa!|")
        output("[REDACTED]\n")
    
```

## B02 - Denah Dinamis

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< doctors : DoctorList;
  layout : Layout >

type DoctorList :
< elements : array [1..N] of Doctor;
  nEff : integer;
  capacity : integer >

type Doctor :
< id : integer;
  name : string;
  specialization : string;
  availability : Availability;
  rating : real >

```

```

username : string;
specialization : string;
aura : integer;
bananaRich : real;
room : string >

type Layout :
< elements : array [1..rowMax][1..colMax] of Room;
  rowEff : integer;
  colEff : integer;
  rowCapacity : integer;
  colCapacity : integer >

type Room :
< code : string;
  capacity : integer;
  doctorId : integer;
  patientInRoom : PatientInRoomList >

type PatientInRoomList :
< patientId : array [1..N] of integer; { Akan menjadi array of integer, bukan pointer }
  nEff : integer >

{ Definisi function & procedure yang di-import dari file source code lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

```

```

function strcat(output destination : string, input source : string)
    { Menggabungkan string source ke akhir string destination. }

function integerToString(input num : integer, output str : string, input
bufferSize : integer) → boolean
    { Mengkonversi integer ke string. }

function strcpy(output destination : string, input source : string)
    { Menyalin string source ke destination. }

function safeMalloc(input size : integer) → pointer to void
    { Mengalokasikan memori dengan pemeriksaan kesalahan. }

procedure free(input ptr : pointer to void)
    { Membebaskan memori yang dialokasikan. }

```

{ Fungsi Utama }

```

function changeLayout(input hospital: pointer to Hospital, input
session: pointer to Session, input newRowCount: integer, input
newColCount: integer) → boolean
    { I.S. Menerima pointer ke struktur Hospital, Session, dan ukuran
baris serta kolom denah yang baru.

    F.S. Mengembalikan true jika denah berhasil diubah; false jika
gagal.

        Jika gagal, pesan error yang sesuai ditampilkan.

    }

```

KAMUS LOKAL

```

i, j, k : integer
room : pointer to Room
errorMessage : string
newLayout : array of array of Room { Menggambarkan Room** }
code : string
numberStr : string

```

#### ALGORITMA

```

if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat mengubah
denah.")
    → false
if (newRowCount ≤ 0 or newColCount ≤ 0) then
    printError("Ukuran denah tidak valid!")
    → false

i traversal [0...hospital.layout.rowEff - 1]
j traversal [0...hospital.layout.colEff - 1]
if (i ≥ newRowCount or j ≥ newColCount) then
    room ← &hospital.layout.elements[i][j]
    if (room.doctorId ≠ -1) then
        k traversal [0...hospital.doctors.nEff - 1]
        if (hospital.doctors.elements[k].id =
room.doctorId) then
            strcpy(errorMessage, "")
            strcat(errorMessage, "Ruang ")
            strcat(errorMessage, room.code)
            strcat(errorMessage, " masih ditempati
oleh Dr.")
            strcat(errorMessage,
hospital.doctors.elements[k].username)

```

```

        printError(errorMessage)
        → false

newLayout ← (array of array of Room) safeMalloc(newRowCount *
sizeof(pointer to Room))
if (newLayout = NIL) then
    printError("Gagal alokasi memori untuk layout baru (rows)!")
    → false

i traversal [0...newRowCount - 1]
    newLayout[i] ← (array of Room) safeMalloc(newColCount *
sizeof(Room))
    if (newLayout[i] = NIL) then
        printError("Gagal alokasi memori untuk layout baru
(cols)!")
        k traversal [0...i - 1]
            free(newLayout[k])
        free(newLayout)
    → false

j traversal [0...newColCount - 1]
    strcpy(code, "")
    code[0] ← 'A' + i
    strcpy(numberStr, "")
    integerToString(j + 1, numberStr, 3)
    strcat(code, numberStr)
    strcpy(newLayout[i][j].code, code)
    newLayout[i][j].capacity ← 3
    newLayout[i][j].doctorId ← -1
    newLayout[i][j].patientInRoom.patientId ← (array of
integer) safeMalloc(3 * sizeof(integer))

```

```

{Asumsi PatientInRoomList.patientId menjadi array dinamis}

i traversal [0...hospital.layout.rowEff - 1]
    if (i < newRowCount) then { Hanya salin jika masih dalam batas
baris baru }

        j traversal [0...hospital.layout.colEff - 1]
            if (j < newColCount) then { Hanya salin jika masih
dalam batas kolom baru }

                newLayout[i][j].doctorId ←
hospital.layout.elements[i][j].doctorId
                newLayout[i][j].patientInRoom.nEff ←
hospital.layout.elements[i][j].patientInRoom.nEff
                k traversal [0...
hospital.layout.elements[i][j].patientInRoom.nEff - 1]
                    newLayout[i][j].patientInRoom.patientId[k] ←
hospital.layout.elements[i][j].patientInRoom.patientId[k]

i traversal [0...hospital.layout.rowEff - 1]
    j traversal [0...hospital.layout.colEff - 1]

    free(hospital.layout.elements[i][j].patientInRoom.patientId)
    free(hospital.layout.elements[i])
    free(hospital.layout.elements)

    hospital.layout.elements ← newLayout
    hospital.layout.rowEff ← newRowCount
    hospital.layout.colEff ← newColCount
    hospital.layout.rowCapacity ← newRowCount
    hospital.layout.colCapacity ← newColCount

```

```
printSuccess("Denah rumah sakit berhasil diubah!")
→ true
```

```
function moveDoctor(input hospital: pointer to Hospital, input session:
pointer to Session, input username: String, input newRoomCode: String)
→ boolean

{ I.S. Menerima pointer ke struktur Hospital, Session, username
dokter, dan kode ruangan baru.

F.S. Mengembalikan true jika dokter (dan pasiennya) berhasil
dipindahkan ke ruangan baru; false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}
```

#### KAMUS LOKAL

```
doctorToMove : pointer to Doctor
i, j, k : integer
targetRow : integer
targetCol : integer
targetRoom : pointer to Room
currentRoom : pointer to Room
patientId : integer
patientToMove : pointer to Patient
queue : pointer to Queue
```

#### ALGORITMA

```
if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat
memindahkan dokter.")
→ false
```

```

doctorToMove ← NIL
i traversal [0...hospital.doctors.nEff - 1]
    if (strcmp(hospital.doctors.elements[i].username, username) =
0) then
        doctorToMove ← &hospital.doctors.elements[i]
        break

if (doctorToMove = NIL) then
    printError("Dokter tidak ditemukan!")
    → false

targetRow ← ASCII(newRoomCode[0]) - ASCII('A')
targetCol ← atoi(newRoomCode[1...]) - 1 { Mengambil substring
dari indeks 1 dan mengkonversinya }

if (targetRow < 0 or targetRow ≥ hospital.layout.rowEff or
targetCol < 0 or targetCol ≥ hospital.layout.colEff) then
    printError("Kode ruangan tidak valid!")
    → false

targetRoom ← &hospital.layout.elements[targetRow][targetCol]
if (targetRoom.doctorId ≠ -1) then
    printError("Pemindahan gagal. Ruangan tersebut sudah
ditempati.")
    → false

currentRoom ← NIL
i traversal [0...hospital.layout.rowEff - 1]
    j traversal [0...hospital.layout.colEff - 1]

```

```

    if (hospital.layout.elements[i][j].doctorId =
doctorToMove.id) then
        currentRoom ← &hospital.layout.elements[i][j]
        break
    if (currentRoom) then
        break

    if (not currentRoom) then
        printError("Dokter tidak memiliki ruangan.")
        → false

currentRoom.doctorId ← -1
targetRoom.doctorId ← doctorToMove.id
strcpy(doctorToMove.room, newRoomCode)

i traversal [0...currentRoom.patientInRoom.nEff - 1]
    patientId ← currentRoom.patientInRoom.patientId[i]

    patientToMove ← NIL
    j traversal [0...hospital.patients.nEff - 1]
        if (hospital.patients.elements[j].id = patientId) then
            patientToMove ← &hospital.patients.elements[j]
            break

    if (patientToMove) then
        k traversal [0...targetRoom.patientInRoom.capacity - 1]
            if (targetRoom.patientInRoom.patientId[k] = 0) then {
                Asumsi 0 berarti slot kosong }

                targetRoom.patientInRoom.patientId[k] ← patientId

```

```

        targetRoom.patientInRoom.nEff ←
targetRoom.patientInRoom.nEff + 1
            break

currentRoom.patientInRoom.nEff ← 0 { Mengosongkan daftar pasien
di ruangan lama }
{ Asumsi perlu mengosongkan elemen array patientId juga, atau
menimpa dengan 0 }

i traversal [0...hospital.queues.nRooms - 1]
queue ← &hospital.queues.queues[i]
if (strcmp(queue.roomCode, currentRoom.code) = 0) then
    strcpy(queue.roomCode, newRoomCode)

printSuccess("Dokter dan pasien berhasil dipindahkan!")
→ true

```

B03 - Aura!

```

procedure displayDoctors(input hospital: pointer to Hospital, input
session: pointer to Session)
{ I.S. Menerima pointer ke struktur Hospital dan Session.
  F.S. Menampilkan daftar semua dokter dalam sistem
       jika pengguna yang login adalah Manajer; jika tidak, pesan
       error ditampilkan.
}

```

#### KAMUS LOKAL

```

widths_doctors : array [1..3] of integer
headers_doctors : array [1..3] of string
doctor : pointer to Doctor
idStr_doctors : character
auraStr : character
row_doctors : array [1..3] of string
aura : integer

```

## ALGORITMA

```
depend on (true) :
(hospital = NIL or session = NIL) :
    printError("Struktur rumah sakit atau sesi tidak valid!")
    return
(not session.isLoggedIn or session.role ≠ MANAGER) :
    printError("Akses ditolak! Hanya Manajer yang dapat
melihat daftar dokter.")
    return
(hospital.doctors.nEff = 0) :
    output("Tidak ada dokter terdaftar.")
    return
(otherwise) :
    printHeader("Daftar Dokter")
    widths_doctors[0] ← 5
    widths_doctors[1] ← 20
    widths_doctors[2] ← 10
    headers_doctors[0] ← "ID"
    headers_doctors[1] ← "Username"
    headers_doctors[2] ← "Aura"
    printTableBorder(widths_doctors, 3, 1)
    printTableRow(headers_doctors, widths_doctors, 3)
    printTableBorder(widths_doctors, 3, 2)

    i traversal [0...hospital.doctors.nEff-1]
        doctor ← &hospital.doctors.elements[i]

        id ← doctor.id
        if (not integerToString(id, idStr_doctors, 10)) then
-- sizeof(idStr) = 10
            printError("Gagal mengonversi ID ke string!")
            return

        aura ← doctor.aura
        if (not integerToString(aura, auraStr, 10)) then --
sizeof(auraStr) = 10
            printError("Gagal mengonversi aura ke string!")
            return
```

```

    row_doctors[0] ← idStr_doctors
    row_doctors[1] ← doctor.username
    row_doctors[2] ← auraStr
    printTableRow(row_doctors, widths_doctors, 3)
    printTableBorder(widths_doctors, 3, 3)

```

B04 - Banarich!!!

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit
}

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
< userId : integer;
  username : string;
  role : Role;
  isLoggedIn : boolean >

type Hospital :
< patients : PatientList >

type PatientList :
< elements : array [1..N] of Patient;
  nEff : integer;
  capacity : integer >

type Patient :
< id : integer;
  username : string;
  password : PasswordData;
  role : Role;

```

```

bananaRich : real >

type PasswordData :
< encryptedContent : string >

{ Definisi function & procedure yang di-import dari file source code
lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

function strcmp(input s1 : string, input s2 : string) → integer
{ Membandingkan dua string. Mengembalikan 0 jika sama. }

function generateSeed(input patient : pointer to Patient) → unsigned
long
{ Menghasilkan seed untuk random number generator berdasarkan
data pasien. }

function lcgRandom(input seed : unsigned long) → unsigned long
{ Menghasilkan angka acak menggunakan Linear Congruential
Generator. }

procedure printHeader(input message : string)
{ Mencetak header yang diformat. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)

```

```
{ Menggabungkan string source ke akhir string destination. }
```

```
function floatToString(input num : real, output str : string, input
bufferSize : integer, input decimalPlaces : integer) → boolean
{ Mengkonversi float ke string dengan jumlah tempat desimal
tertentu. }
```

```
{ Fungsi Utama }
```

```
function gacha(input hospital: pointer to Hospital, input session:
pointer to Session) → boolean
{ I.S. Menerima pointer ke struktur Hospital dan Session.
F.S. Mengembalikan true jika gacha berhasil dilakukan dan hadiah
diberikan; false jika gagal.
Jika gagal, pesan error yang sesuai ditampilkan.
}
```

#### KAMUS LOKAL

```
patientIndex : integer
patient : pointer to Patient
seed : unsigned long
rand1 : unsigned long
rand2 : unsigned long
rand3 : unsigned long
reward : real
rewardMessage : string
rewardType : integer
itemChoice : integer
smallRewards : array [1..3] of integer
```

```

mediumRewards : array [1..3] of integer
largeRewards : array [1..2] of integer
rewardStr : string
beforeStr : string
afterStr : string
beforeBalance : real
afterBalance : real

```

#### ALGORITMA

```

depend on (true) :
    (not session.isLoggedIn or session.role ≠ PATIENT) :
        printError("Akses ditolak! Hanya Pasien yang dapat
melakukan gacha.")
        → false
    (otherwise) :
        patientIndex ← -1
        FOR (i ← 0 to hospital.patients.nEff - 1) DO
            if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then
                patientIndex ← i
                break
            if (patientIndex = -1) then
                printError("Pasien tidak ditemukan!")
                → false

        patient ← &hospital.patients.elements[patientIndex]

        seed ← generateSeed(patient)

```

```

rand1 ← lcgRandom(seed)
rand2 ← lcgRandom(rand1)
rand3 ← lcgRandom(rand2)

reward ← 0.0
strcpy(rewardMessage, "")

rewardType ← rand2 % 100

itemChoice ← rand3 % 3

if (rewardType < 40) then
    printHeader("Hasil Gacha")
    output("Selamat datang " & patient.username & ", di
Mesin Gacha " & FORMAT_BOLD & COLOR_BLUE & "XXeon06" &
FORMAT_RESET & "!\n")
    strcpy(rewardMessage, "[✖️ Yahhh] - Kamu kurang
beruntung, coba lagi nanti!")
    output(COLOR_YELLOW & FORMAT_BOLD & rewardMessage &
"\n" & FORMAT_RESET)
    → true

else if (rewardType < 75) then
    smallRewards ← [2, 5, 10]
    reward ← smallRewards[itemChoice]
    strcat(rewardMessage, "[❗ WOW] - ")
    strcat(rewardMessage, "Selamat! ")
    strcat(rewardMessage, "Anda mendapatkan ")
    floatToString(reward, rewardStr, 10, 2)
    strcat(rewardMessage, rewardStr)

```

```

        strcat(rewardMessage, " BananaRich!")

else if (rewardType < 90) then

    mediumRewards ← [15, 20, 25]
    reward ← mediumRewards[itemChoice]
    strcat(rewardMessage, "[🎉 YAY] - ")
    strcat(rewardMessage, "Selamat! ")
    strcat(rewardMessage, "Anda mendapatkan ")
    floatToString(reward, rewardStr, 10, 2)
    strcat(rewardMessage, rewardStr)
    strcat(rewardMessage, " BananaRich!")

else if (rewardType < 98) then

    largeRewards ← [50, 100]
    reward ← largeRewards[itemChoice % 2]
    strcat(rewardMessage, "[🎊 HOREE] - ")
    strcat(rewardMessage, "Selamat! ")
    strcat(rewardMessage, "Anda mendapatkan ")
    floatToString(reward, rewardStr, 10, 2)
    strcat(rewardMessage, rewardStr)
    strcat(rewardMessage, " BananaRich!")

else

    reward ← 200
    strcat(rewardMessage, "[🎰 JACKPOT] - ")
    strcat(rewardMessage, "Selamat! ")
    strcat(rewardMessage, "Anda mendapatkan ")
    floatToString(reward, rewardStr, 10, 2)
    strcat(rewardMessage, rewardStr)
    strcat(rewardMessage, " BananaRich!")

patient.bananaRich ← patient.bananaRich + reward

```

```

printHeader("Hasil Gacha")

    output("Selamat datang " & patient.username & ", di
Mesin Gacha " & FORMAT_BOLD & COLOR_BLUE & "XXeon06" &
FORMAT_RESET & "!\n")

    output(COLOR_YELLOW & FORMAT_BOLD & rewardMessage &
"\n" & FORMAT_RESET)

beforeBalance ← patient.bananaRich - reward
afterBalance ← patient.bananaRich

if (not floatToString(beforeBalance, beforeStr, 50, 2)
or not floatToString(afterBalance, afterStr, 50, 2)) then
    printError("Gagal mengonversi saldo ke string!")
    → false

    output("\n[  | Info]: Saldo sebelum: " & beforeStr &
"\n")
    output("[  | Info]: Saldo sesudah: " & afterStr &
"\n")

    → true

```

```

procedure viewFinancial(input hospital: pointer to Hospital, input
session: pointer to Session)
{ I.S. Menerima pointer ke struktur Hospital dan Session.

```

F.S. Menampilkan laporan finansial rumah sakit atau pesan error jika akses ditolak.

}

#### KAMUS LOKAL

```
balance : real
balanceStr : string
widths : array [1..2] of integer
headers : array [1..2] of string
row : array [1..2] of string
```

#### ALGORITMA

```
if (not session.isLoggedIn or session.role ≠ MANAGER) then
    printError("Akses ditolak! Hanya Manajer yang dapat melihat
finansial.")

    return

printHeader("Finansial Rumah Sakit")
balance ← hospital.finance.hospitalBalance
strcpy(balanceStr, "")

if (not floatToString(balance, balanceStr, 50, 2)) then
    printError("Gagal mengonversi float ke string!")
    return

strcat(balanceStr, " BananaRich")
widths ← [15, 20]
headers ← ["Entitas", "Saldo"]
printTableBorder(widths, 2, 1)
printTableRow(headers, widths, 2)
printTableBorder(widths, 2, 2)
row ← ["Rumah Sakit", balanceStr]
printTableRow(row, widths, 2)
```

```
printTableBorder(widths, 2, 3)
```

```
procedure viewWallet(input hospital: pointer to Hospital, input session:  
pointer to Session)  
    { I.S. Menerima pointer ke struktur Hospital dan Session.  
      F.S. Menampilkan saldo dompet pengguna (pasien atau dokter) atau  
      pesan error jika akses ditolak.  
    }
```

#### KAMUS LOKAL

```
balance : real  
userType : string  
i : integer  
balanceStr : string  
widths : array [1..2] of integer  
headers : array [1..2] of string  
row : array [1..2] of string
```

#### ALGORITMA

```
if (not session.isLoggedIn or (session.role ≠ DOCTOR and  
session.role ≠ PATIENT)) then  
    printError("Akses ditolak! Hanya Dokter dan Pasien yang dapat  
    melihat dompet.")  
    →  
    printHeader("Saldo Dompet")  
    balance ← 0.0  
  
if (session.role = PATIENT) then  
    userType ← "Pasien"  
else
```

```

userType ← "Dokter"

if (session.role = PATIENT) then
    i traversal [0...hospital.patients.nEff - 1]
        if (strcmp(hospital.patients.elements[i].username,
session.username) = 0) then
            balance ← hospital.patients.elements[i].bananaRich
            break
    else
        i traversal [0...hospital.doctors.nEff - 1]
            if (strcmp(hospital.doctors.elements[i].username,
session.username) = 0) then
                balance ← hospital.doctors.elements[i].bananaRich
                break

strcpy(balanceStr, "")
if (not floatToString(balance, balanceStr, 50, 2)) then
    printError("Gagal mengonversi float ke string!")
    →

strcat(balanceStr, " BananaRich")

widths ← [15, 20]
headers ← [userType, "Saldo"]

printTableBorder(widths, 2, 1)
printTableRow(headers, widths, 2)
printTableBorder(widths, 2, 2)

row ← [session.username, balanceStr]

```

```

printTableRow(row, widths, 2)
printTableBorder(widths, 2, 3)

```

## B05 - Dead or Alive?!

```

{ Definisi function & procedure yang di-import dari file source code
lain }

procedure deletePatient (input/output hospital : pointer to Hospital,
input patientid : integer)
    { Menghapus pasien dari sistem }

```

### ALGORITMA

```

patient.life ← patient.life - 1

if (patient.life ≤ 0) then
    output("Pasien kehabisan nyawa! Pasien akan dihapus. 💀")
    {Prosedur printError dari fitur lain}
    deletePatient(hospital, patient.id)
    → false

else
    output("[💀 | Info ] - Obat Salah! Nyawa berkurang satu.\n")
    output("→ Sisa nyawa:")
    FOR (i ← 0 to 2) DO
        if (i < patient.life) then
            output("❤️")
        else
            output("💔")
    output("\n")
    FOR (i ← 1 to patient.medicationsPrescribed.nEff - 1) DO
        patient.medicationsPrescribed.medicationId[i - 1] ←
        patient.medicationsPrescribed.medicationId[i]

```

```

patient.medicationsPrescribed.nEff ←
patient.medicationsPrescribed.nEff - 1
→ false

```

## B06 - Mainin Antrian

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Session :
<   userId : integer;
    username : string;
    role : Role;
    isLoggedIn : boolean >

type Hospital :
<   users : UserList;
    doctors : DoctorList;
    layout : Layout;
    queues : QueueList >

type QueueList :
<   elements : array [1..N] of Queue;
    nEff : integer;
    capacity : integer >

type Queue :

```

```

< patientId : array [1..M] of integer;
  head : integer;
  tail : integer;
  size : integer;
  roomCode : string >

{ Definisi function & procedure yang di-import dari file source code
lain }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure printSuccess(input message : string)
{ Mencetak pesan sukses. }

function strlen(input s : string) → integer
{ Mencari panjang string s. }

function strcpy(output destination : string, input source : string)
{ Menyalin string source ke destination. }

function strcat(output destination : string, input source : string)
{ Menggabungkan string source ke akhir string destination. }

function findQueueByRoomCode(input hospital : pointer to Hospital, input
roomCode : string) → pointer to Queue
{ Mencari antrian berdasarkan kode ruangan. }

function isQueueEmpty(input q : pointer to Queue) → boolean
{ Memeriksa apakah antrian kosong. }

function queueSize(input q : pointer to Queue) → integer

```

```

{ Mengembalikan ukuran antrian. }

function dequeue(input q : pointer to Queue, output patientId : integer)
→ boolean

{ Mengambil pasien dari depan antrian. }

function enqueue(input q : pointer to Queue, input patientId : integer)
→ boolean

{ Menambahkan pasien ke akhir antrian. }

procedure updatePatientPositionsInQueue(input hospital : pointer to
Hospital, input q : pointer to Queue)

{ Memperbarui posisi pasien dalam antrian. }

```

{ Fungsi Utama }

```

function skipPatientInQueue(input hospital: pointer to Hospital, input
session: pointer to Session, input roomCode: String) → boolean

{ I.S. Menerima pointer ke struktur Hospital, Session, dan kode ruangan.

F.S. Mengembalikan true jika pasien terdepan berhasil dipindahkan ke
akhir antrian; false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}

```

#### KAMUS LOKAL

```

q : pointer to Queue

err : string

patientIdToSkip : integer

successMsg : string

```

## ALGORITMA

```
depend on (true) :  
  
(hospital = NIL or session = NIL or roomCode = NIL) :  
  
printError("Data input tidak valid untuk skip antrian.")  
  
→ false  
  
(not session.isLoggedIn or session.role ≠ MANAGER) :  
  
printError("Akses ditolak! Hanya Manajer yang dapat melakukan  
operasi skip antrian.")  
  
→ false  
  
(otherwise) :  
  
q ← findQueueByRoomCode(hospital, roomCode)  
  
if (q = NIL) then  
  
strcpy(err, "Antrian untuk ruangan ")  
  
strcat(err, roomCode)  
  
strcat(err, " tidak ditemukan.")  
  
printError(err)  
  
→ false  
  
if (isQueueEmpty(q) or queueSize(q) < 2) then  
  
printError("Tidak cukup pasien dalam antrian untuk melakukan skip  
(perlu minimal 2).")  
  
→ false  
  
if (not dequeue(q, patientIdToSkip)) then  
  
printError("Gagal mengambil pasien dari depan antrian saat skip.")
```

```

→ false

if (not enqueue(q, patientIdToSkip)) then

printError("Gagal menambahkan pasien ke akhir antrian saat skip.")

→ false

updatePatientPositionsInQueue(hospital, q)

strcpy(successMsg, "Pasien terdepan di antrian ruangan ")

strcat(successMsg, roomCode)

strcat(successMsg, " telah dipindahkan ke akhir. Antrian diperbarui.")

printSuccess(successMsg)

→ true

```

```

function cancelPatientFromQueue(input hospital: pointer to Hospital,
input session: pointer to Session, input patientUsernameToCancel:
String) → boolean

{ I.S. Menerima pointer ke struktur Hospital, Session, dan username
pasien yang akan dibatalkan antriannya.

F.S. Mengembalikan true jika pasien berhasil dibatalkan antriannya;
false jika gagal.

Jika gagal, pesan error yang sesuai ditampilkan.

}

```

#### KAMUS LOKAL

```

patientToCancelIdx : integer

err : string

patientToCancel : pointer to Patient

errMsg : string

```

```

originalRoomCode : string
q : pointer to Queue
removed : boolean
patientIdToCancel : integer
firstPatientId : integer
current : pointer to QueueNode
prev : pointer to QueueNode
idStr : string
successMsg : string

```

#### ALGORITMA

```

depend on (true) :
    (hospital = NIL or session = NIL or patientUsernameToCancel =
NIL) :
        printError("Data input tidak valid untuk pembatalan
antrian.")
        → false
    (not session.isLoggedIn) :
        printError("Akses ditolak. Silakan login terlebih
dahulu.")
        → false
    (otherwise) :
        patientToCancelIdx ← -1
        FOR (i ← 0 to hospital.patients.nEff - 1) DO
            if (strcmp(hospital.patients.elements[i].username,
patientUsernameToCancel) = 0) then
                patientToCancelIdx ← i
                break
            if (patientToCancelIdx = -1) then
                strcpy(err, "Pasien ")
                strcat(err, patientUsernameToCancel)

```

```

        strcat(err, "' yang akan dibatalkan antriannya tidak
ditemukan.")

        printError(err)
        → false

        patientToCancel ←
&hospital.patients.elements[patientToCancelIdx]

        if (session.role = PATIENT and strcmp(session.username,
patientUsernameToCancel) ≠ 0) then
            printError("Akses ditolak! Pasien hanya dapat
membatalkan antriannya sendiri.")

            → false

        if (patientToCancel.queueRoom[0] = '\0') then
            strcpy(errMsg, "Pasien ")
            strcat(errMsg, patientUsernameToCancel)
            strcat(errMsg, " tidak sedang dalam antrian apapun.")
            printError(errMsg)
            → false

            strcpy(originalRoomCode, patientToCancel.queueRoom)
            q ← findQueueByRoomCode(hospital, originalRoomCode)
            if (q = NIL) then
                strcpy(errMsg, "Error: Antrian untuk ruangan ")
                strcat(errMsg, originalRoomCode)
                strcat(errMsg, " (tempat pasien ")
                strcat(errMsg, patientUsernameToCancel)
                strcat(errMsg, " terdaftar) tidak ditemukan dalam
sistem.")

                printError(errMsg)
                patientToCancel.queueRoom[0] ← '\0'
                patientToCancel.queuePosition ← 0
                → false

```

```

removed ← false
patientIdToCancel ← patientToCancel.id
if (peekQueue(q, firstPatientId) and firstPatientId =
patientIdToCancel) then
    if (dequeue(q, firstPatientId)) then
        removed ← true
    else
        current ← q.head
        prev ← NIL
        while (current ≠ NIL) do
            if (current.info.patientId = patientIdToCancel)
            then
                if (prev = NIL) then
                    q.head ← current.next
                else
                    prev.next ← current.next
                if (current.next = NIL) then
                    q.tail ← prev
                else
                    current.next.prev ← prev
                free(current) { Asumsi ada free untuk memori
QueueNode }

q.size ← q.size - 1
removed ← true
break
prev ← current
current ← current.next
if (q.head = NIL) then
    q.tail ← NIL

```

```

if (not removed) then
    if (not integerToString(patientIdToCancel, idStr, 12))
then
        strcpy(idStr, "ERR")
        strcpy(errMsg, "Pasien ")
        strcat(errMsg, patientToCancel.username)
        strcat(errMsg, " (ID: ")
        strcat(errMsg, idStr)
        strcat(errMsg, ") secara teknis tidak ditemukan dalam
struktur antrian ")
        strcat(errMsg, originalRoomCode)
        strcat(errMsg, " meskipun terdaftar di sana. Data
mungkin korup.")
        printError(errMsg)

patientToCancel.queueRoom[0] ← '\0'
patientToCancel.queuePosition ← 0
if (q ≠ NIL) then
    updatePatientPositionsInQueue(hospital, q)

if (removed) then
    strcpy(successMsg, "Antrian untuk pasien ")
    strcat(successMsg, patientUsernameToCancel)
    strcat(successMsg, " di ruangan ")
    strcat(successMsg, originalRoomCode)
    strcat(successMsg, " berhasil dibatalkan.")

else
    strcpy(successMsg, "Status antrian untuk pasien ")
    strcat(successMsg, patientUsernameToCancel)

```

```

        strcat(successMsg, " telah dibersihkan karena
inkonsistensi data.")

printSuccess(successMsg)
→ true

```

## B07 - Search Suggestion

```

{ Prosedur Utama }
procedure displaySubstring(input hospital : pointer to Hospital,
input name : string, input found : boolean)
{ Menampilkan daftar username yang mengandung substring tertentu }

KAMUS LOKAL
i, m, k, widths : integer
roleStr, lowerName, lowerUsername : string
headers : array [0..2] of string

ALGORITMA
output("Berikut rekomendasi username yang relevan:")
widths ← [5, 20, 20]
headers ← ["ID", "Username", "Role"]
printTableBorder(widths, 3, 1)
printTableRow(headers, widths, 3)
printTableBorder(widths, 3, 2)

i traversal [0..hospital↑.users.nEff-1]
depend on (hospital↑.users.elements[i].role) :
    (MANAGER) :
        roleStr ← "Manager"
    (DOCTOR) :
        roleStr ← "Dokter"
    (PATIENT) :
        roleStr ← "Pasien"

lowerName ← toLower(name)
lowerUsername ← toLower(hospital↑.users.elements[i].username)

if (containsSubstring(lowerUsername, lowerName)) then
    if (found and lowerUsername = lowerName) then
        dealokasi(lowerName)
        dealokasi(lowerUsername)

```

```

else
    output(hospital↑.users.elements[i].id + " | " +
hospital↑.users.elements[i].username + " | " + roleStr)

    dealokasi(lowerName)
    dealokasi(lowerUsername)

printTableBorder(widths, 3, 3)

```

{ Fungsi Helper }

```

function isContainsSubstring(hospital: pointer to Hospital, name:
string) → boolean
{ Mengecek apakah ada username yang mengandung substring name
(case-insensitive) }

function containsSubstring(str: string, substr: string) → boolean
{ Mengembalikan true jika str mengandung substr sebagai substring
(case-sensitive) }

```

## PROGRAM UTAMA

Main: **hospitalSystem.c**

```

{ Definisi type yang dipakai berdasarkan sistem manajemen rumah sakit }

type Role: enumerasi [PATIENT, DOCTOR, MANAGER]

type Hospital :
    < users : UserList;
        doctors : DoctorList;
        patients : PatientList;
        layout : Layout;
        finance : FinanceData;

```

```

queues : QueueList;
treatmentHistory : TreatmentHistoryList;
diseases : DiseaseList >

type Session :
< userId : integer;
  username : string;
  role : integer;
  isLoggedIn : boolean >

{ Definisi function & procedure yang di-import dari file source code
lain }

function stringToInt(input s : string) → integer
{ Mengonversi string ke integer. }

procedure printError(input message : string)
{ Mencetak pesan error. }

procedure initHospital(output hospital : pointer to Hospital, input
userCapacity : integer, input doctorCapacity : integer, input
patientCapacity : integer, input initialRows : integer, input
initialCols : integer) → boolean
{ Menginisialisasi struktur Hospital. }

procedure printHeader(input title : string)
{ Mencetak judul header. }

function readValidString(output str : string, input maxLength : integer,
input prompt : string, input allowSpaces : boolean) → boolean
{ Membaca string dengan validasi. }

```

```

procedure normalizeCommand(output command : string)
    { Mengubah perintah menjadi huruf kapital dan menghilangkan
underscore. }

function strcmp(input s1 : string, input s2 : string) → integer
    { Membandingkan dua string s1 dan s2. Mengembalikan 0 jika sama. }

function readUsernameWithTrim(output username : string, input size :
integer, input prompt : string) → boolean
    { Membaca username dengan menghilangkan spasi di awal/akhir. }

function login(input hospital : pointer to Hospital, input/output
session : pointer to Session, input username : string, input password :
string) → boolean
    { Mengautentikasi pengguna. }

function registerPatient(input hospital : pointer to Hospital,
input/output session : pointer to Session, input username : string,
input password : string) → boolean
    { Mendaftarkan pasien baru. }

function logout(input/output session : pointer to Session) → boolean
    { Mengakhiri sesi pengguna. }

function forgotPassword(input hospital : pointer to Hospital, input
username : string, input rleCode : string, input newPassword : string)
→ boolean
    { Mengatur ulang password. }

procedure displayMenu(input session : pointer to Session)

```

```

{ Menampilkan daftar perintah. }

procedure displayHelp(input session : pointer to Session, input command
: string)
    { Menampilkan bantuan untuk perintah tertentu. }

procedure displayLayout(input hospital : pointer to Hospital, input
session : pointer to Session, input printHeaderFlag : boolean)
    { Menampilkan denah rumah sakit. }

procedure displayRoomDetails(input hospital : pointer to Hospital, input
session : pointer to Session, input roomCode : string)
    { Menampilkan detail ruangan. }

procedure printTableBorder(input widths : array of integer, input
numCols : integer, input type : integer)
    { Mencetak batas tabel. }

procedure printTableRow(input data : array of string, input widths :
array of integer, input numCols : integer)
    { Mencetak baris tabel. }

function readValidInt(output value : pointer to integer, input prompt :
string) → boolean
    { Membaca dan memvalidasi input integer. }

procedure displayUsers(input hospital : pointer to Hospital, input
session : pointer to Session, input sortBy : integer, input sortOrder :
integer)
    { Menampilkan daftar semua pengguna. }

```

```

procedure displayPatients(input hospital : pointer to Hospital, input
session : pointer to Session, input sortBy : integer, input sortOrder :
integer)
    { Menampilkan daftar pasien. }

procedure displayDoctors(input hospital : pointer to Hospital, input
session : pointer to Session, input sortBy : integer, input sortOrder :
integer)
    { Menampilkan daftar dokter. }

function readStringWithSpaces(output str : string, input size : integer,
input prompt : string) → boolean
    { Membaca string dengan spasi. }

procedure findUser(input hospital : pointer to Hospital, input session :
pointer to Session, input query : string, input getById : boolean)
    { Mencari pengguna. }

procedure findPatient(input hospital : pointer to Hospital, input
session : pointer to Session, input query : string, input getById :
boolean, input byDisease : boolean)
    { Mencari pasien. }

procedure findDoctor(input hospital : pointer to Hospital, input session
: pointer to Session, input query : string, input getById : boolean)
    { Mencari dokter. }

procedure displayQueue(input hospital : pointer to Hospital, input
session : pointer to Session)

```

```

{ Menampilkan antrian. }

function readValidFloat(output value : pointer to real, input prompt :
string) → boolean
    { Membaca dan memvalidasi input float. }

procedure addDoctor(input hospital : pointer to Hospital, input session
: pointer to Session, input username : string, input password : string,
input specialization : string, input checkupCost : real) → boolean
    { Menambah dokter baru. }

procedure assignDoctor(input hospital : pointer to Hospital, input
session : pointer to Session, input username : string, input roomCode :
string) → boolean
    { Menugaskan dokter ke ruangan. }

procedure diagnosePatient(input hospital : pointer to Hospital, input
session : pointer to Session) → boolean
    { Mendiagnosis pasien. }

procedure treatPatient(input hospital : pointer to Hospital, input
session : pointer to Session) → boolean
    { Merawat pasien. }

function canGoHome(input hospital : pointer to Hospital, input session :
pointer to Session) → boolean
    { Memeriksa status pulang pasien. }

```

```

function registerCheckup(input hospital : pointer to Hospital, input
session : pointer to Session, input healthData : array of real) →
boolean

{ Mendaftarkan check-up pasien. }

procedure viewMySpecificQueueStatus(input hospital : pointer to
Hospital, input session : pointer to Session)
{ Menampilkan status antrian spesifik pasien. }

procedure takeMedication(input hospital : pointer to Hospital, input
session : pointer to Session)
{ Pasien minum obat. }

procedure takeAntidote(input hospital : pointer to Hospital, input
session : pointer to Session)
{ Pasien minum penawar. }

procedure exitProgram(input hospital : pointer to Hospital, input
session : pointer to Session)
{ Keluar dari program. }

procedure changeLayout(input hospital : pointer to Hospital, input
session : pointer to Session, input rows : integer, input cols :
integer) → boolean

{ Mengubah ukuran denah. }

procedure moveDoctor(input hospital : pointer to Hospital, input session
: pointer to Session, input username : string, input roomCode : string)
→ boolean

{ Memindahkan dokter ke ruangan lain. }

```

```

procedure viewWallet(input hospital : pointer to Hospital, input session
: pointer to Session)
{ Menampilkan saldo BananaRich. }

procedure viewFinancial(input hospital : pointer to Hospital, input
session : pointer to Session)
{ Menampilkan laporan keuangan. }

procedure gacha(input hospital : pointer to Hospital, input session :
pointer to Session) → boolean
{ Melakukan gacha. }

procedure skipPatientInQueue(input hospital : pointer to Hospital, input
session : pointer to Session, input roomCode : string) → boolean
{ Melewati pasien di antrian. }

procedure cancelPatientFromQueue(input hospital : pointer to Hospital,
input session : pointer to Session, input patientUsernameToCancel :
string) → boolean
{ Membatalkan antrian pasien. }

function caseInsensitivestrcmp(input s1 : string, input s2 : string) →
integer
{ Membandingkan string tanpa memperhatikan huruf besar/kecil. }

procedure viewLifeStatus(input hospital : pointer to Hospital, input
session : pointer to Session)
{ Menampilkan status nyawa pasien. }

```

KAMUS LOKAL

```

hospital : Hospital
session : Session
hospitalRows : integer
hospitalCols : integer
command : character [50]
username : character [50]
password : character [100]
new_password : character [100]
rleCode : character [100]
helpCommand : character [50]
roomCode : character [5]
choiceSortBy : integer
choiceSortOrder : integer
widths : array [1..2] of integer
headers : array [1..2] of string
row1 : array [1..2] of string
row2 : array [1..2] of string
row3 : array [1..2] of string
_widths : array [1..2] of integer
_headers : array [1..2] of string
_row1 : array [1..2] of string
_row2 : array [1..2] of string
query : character [50]
byId : boolean
byDisease : boolean
choice : integer
specialization : character [50]
checkupCost : real
rows : integer
cols : integer

```

```
healthData : array [1..11] of real
prompts : array [1..10] of string
ranges : array [1..10][1..2] of real
errorMsg : character [100]
patientUsernameToCancel : character [51]
i, j : integer
```

#### PROGRAM UTAMA

```
{
```

I.S. Program dimulai dari konsol, mungkin dengan argumen ukuran denah.

F.S. Program berjalan, menerima perintah dari pengguna, hingga perintah 'EXIT' diberikan.

```
}
```

#### ALGORITMA

```
hospitalRows ← 5
hospitalCols ← 5
```

```
if (argc ≥ 2 and argc ≤ 3) then
    hospitalRows ← stringToInt(argv[1])
    if (argc = 3) then
        hospitalCols ← stringToInt(argv[2])
    else
        hospitalCols ← hospitalRows

    if (hospitalRows ≤ 0 or hospitalCols ≤ 0 or hospitalRows >
10 or hospitalCols > 10) then
        printError("Argumen ukuran denah tidak valid!")
    return 1
```

```

if (not initHospital(&hospital, 100, 100, 100, hospitalRows,
hospitalCols)) then
    printError("Inisialisasi rumah sakit gagal. Program
dihentikan.")

return 1

session.isLoggedIn ← false
session.userId ← -1
session.username[0] ← '\0'
session.role ← -1

printHeader("Sistem Manajemen Rumah Sakit Nimons")
output("Ketik 'MENU' untuk melihat daftar perintah.")

while (true) do
    if (not readValidString(command, 50, "\n>>> Masukkan perintah:
", false)) then
        printError("Perintah tidak valid!")
        continue
    normalizeCommand(command)

    depend on (command) :
        ("LOGIN") :
            if (not readUsernameWithTrim(username, 50, "Masukkan
username: ") or
                not readValidString(password, 100, "Masukkan
password: ", false)) then
                printError("Input tidak valid!")
                continue
            login(&hospital, &session, username, password)

```

```

stop

("REGISTERPASIEN") :
    if (not readUsernameWithTrim(username, 50, "Masukkan
username: ") or
        not readValidString(password, 100, "Masukkan
password: ", false)) then
        printError("Input tidak valid!")
        continue
    registerPatient(&hospital, &session, username,
password)

stop

("LOGOUT") :
    logout(&session)
stop

("LUPAPASSWORD") :
    if (not readUsernameWithTrim(username, 51, "Masukkan
username: ") or
        not readValidString(rleCode, 100, "Masukkan kode
RLE: ", false) or
        not readValidString(new_password, 100, "Masukkan
password baru: ", false)) then
        printError("Input tidak valid!")
        continue
    forgotPassword(&hospital, username, rleCode,
new_password)

stop

("MENU") :
    displayMenu(&session)
stop

("HELP") :

```

```

    if (not readValidString(helpCommand, 50, "Masukkan
nama perintah untuk bantuan: ", false)) then
        printError("Input tidak valid!")
        continue
        displayHelp(&session, helpCommand)
        stop
    ("LIHATDENAH") :
        displayLayout(&hospital, &session, true)
        stop
    ("LIHATTRUANGAN") :
        if (not readValidString(roomCode, 5, "Masukkan kode
ruangan: ", false)) then
            printError("Kode ruangan tidak valid!")
            continue
            displayRoomDetails(&hospital, &session, roomCode)
            stop
    ("LIHATUSER") :
        widths[0] ← 5
        widths[1] ← 20
        headers[0] ← "No."
        headers[1] ← "Urutan berdasarkan"
        printTableBorder(widths, 2, 1)
        printTableRow(headers, widths, 2)
        printTableBorder(widths, 2, 2)
        row1[0] ← "1."
        row1[1] ← "ID"
        row2[0] ← "2."
        row2[1] ← "Nama"
        printTableRow(row1, widths, 2)
        printTableRow(row2, widths, 2)

```

```

printTableBorder(widths, 2, 3)

if (not readValidInt(&choiceSortBy, "">>>> Pilihan: "))
then
    printError("Pilihan tidak valid!")
    continue

if (choiceSortBy < 1 or choiceSortBy > 2) then
    printError("Pilihan tidak valid!")
    continue

output("\n")

_widths[0] ← 5
_widths[1] ← 20
_headers[0] ← "No."
_headers[1] ← "Urutan Sorting"
printTableBorder(_widths, 2, 1)
printTableRow(_headers, _widths, 2)
printTableBorder(_widths, 2, 2)
_row1[0] ← "1."
_row1[1] ← "ASC (A-Z)"
_row2[0] ← "2."
_row2[1] ← "DESC (Z-A)"
printTableRow(_row1, _widths, 2)
printTableRow(_row2, _widths, 2)
printTableBorder(_widths, 2, 3)

if (not readValidInt(&choiceSortOrder, "">>>> Pilihan:
"))
then

```

```

        printError("Pilihan tidak valid!")
        continue

    if (choiceSortOrder < 1 or choiceSortOrder > 2) then
        printError("Pilihan tidak valid!")
        continue

    output("\\n")

    displayUsers(&hospital, &session, choiceSortBy,
choiceSortOrder)
    stop

("LIHATPASIEN") :
    widths[0] ← 5
    widths[1] ← 20
    headers[0] ← "No."
    headers[1] ← "Urutan berdasarkan"
    printTableBorder(widths, 2, 1)
    printTableRow(headers, widths, 2)
    printTableBorder(widths, 2, 2)
    row1[0] ← "1."
    row1[1] ← "ID"
    row2[0] ← "2."
    row2[1] ← "Nama"
    printTableRow(row1, widths, 2)
    printTableRow(row2, widths, 2)
    printTableBorder(widths, 2, 3)

if (not readValidInt(&choiceSortBy, "">>>> Pilihan: "))
then

```

```

        continue

    if (choiceSortBy < 1 or choiceSortBy > 2) then
        printError("Pilihan tidak valid!")
        continue

    output("\\n")

    _widths[0] ← 5
    _widths[1] ← 20
    _headers[0] ← "No."
    _headers[1] ← "Urutan Sorting"
    printTableBorder(_widths, 2, 1)
    printTableRow(_headers, _widths, 2)
    printTableBorder(_widths, 2, 2)
    _row1[0] ← "1."
    _row1[1] ← "ASC (A-Z)"
    _row2[0] ← "2."
    _row2[1] ← "DESC (Z-A)"
    printTableRow(_row1, _widths, 2)
    printTableRow(_row2, _widths, 2)
    printTableBorder(_widths, 2, 3)

    if (not readValidInt(&choiceSortOrder, " >> Pilihan:
")) then
        printError("Pilihan tidak valid!")
        continue

    if (choiceSortOrder < 1 or choiceSortOrder > 2) then
        printError("Pilihan tidak valid!")

```

```

        continue

output("\\n")

    displayPatients(&hospital, &session, choiceSortBy,
choiceSortOrder)

stop

("LIHATDOKTER") :
    widths[0] ← 5
    widths[1] ← 20
    headers[0] ← "No."
    headers[1] ← "Urutan berdasarkan"
    printTableBorder(widths, 2, 1)
    printTableRow(headers, widths, 2)
    printTableBorder(widths, 2, 2)
    row1[0] ← "1."
    row1[1] ← "ID"
    row2[0] ← "2."
    row2[1] ← "Nama"
    row3[0] ← "3."
    row3[1] ← "Aura"
    printTableRow(row1, widths, 2)
    printTableRow(row2, widths, 2)
    printTableRow(row3, widths, 2)
    printTableBorder(widths, 2, 3)

if (not readValidInt(&choiceSortBy, "">>>> Pilihan:"))

then

    printError("Pilihan tidak valid!")
    continue

```

```

if (choiceSortBy < 1 or choiceSortBy > 2) then
    printError("Pilihan tidak valid!")
    continue

output("\\n")

_widths[0] ← 5
_widths[1] ← 20
_headers[0] ← "No."
_headers[1] ← "Urutan Sorting"
printTableBorder(_widths, 2, 1)
printTableRow(_headers, _widths, 2)
printTableBorder(_widths, 2, 2)
_row1[0] ← "1."
_row1[1] ← "ASC (A-Z)"
_row2[0] ← "2."
_row2[1] ← "DESC (Z-A)"
printTableRow(_row1, _widths, 2)
printTableRow(_row2, _widths, 2)
printTableBorder(_widths, 2, 3)

if (not readValidInt(&choiceSortOrder, " >> Pilihan:
")) then
    printError("Pilihan tidak valid!")
    continue

if (choiceSortOrder < 1 or choiceSortOrder > 2) then
    printError("Pilihan tidak valid!")
    continue

```

```

output("\\n")

    displayDoctors(&hospital, &session, choiceSortBy,
choiceSortOrder)

    stop

("CARIUSER") :
    widths[0] ← 5
    widths[1] ← 20
    headers[0] ← "No."
    headers[1] ← "Cari berdasarkan"
    printTableBorder(widths, 2, 1)
    printTableRow(headers, widths, 2)
    printTableBorder(widths, 2, 2)
    row1[0] ← "1."
    row1[1] ← "ID"
    row2[0] ← "2."
    row2[1] ← "Nama"
    printTableRow(row1, widths, 2)
    printTableRow(row2, widths, 2)
    printTableBorder(widths, 2, 3)

    if (not readValidInt(&choice, " >>> Pilihan: ")) then
        printError("Pilihan tidak valid!")
        continue

output("\\n")

if (choice = 1) then
    byId ← true

```

```

        if (not readValidString(query, 50, "">>>> Masukkan
nomor ID user: ", false)) then
            printError("Input ID tidak valid!")
            continue
        output("\\n")
    else if (choice = 2) then
        byId ← false
        if (not readStringWithSpaces(query, 50, "">>>>
Masukkan nama user: ")) then
            printError("Input Nama tidak valid!")
            continue
        output("\\n")
    else
        printError("Pilihan tidak valid!")
        continue

    findUser(&hospital, &session, query, byId)
    stop
("CARIPASIEN") :
    widths[0] ← 5
    widths[1] ← 20
    headers[0] ← "No."
    headers[1] ← "Cari berdasarkan"
    printTableBorder(widths, 2, 1)
    printTableRow(headers, widths, 2)
    printTableBorder(widths, 2, 2)
    row1[0] ← "1."
    row1[1] ← "ID"
    row2[0] ← "2."
    row2[1] ← "Nama"

```

```

row3[0] ← "3."
row3[1] ← "Penyakit"
printTableRow(row1, widths, 2)
printTableRow(row2, widths, 2)
printTableRow(row3, widths, 2)
printTableBorder(widths, 2, 3)

if (not readValidInt(&choice, " >>> Pilihan: ")) then
    continue

output("\n")

if (choice = 1) then
    byId ← true
    byDisease ← false
    if (not readValidString(query, 50, " >>> Masukkan
nomor ID user: ", false)) then
        printError("Input ID tidak valid!")
        continue
    output("\n")
else if (choice = 2) then
    byId ← false
    byDisease ← false
    if (not readStringWithSpaces(query, 50, " >>>
Masukkan nama user: ")) then
        printError("Input Nama tidak valid!")
        continue
    output("\n")
else if (choice = 3) then
    byId ← false

```

```

        byDisease ← true
        if (not readStringWithSpaces(query, 50, " >>>
Masukkan nama penyakit: ")) then
            printError("Input Penyakit tidak valid!")
            continue
            output("\\n")
else
            printError("Pilihan tidak valid!")
            continue

        findPatient(&hospital, &session, query, byId,
byDisease)
stop
("CARIDOKTER") :
    widths[0] ← 5
    widths[1] ← 20
    headers[0] ← "No."
    headers[1] ← "Cari berdasarkan"
    printTableBorder(widths, 2, 1)
    printTableRow(headers, widths, 2)
    printTableBorder(widths, 2, 2)
    row1[0] ← "1."
    row1[1] ← "ID"
    row2[0] ← "2."
    row2[1] ← "Nama"
    printTableRow(row1, widths, 2)
    printTableRow(row2, widths, 2)
    printTableBorder(widths, 2, 3)

if (not readValidInt(&choice, " >>> Pilihan: ")) then

```

```

        printError("Pilihan tidak valid!")
        continue

        output("\\n")

        if (choice = 1) then
            byId ← true
            if (not readValidString(query, 51, "">>> Masukkan
nomor ID dokter: ", false)) then
                printError("Input ID tidak valid!")
                continue
            output("\\n")
        else if (choice = 2) then
            byId ← false
            if (not readStringWithSpaces(query, 51, "">>>
Masukkan nama dokter: ")) then
                printError("Input Nama tidak valid!")
                continue
            output("\\n")
        else
            printError("Pilihan tidak valid!")
            continue

        findDoctor(&hospital, &session, query, byId)
        stop
    ("LIHATANTRIAN") :
        displayQueue(&hospital, &session)
        stop
    ("TAMBAHDOKTER") :

```

```

        if (not readUsernameWithTrim(username, 51, "Masukkan
username dokter: ") or
            not readValidString(password, 100, "Masukkan
password dokter: ", false) or
            not readStringWithSpaces(specialization, 50,
"Masukkan spesialisasi dokter: ") or
            not readValidFloat(&checkupCost, "Masukkan biaya
checkup dokter: ")) then
                printError("Input tidak valid!")
                continue
            addDoctor(&hospital, &session, username, password,
specialization, checkupCost)
            stop
        ("ASSIGNDOKTER") :
        if (not readUsernameWithTrim(username, 51, "Masukkan
username dokter: ") or
            not readValidString(roomCode, 5, "Masukkan kode
ruangan: ", false)) then
                printError("Input tidak valid!")
                continue
            assignDoctor(&hospital, &session, username, roomCode)
            stop
        ("DIAGNOSIS") :
        diagnosePatient(&hospital, &session)
        stop
        ("NGOBATIN") :
        treatPatient(&hospital, &session)
        stop
        ("PULANGDOK") :
        canGoHome(&hospital, &session)
    
```

```

stop

("DAFTARCHECKUP") :
    prompts[0] ← "Suhu Tubuh (Celcius): "
    prompts[1] ← "Tekanan Darah Sistol (mmHg): "
    prompts[2] ← "Tekanan Darah Diastol (mmHg): "
    prompts[3] ← "Detak Jantung (bpm): "
    prompts[4] ← "Saturasi Oksigen (%): "
    prompts[5] ← "Kadar Gula Darah (mg/dL): "
    prompts[6] ← "Berat Badan (kg): "
    prompts[7] ← "Tinggi Badan (cm): "
    prompts[8] ← "Kadar Kolesterol (mg/dL): "
    prompts[9] ← "Trombosit (ribu/µL): "

ranges[0][0] ← 30.0; ranges[0][1] ← 45.0
ranges[1][0] ← 50.0; ranges[1][1] ← 200.0
ranges[2][0] ← 30.0; ranges[2][1] ← 120.0
ranges[3][0] ← 40.0; ranges[3][1] ← 200.0
ranges[4][0] ← 50.0; ranges[4][1] ← 100.0
ranges[5][0] ← 50.0; ranges[5][1] ← 300.0
ranges[6][0] ← 20.0; ranges[6][1] ← 200.0
ranges[7][0] ← 100.0; ranges[7][1] ← 250.0
ranges[8][0] ← 100.0; ranges[8][1] ← 400.0
ranges[9][0] ← 100.0; ranges[9][1] ← 1000.0

i traversal [0...9]
    while (true) do
        if (not readValidFloat(&healthData[i],
prompts[i])) then
            printError("Input tidak valid, silakan
coba lagi!")

```

```

        continue

        if (healthData[i] < ranges[i][0] or
healthData[i] > ranges[i][1]) then
            errorMsg[0] ← '\0'
            strcat(errorMsg, "Data kesehatan ''")
            strcat(errorMsg, prompts[i])
            strcat(errorMsg, "' tidak valid! Harap
masukkan nilai dalam rentang yang benar.")
            printError(errorMsg)
            continue
            break

        registerCheckup(&hospital, &session, healthData)
stop

("ANTRIANSAYA") :
    viewMySpecificQueueStatus(&hospital, &session)
stop

("MINUMOBAT") :
    takeMedication(&hospital, &session)
stop

("MINUMPENAWAR") :
    takeAntidote(&hospital, &session)
stop

("EXIT") :
    exitProgram(&hospital, &session)
    return 0

("UBAHDENAH") :
    if (not readValidInt(&rows, "Masukkan jumlah baris: ")
or

```

```

        not readValidInt(&cols, "Masukkan jumlah kolom:
")) then

            printError("Ukuran denah tidak valid!")
            continue
            changeLayout(&hospital, &session, rows, cols)
            stop

        ("PINDAHKANDOKTER") :

            if (not readUsernameWithTrim(username, 51, "Masukkan
username dokter: ") or
                not readValidString(roomCode, 5, "Masukkan kode
ruangan: ", false)) then
                printError("Input tidak valid!")
                continue
                moveDoctor(&hospital, &session, username, roomCode)
                stop

        ("LIHATDOMPET") :

            viewWallet(&hospital, &session)
            stop

        ("LIHATFINANSIAL") :

            viewFinancial(&hospital, &session)
            stop

        ("GACHA") :

            gacha(&hospital, &session)
            stop

        ("SKIPANTRIAN") :

            if (not readValidString(roomCode, 5, "Masukkan kode
ruangan antrian yang akan di-skip: ", false)) then
                continue
                skipPatientInQueue(&hospital, &session, roomCode)
                stop

```

```

("CANCELANTRIAN") :
    if (not readUsernameWithTrim(patientUsernameToCancel,
51, "Masukkan username pasien yang antriannya akan dibatalkan
(atau 'saya' untuk diri sendiri): ")) then
        continue

    if (caseInsensitivestrcmp(patientUsernameToCancel,
"saya") = 0) then
        if (session.role = PATIENT) then
            strcpy(patientUsernameToCancel,
session.username)
        else
            printError("Perintah 'saya' hanya valid untuk
Pasien. Manajer harus memasukkan username pasien secara
spesifik.")
        continue
    cancelPatientFromQueue(&hospital, &session,
patientUsernameToCancel)
    stop

("LIHATNYAWA") :
    viewLifeStatus(&hospital, &session)
    stop

(otherwise) :
    printError("Perintah tidak dikenal! Ketik 'MENU' untuk
bantuan.")
    stop

```

## TANGKAPAN LAYAR

F01 - Login

Kasus 1 : Login sebagai Manager
>>> Masukkan perintah: login Masukkan username: NimonsDawg Masukkan password: admoontothemoon <b>SUKSES: Teks berhasil dienkripsi!</b> <b>SUKSES: Login berhasil sebagai NimonsDawg! (Peran: Manajer)</b>
Kasus 2 : Login sebagai Dokter
>>> Masukkan perintah: login Masukkan username: Zulfa Masukkan password: puhsepuh <b>SUKSES: Teks berhasil dienkripsi!</b> <b>SUKSES: Login berhasil sebagai Zulfa! (Peran: Dokter)</b>
Kasus 3 : Login sebagai Pasien
>>> Masukkan perintah: login Masukkan username: Abel Masukkan password: abel123 <b>SUKSES: Teks berhasil dienkripsi!</b> <b>SUKSES: Login berhasil sebagai Abel! (Peran: Pasien)</b>
Kasus 4 : Tidak ada username yang terdaftar

```
>>> Masukkan perintah: login  
Masukkan username: almerresing  
Masukkan password: almergajadiresing  
ERROR: Username tidak ditemukan!
```

Kasus 5 : Kasus password salah

```
>>> Masukkan perintah: login  
Masukkan username: Nisa  
Masukkan password: proalpro  
SUKSES: Teks berhasil dienkripsi!  
ERROR: Password salah!
```

#### F02 - Register Pasien

Kasus 1 : Pasien bernama **Nisa** belum ada

```
>>> Masukkan perintah: login  
Masukkan username: Nisa  
Masukkan password: dibantaitubes  
ERROR: Username tidak ditemukan!
```

Kasus 2 : Pasien bernama **almerresing** sudah ada

```
>>> Masukkan perintah: registerpasien  
Masukkan username: Nisa  
Masukkan password: dibantaitubes  
SUKSES: Teks berhasil dienkripsi!  
SUKSES: Pasien dengan nama Nisa berhasil terdaftar!
```

#### F03 - Logout

Kasus 1 : sedang dalam keadaan logged in

```
>>> Masukkan perintah: logout  
SUKSES: Logout berhasil! Sampai jumpa, Nisa!
```

Kasus 2 : sedang dalam keadaan belum logged in

```
>>> Masukkan perintah: logout  
ERROR: Anda belum login! Silakan login terlebih dahulu.
```

#### F04 - Lupa Password

Kasus 1 : Manager, Dokter, atau Pasien bernama **Nisa** ada

```
>>> Masukkan perintah: lupapassword  
Masukkan username: Nisa  
Masukkan kode RLE: Nisa  
Masukkan password baru: tolongsaya  
SUKSES: Kode RLE berhasil dihasilkan!  
SUKSES: Teks berhasil dienkripsi!  
SUKSES: Password untuk Nisa berhasil diperbarui!
```

Kasus 2 : Manager, Dokter, atau Pasien bernama **Anomali** tidak ada

```
>>> Masukkan perintah: lupapassword  
Masukkan username: anomali  
Masukkan kode RLE: anomali  
Masukkan password baru: gatausih  
ERROR: Username tidak terdaftar!
```

Kasus 3 : Kode RLE untuk username **littlepony** salah

```
>>> Masukkan perintah: lupapassword  
Masukkan username: littlepony  
Masukkan kode RLE: lit2lepony  
Masukkan password baru: kepoluw  
SUKSES: Kode RLE berhasil dihasilkan!  
ERROR: Kode RLE salah!
```

Kasus 4 : Kode RLE untuk username **littlepony** benar

```
>>> Masukkan perintah: lupaPassword  
Masukkan username: littlepony  
Masukkan kode RLE: li2tlepony  
Masukkan password baru: kepoluw  
SUKSES: Kode RLE berhasil dihasilkan!  
SUKSES: Teks berhasil dienkripsi!  
SUKSES: Password untuk littlepony berhasil diperbarui!
```

## F05 - Menu & Help

### Menu

Kasus 1 : belum dalam keadaan logged in

```
>>> Masukkan perintah: menu
```

**Daftar Perintah**

Anda belum login. Perintah yang tersedia:

**LOGIN**  
**REGISTER\_PASIEN**  
**LUPA\_PASSWORD**  
**MENU**  
**HELP**  
**EXIT**

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

Kasus 2 : sudah login sebagai Dokter

```
>>> Masukkan perintah: menu
```

```
Daftar Perintah
```

Halo Dokter Zulfa. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

```
LOGOUT  
MENU  
HELP  
LIHAT_DENAH  
DIAGNOSIS  
NGOBATIN  
LIHAT_DOMPET  
EXIT
```

Footnote:

Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

Jangan lupa untuk memasukkan input yang valid

### Kasus 3 : sudah login sebagai Pasien

```
>>> Masukkan perintah: menu
```

```
Daftar Perintah
```

Selamat datang, Nisa. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

```
LOGOUT  
MENU  
HELP  
LIHAT_DENAH  
PULANG_DOK  
DAFTAR_CHECKUP  
ANTRIAN_SAYA  
MINUM_OBAT  
MINUM_PENAWAR  
LIHAT_DOMPET  
GACHA  
LEWATI_ANTRIAN  
BATALKAN_ANTRIAN  
EXIT
```

Footnote:

Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

Jangan lupa untuk memasukkan input yang valid

### Kasus 4 : sudah login sebagai Manager

```
>>> Masukkan perintah: menu
Daftar Perintah

Halo Manager NimonsDawg. Kenapa kamu memanggil command HELP? Kan kamu manager, tapi yasudahlah kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

LOGOUT
MENU
HELP
LIHAT_DENAH
LIHAT_USER
CARI_USER
LIHAT_ANTRIAN
TAMBAH_DOKTER
UBAH_DENAH
PINDAH_DOKTER
LIHAT_FINANSIAL
EXIT

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

## Help

Kasus 1 : belum dalam keadaan logged in

```
>>> Masukkan perintah: help
Masukkan nama perintah untuk bantuan: LOGIN

Bantuan Perintah

Anda belum login! Silakan login terlebih dahulu.
Bantuan untuk perintah 'LOGIN':
```

---

**🔑 LOGIN**  
Deskripsi: Masuk ke sistem dengan username dan password Anda.

Kasus 2 : sudah login sebagai Dokter

```
>>> Masukkan perintah: help
Masukkan nama perintah untuk bantuan: LOGOUT

Bantuan Perintah

Anda login sebagai Dokter.
Bantuan untuk perintah 'LOGOUT':
```

---

**✖ LOGOUT**  
Deskripsi: Keluar dari sesi Anda saat ini.

Kasus 3 : sudah login sebagai Pasien

```
>>> Masukkan perintah: help  
Masukkan nama perintah untuk bantuan: menu
```

Bantuan Perintah

Anda login sebagai Pasien.

Bantuan untuk perintah 'menu':



MENU

Deskripsi: Tampilkan daftar semua perintah yang tersedia.

Kasus 2 : sudah login sebagai Manajer

```
>>> Masukkan perintah: help  
Masukkan nama perintah untuk bantuan: lihatfinansial
```

Bantuan Perintah

Anda login sebagai Manajer.

Bantuan untuk perintah 'lihatfinansial':



LIHAT\_FINANSIAL

Deskripsi: Tampilkan laporan keuangan rumah sakit.

F06 - Denah Rumah Sakit

Kasus 1 : ruangan terdapat dokter dan pasien

>>> Masukkan perintah: lihatruangan  
Masukkan kode ruangan: A1

**Detail Ruangan A1**

Kapasitas	3
Dokter	Zulfa

No.	Pasien dalam Ruangan
1.	Ziza

Kasus 2 : ruangan terdapat dokter dan tidak ada pasien

>>> Masukkan perintah: lihatruangan  
Masukkan kode ruangan: A2

**Detail Ruangan A2**

Kapasitas	3
Dokter	Alya

No.	Pasien dalam Ruangan
0.	Tidak ada pasien

Kasus 3 : ruangan tidak terdapat dokter

>>> Masukkan perintah: lihatruangan  
Masukkan kode ruangan: A2

**Detail Ruangan A2**

Kapasitas	3
Dokter	Tidak ada dokter

No.	Pasien dalam Ruangan
0.	Tidak ada pasien

## F07 - Lihat User

### Kasus 1 : Melihat data user (dokter/pasien)

>>> Masukkan perintah: lihatuser

No.	Urutan berdasarkan
1.	ID
2.	Nama

>>> Pilihan: 2

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

>>> Pilihan: 1

**Daftar Pengguna**

Menampilkan semua pengguna dengan nama terurut ascending...

ID	Username	Role	Penyakit
1	NimonsDawg	Manajer	-
3	Nisa	Pasien	Influenza
2	Zulfa	Dokter	-
4	almer	Pasien	-

```
>>> Masukkan perintah: lihatuser
```

No.	Urutan berdasarkan
1.	ID
2.	Nama

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

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

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

### Daftar Pengguna

Menampilkan semua pengguna dengan ID terurut descending...

ID	Username	Role	Penyakit
4	almer	Pasien	-
3	Nisa	Pasien	Influenza
2	Zulfa	Dokter	-
1	NimonsDawg	Manajer	-

```
>>> Masukkan perintah: lihatuser
```

No.	Urutan berdasarkan
1.	ID
2.	Nama

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

### Daftar Pengguna

Menampilkan semua pengguna dengan nama terurut ascending...

ID	Username	Role	Penyakit
1	NimonsDawg	Manajer	-
3	Nisa	Pasien	Influenza
2	Zulfa	Dokter	-
4	almer	Pasien	-

>>> Masukkan perintah: lihatuser

No.	Urutan berdasarkan
1.	ID
2.	Nama

>>> Pilihan: 2

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

>>> Pilihan: 2

### Daftar Pengguna

Menampilkan semua pengguna dengan nama terurut descending...

ID	Username	Role	Penyakit
4	almer	Pasien	-
2	Zulfa	Dokter	-
3	Nisa	Pasien	Influenza
1	NimonsDawg	Manajer	-

Kasus 2 : Spesifik melihat data pasien

```
>>> Masukkan perintah: lihatpasien
```

No.	Urutan berdasarkan
1.	ID
2.	Nama

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

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

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

### Daftar Pasien

Menampilkan pasien dengan ID terurut ascending...

ID	Username	Penyakit
3	Nisa	Influenza
4	almer	-

>>> Masukkan perintah: lihatpasien

No.	Urutan berdasarkan
1.	ID
2.	Nama

>>> Pilihan: 1

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

>>> Pilihan: 2

### Daftar Pasien

Menampilkan pasien dengan ID terurut descending...

ID	Username	Penyakit
4	almer	-
3	Nisa	Influenza

Kasus 3 : Spesifik melihat data dokter

```
>>> Masukkan perintah: lihatdokter
```

No.	Urutan berdasarkan
1.	ID
2.	Nama
3.	Aura

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

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

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

### Daftar Dokter

```
Menampilkan dokter dengan ID terurut descending...
```

ID	Username	Aura
2	Zulfa	0

```
>>> Masukkan perintah: lihatdokter
```

No.	Urutan berdasarkan
1.	ID
2.	Nama
3.	Aura

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

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

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

#### Daftar Dokter

Menampilkan dokter dengan ID terurut ascending...

ID	Username	Aura
2	Zulfa	0

Kasus 4: Pilihan tidak valid

```
>>> Masukkan perintah: lihatuser
```

No.	Urutan berdasarkan
1.	ID
2.	Nama

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

ERROR: Pilihan tidak valid!

## F08 - Cari User

Kasus 1 : Mencari data user (dokter/pasien) berdasarkan ID

>>> Masukkan perintah: cariuser

No.	Cari berdasarkan
1.	ID
2.	Nama

>>> Pilihan: 1

>>> Masukkan nomor ID user: 4

Hasil Pencarian Pengguna

ID	Username	Role	Penyakit
4	Ziza	Pasien	-

Kasus 2 : User yang dicari tidak ditemukan

>>> Masukkan perintah: cariuser

No.	Cari berdasarkan
1.	ID
2.	Nama

>>> Pilihan: 1

>>> Masukkan nomor ID user: 10

Hasil Pencarian Pengguna

Pengguna dengan ID '10' tidak ditemukan.

Kasus 3 : Mencari data pasien

>>> Masukkan perintah: caripasien

No.	Cari berdasarkan
1.	ID
2.	Nama
3.	Penyakit

>>> Pilihan: 1

>>> Masukkan nomor ID user: 5

**Hasil Pencarian Pasien**

ID	Username	Penyakit
5	Fira fura	-

>>> Masukkan perintah: caripasien

No.	Cari berdasarkan
1.	ID
2.	Nama
3.	Penyakit

>>> Pilihan: 1

>>> Masukkan nomor ID user: 7

**Hasil Pencarian Pasien**

Pasien dengan ID '7' tidak ditemukan.

>>> Masukkan perintah: caripasien

No.	Cari berdasarkan
1.	ID
2.	Nama
3.	Penyakit

>>> Pilihan: 2

>>> Masukkan nama user: Ziza

**Hasil Pencarian Pasien**

ID	Username	Penyakit
4	Ziza	-

```
>>> Masukkan perintah: caripasien
```

No.	Cari berdasarkan
1.	ID
2.	Nama
3.	Penyakit

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

```
>>> Masukkan nama user: Nisa
```

Hasil Pencarian Pasien

```
Pasien dengan nama 'Nisa' tidak ditemukan.
```

```
>>> Masukkan perintah: caripasien
```

No.	Cari berdasarkan
1.	ID
2.	Nama
3.	Penyakit

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

```
>>> Masukkan nama penyakit: Maag
```

Hasil Pencarian Pasien

```
Pasien dengan penyakit 'Maag' tidak ditemukan.
```

Kasus 4 : Mencari data dokter

```
>>> Masukkan perintah: caridokter
```

No.	Cari berdasarkan
1.	ID
2.	Nama

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

```
>>> Masukkan nomor ID dokter: 5
```

Hasil Pencarian Dokter

```
Dokter dengan ID '5' tidak ditemukan.
```

```
>>> Masukkan perintah: caridokter
```

No.	Cari berdasarkan
1.	ID
2.	Nama

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

```
>>> Masukkan nomor ID dokter: 2
```

Hasil Pencarian Dokter

ID	Username	Aura
2	Zulfa	0

F09 - Lihat Antrian

>>> Masukkan perintah: lihatantrian

Status Antrian

A1	A2	A3
B1	B2	B3
C1	C2	C3

Antrian Ruangan A1

Kapasitas	3 orang
Dokter	Zulfa

Pasien di dalam ruangan:

1. Ziza

Pasien di antrian:

Tidak ada pasien di antrian

### Antrian Ruangan A2

Kapasitas	3 orang
Dokter	Alya

Pasien di dalam ruangan:

Tidak ada pasien

Pasien di antrian:

Tidak ada pasien di antrian

>>> Masukkan perintah: █

### F10 - Tambah Dokter

Kasus 1 : Dokter bernama **Azza** belum ada

```
>>> Masukkan perintah: tambahdokter
Masukkan username dokter: Azza
Masukkan password dokter: propleyer
Masukkan spesialisasi dokter: Jantung
Masukkan biaya checkup dokter: 99
SUKSES: Teks berhasil dienkripsi!
SUKSES: Dokter Azza berhasil ditambahkan!
```

Kasus 2 : Dokter bernama **Azza** sudah ada

```
>>> Masukkan perintah: tambahdokter
Masukkan username dokter: Azza
Masukkan password dokter: propleyer
Masukkan spesialisasi dokter: Paru
Masukkan biaya checkup dokter: 98
ERROR: Username sudah terdaftar!
```

Kasus 3 : Ruangan kosong dan dokter belum di assign di ruang manapun

```
>>> Masukkan perintah: assigndokter  
Masukkan username dokter: Azza  
Masukkan kode ruangan: C3  
SUKSES: Dokter Azza berhasil diassign ke ruangan C3!
```

Kasus 4 : Ruangan kosong dan dokter sudah di assign di ruangan lain

```
>>> Masukkan perintah: assigndokter  
Masukkan username dokter: Azza  
Masukkan kode ruangan: B2  
ERROR: Dokter Azza sudah diassign ke ruangan C3!
```

Kasus 5 : Ruangan tidak kosong dan dokter belum di assign di ruangan manapun

```
>>> Masukkan perintah: assigndokter  
Masukkan username dokter: Fans alpro  
Masukkan kode ruangan: C3  
ERROR: Dokter Azza sudah menempati ruangan C3!  
ERROR: Silakan cari ruangan lain untuk dokter Fans alpro.
```

Kasus 6 : Ruangan tidak kosong dan dokter sudah di assign di ruangan lain

```
>>> Masukkan perintah: assigndokter  
Masukkan username dokter: Fans alpro  
Masukkan kode ruangan: C3  
ERROR: Dokter Fans alpro sudah menempati ruangan C1!  
ERROR: Ruangan C3 juga sudah ditempati dokter Azza!
```

## F11 - Diagnosis

Kasus 1: Dokter memiliki pasien yang perlu diperiksa

```
>>> Masukkan perintah: diagnosis  
dindun bud terdiagnosa penyakit COVID-19!
```

Kasus 2: Pasien tidak terjangkit penyakit apapun setelah diperiksa

```
>>> Masukkan perintah: diagnosis  
SUKSES: Pasien tidak terdiagnosis penyakit apapun!
```

Kasus 3: Antrian sudah kosong dan tidak ada pasien yang perlu diperiksa

```
>>> Masukkan perintah: diagnosis  
ERROR: Ruangan dokter ini tidak memiliki pasien yang mengantri.
```

F12 - Ngobatin

Kasus 1: Pasien memiliki penyakit tertentu

```
>>> Masukkan perintah: ngobatin  
Dokter sedang mengobati pasien!  
Pasien memiliki penyakit Influenza
```

Obat yang harus diberikan:

No.	Nama Obat
1.	Oseltamivir
2.	Vitamin C

F13 - Aku boleh pulang ga, dok?

Kasus 1: Pasien belum diberikan diagnosa penyakit

```
>>> Masukkan perintah: pulangdok
```

Status Pulang

Status	Keterangan
Belum Diagnosa	Anda belum didiagnosa oleh dokter.

ERROR: Anda belum boleh pulang!

Kasus 2: Pasien belum diberikan resep obat

>>> Masukkan perintah: pulangdok

Status Pulang

Status	Keterangan
Belum Diberi Obat	Anda belum diberikan resep obat.

ERROR: Anda belum boleh pulang!

### Kasus 3: Pasien belum menghabiskan seluruh obat

>>> Masukkan perintah: pulangdok

Status Pulang

Status	Keterangan
Belum Minum Semua Obat	Anda belum mengonsumsi semua obat yang telah diresepkan.

ERROR: Anda belum boleh pulang!

### Kasus 4: Pasien sudah menghabiskan obat dan semuanya valid

>>> Masukkan perintah: pulangdok

Status Pulang

Status	Keterangan
Boleh Pulang	Anda telah selesai menjalani perawatan.

SUKSES: Selamat, Anda boleh pulang!

### Kasus 5: Pasien telah didiagnosa dan belum diberi obat

>>> Masukkan perintah: pulangdok

Status Pulang

Status	Keterangan
Belum Diberi Obat	Anda belum diberikan resep obat.

ERROR: Anda belum boleh pulang!

#### F14 - Daftar Check-Up

##### Kasus 1 : Pasien belum terdaftar dalam antrian

>>> Masukkan perintah: daftarcheckup  
Suhu Tubuh (Celcius): 37  
Tekanan Darah Sistol (mmHg): 100  
Tekanan Darah Diastol (mmHg): 80  
Detak Jantung (bpm): 80  
Saturasi Oksigen (%): 96  
Kadar Gula Darah (mg/dL): 150  
Berat Badan (kg): 70  
Tinggi Badan (cm): 180  
Kadar Kolesterol (mg/dL): 200  
Trombosit (ribu/ $\mu$ L): 200

Daftar Dokter yang Tersedia:

No	Dokter	Spesialisasi	Ruangan	Antrian	Aura	Biaya Checkup
1.	Azza	Jantung	C3	0	0	99.00
2.	Fans alpro	Penyakit Dalam	C1	0	0	88.00

Pilih dokter (1 - 2): 2

Pendaftaran Checkup

Dokter	Ruangan
Fans alpro	C1

[ | Info]: Anda telah dipindahkan langsung ke ruangan dokter!

SUKSES: Pendaftaran checkup untuk dindun bud pada dr.Fans alpro di ruangan C1 berhasil!

##### Kasus 2 : Role selain pasien yang mengakses fitur

```
>>> Masukkan perintah: daftarcheckup
Suhu Tubuh (Celcius): 37
Tekanan Darah Sistol (mmHg): 100
Tekanan Darah Diastol (mmHg): 80
Detak Jantung (bpm): 80
Saturasi Oksigen (%): 96
Kadar Gula Darah (mg/dL): 150
Berat Badan (kg): 70
Tinggi Badan (cm): 180
Kadar Kolestrol (mg/dL): 200
Trombosit (ribu/µL): 200
ERROR: Akses ditolak! Hanya Pasien yang dapat mendaftar checkup.
```

### Kasus 3 : Input tidak valid

```
>>> Masukkan perintah: daftarcheckup
Suhu Tubuh (Celcius): 89
ERROR: Data kesehatan 'Suhu Tubuh (Celcius)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Suhu Tubuh (Celcius): 35
Tekanan Darah Sistol (mmHg): 7
ERROR: Data kesehatan 'Tekanan Darah Sistol (mmHg)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Tekanan Darah Sistol (mmHg): 100
Tekanan Darah Diastol (mmHg): 6
ERROR: Data kesehatan 'Tekanan Darah Diastol (mmHg)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Tekanan Darah Diastol (mmHg): 100
Detak Jantung (bpm): 17
ERROR: Data kesehatan 'Detak Jantung (bpm)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Detak Jantung (bpm): 100
Saturasi Oksigen (%): 7
ERROR: Data kesehatan 'Saturasi Oksigen (%): ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Saturasi Oksigen (%): 100
Kadar Gula Darah (mg/dL): 12
ERROR: Data kesehatan 'Kadar Gula Darah (mg/dL)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Kadar Gula Darah (mg/dL): 100
Berat Badan (kg): 12
ERROR: Data kesehatan 'Berat Badan (kg)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Berat Badan (kg): 100
Tinggi Badan (cm): 12
ERROR: Data kesehatan 'Tinggi Badan (cm)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Tinggi Badan (cm): 180
Kadar Kolestrol (mg/dL): 2
ERROR: Data kesehatan 'Kadar Kolestrol (mg/dL)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Kadar Kolestrol (mg/dL): 100
Trombosit (ribu/µL): 18
ERROR: Data kesehatan 'Trombosit (ribu/µL)': ' tidak valid! Harap masukkan nilai dalam rentang yang benar.
Trombosit (ribu/µL): 100
ERROR: Akses ditolak! Hanya Pasien yang dapat mendaftar checkup.
```

### Kasus 4 : Pasien sudah terdaftar dalam antrian

```
>>> Masukkan perintah: daftarcheckup
Suhu Tubuh (Celcius): 37
Tekanan Darah Sistol (mmHg): 100
Tekanan Darah Diastol (mmHg): 100
Detak Jantung (bpm): 100
Saturasi Oksigen (%): 100
Kadar Gula Darah (mg/dL): 100
Berat Badan (kg): 100
Tinggi Badan (cm): 100
Kadar Kolestrol (mg/dL): 100
Trombosit (ribu/µL): 100
ERROR: Anda sudah terdaftar dalam antrian!
```

## F15 - Antrian Saya!

### Kasus 1: Pasien sudah berada di ruangan

```
>>> Masukkan perintah: antriansaya  
Anda sedang berada di dalam ruangan dokter! Silakan menyelesaikan pemeriksaan.
```

### Kasus 2: Pasien sudah terdaftar dalam antrian

```
>>> Masukkan perintah: antriansaya
```

```
Status antrian Anda:
```

Dokter	Alya
Ruangan	A2
Posisi antrian	1/2

### Kasus 3: Pasien belum terdaftar dalam antrian

```
>>> Masukkan perintah: antriansaya  
ERROR: Anda belum terdaftar dalam antrian check-up! Silakan daftar terlebih dahulu dengan command DAFTAR_CHECKUP.
```

## F16 - Minum Obat

### Kasus 1: Saat obat yang dipilih tidak ada

```
>>> Masukkan perintah: minumobat
```

```
Daftar Obat yang Diresepkan
```

```
[ 📱 | Info] - Daftar obat yang harus diminum sesuai dengan urutan
```

Urutan Minum	Nama Obat
1	Oseltamivir
2	Vitamin C

```
>>> Pilih obat untuk diminum: 3
```

```
ERROR: Pilihan nomor untuk obat tidak tersedia!
```

### Kasus 2: Obat yang dipilih ada dan sesuai urutan

```
>>> Masukkan perintah: minumobat
Daftar Obat yang Diresepkan
[ 📄 | Info] - Daftar obat yang harus diminum sesuai dengan urutan

Urutan Minum | Nama Obat
---|---
1 | Oseltamivir
2 | Vitamin C

>>> Pilih obat untuk diminum: 1
SUKSES: Obat berhasil diminum!

>>> Masukkan perintah: minumobat
Daftar Obat yang Diresepkan
[ 📄 | Info] - Daftar obat yang harus diminum sesuai dengan urutan

Urutan Minum | Nama Obat
---|---
2 | Vitamin C

>>> Pilih obat untuk diminum: 2
SUKSES: Obat berhasil diminum!
```

### Kasus 3: Daftar obat setelah 1 obat diminum

```
>>> Masukkan perintah: minumobat
Daftar Obat yang Diresepkan
[ 📄 | Info] - Daftar obat yang harus diminum sesuai dengan urutan

Urutan Minum | Nama Obat
---|---
2 | Vitamin C

>>> Pilih obat untuk diminum: 2
SUKSES: Obat berhasil diminum!
```

### Kasus 4: Obat sudah diminum semua

```
>>> Masukkan perintah: minumobat
ERROR: Semua obat yang diresepkan sudah dikonsumsi!
```

### Kasus 5: Belum diberikan resep obat

```
>>> Masukkan perintah: minumobat  
ERROR: Anda belum diberikan resep obat!
```

#### F17 - Minum Penawar

##### Kasus 1: Terdapat obat yang terakhir diminum

```
>>> Masukkan perintah: minumpenawar
```

Konsumsi Penawar

Obat Dibatalkan	Nyawa Tersisa	Status
Vitamin C	• • o	Penawar berhasil

```
SUKSES: Penawar berhasil! Obat Vitamin C dibatalkan.
```

##### Kasus 2: Belum ada obat yang diminum

```
>>> Masukkan perintah: minumpenawar
```

```
ERROR: Anda belum mengonsumsi obat apapun!
```

#### F18 - Exit

##### Kasus 1 : Keluar program

```
>>> Masukkan perintah: exit
```

Keluar Program

```
SUKSES: Terima kasih telah menggunakan Sistem Manajemen Rumah Sakit Nimons!
```

Sampai Jumpa!

```
SUKSES: Memori rumah sakit berhasil dibebaskan!
```

##### Kasus 2 : Input tidak valid

```
>>> Masukkan perintah: exiit  
ERROR: Perintah tidak dikenal! Ketik 'MENU' untuk bantuan.
```

## B02 - Denah Dinamis

### Kasus 1: Denah diubah tanpa memengaruhi keberadaan dokter di ruangan

```
>>> Masukkan perintah: ubahdenah  
Masukkan jumlah baris: 6  
Masukkan jumlah kolom: 10  
SUKSES: Denah rumah sakit berhasil diubah!
```

```
>>> Masukkan perintah: lihatdenah
```

Denah Rumah Sakit

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
F1	F2	F3	F4	F5	F6	F7	F8	F9	F10

### Kasus 2: Denah diubah dengan memengaruhi keberadaan dokter di ruangan

```
>>> Masukkan perintah: ubahdenah  
Masukkan jumlah baris: 1  
Masukkan jumlah kolom: 1  
ERROR: Ruangan A3 masih ditempati oleh Dr.Abelbel
```

## B03 - Aura!

### Kasus 1: Dokter tersedia

>>> Masukkan perintah: lihatdokter

No.	Urutan berdasarkan
1.	ID
2.	Nama
3.	Aura

>>> Pilihan: 3

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

>>> Pilihan: 1

### Daftar Dokter

Menampilkan dokter dengan aura terurut ascending...

ID	Username	Aura
2	Zulfa	0
3	Almer	0

Kasus 2: Tidak ada dokter yang terdaftar

>>> Masukkan perintah: lihatdokter

No.	Urutan berdasarkan
1.	ID
2.	Nama
3.	Aura

>>> Pilihan: 3

No.	Urutan Sorting
1.	ASC (A-Z)
2.	DESC (Z-A)

>>> Pilihan: 2

Daftar Dokter

Tidak ada dokter terdaftar.

B04 - Banarich!!

Kasus 1: Jumlah banarich yang dimiliki pasien

>>> Masukkan perintah: lihatdompet

Saldo Dompet

Pasien	Saldo
Ziza	120.00 BananaRich

Kasus 2: Jumlah banarich milik rumah sakit

```
>>> Masukkan perintah: lihatfinansial
```

### Finansial Rumah Sakit

Entitas	Saldo
Rumah Sakit	10003.00 BananaRich

### Kasus 3: Banarich pasien tidak cukup untuk daftar check-up

```
>>> Masukkan perintah: login  
Masukkan username: Ziza  
Masukkan password: 123456  
SUKSES: Teks berhasil dienkripsi!  
SUKSES: Login berhasil sebagai Ziza! (Peran: Pasien)
```

```
>>> Masukkan perintah: daftarcheckup  
Suhu Tubuh (Celcius): 38  
Tekanan Darah Sistol (mmHg): 100  
Tekanan Darah Diastol (mmHg): 100  
Detak Jantung (bpm): 100  
Saturasi Oksigen (%): 100  
Kadar Gula Darah (mg/dL): 100  
Berat Badan (kg): 100  
Tinggi Badan (cm): 100  
Kadar Kolesterol (mg/dL): 100  
Trombosit (ribu/µL): 100
```

Daftar Dokter yang Tersedia:

No	Dokter	Spesialisasi	Ruangan	Antrian	Aura	Biaya Checkup
1.	Zulfa	Bedah	A1	0	0	1.00
2.	Abelbel	Jantung	A3	0	0	14.00
3.	Nadia	Paru	A5	0	0	1000.00

Pilih dokter (1 - 3): 3

ERROR: Saldo BananaRich tidak cukup untuk checkup!

### Kasus 4: Gacha banarich pasien berhasil

```
>>> Masukkan perintah: gacha
```

### Hasil Gacha

Selamat datang Ziza, di Mesin Gacha XXeon06!

[🎉 YAY] - Selamat! Anda mendapatkan 15.00 BananaRich!

### Kasus 5: Gacha banarich pasien gagal

```
>>> Masukkan perintah: gacha
```

```
Hasil Gacha
```

Selamat datang Ziza, di Mesin Gacha XXeon06!

[ | Yahhh] - Kamu kurang beruntung, coba lagi nanti!

## B05 - Dead or Alive?!

### Kasus 1: Saat minum obat tidak sesuai urutan

```
>>> Masukkan perintah: minumobat
```

```
Daftar Obat yang Diresepkan
```

[ | Info] - Daftar obat yang harus diminum sesuai dengan urutan

Urutan Minum	Nama Obat
1	Oseltamivir
2	Vitamin C

```
>>> Pilih obat untuk diminum: 2
```

SUKSES: Obat berhasil diminum!

[ | Info] - Obat Salah! Nyawa berkurang satu.

→ Sisa nyawa:● ● ○

### Kasus 2: Pasien kehabisan nyawa

```
>>> Masukkan perintah: minumobat
```

```
Daftar Obat yang Diresepkan
```

[ | Info] - Daftar obat yang harus diminum sesuai dengan urutan

Urutan Minum	Nama Obat
1	Oseltamivir

```
>>> Pilih obat untuk diminum: 2
```

SUKSES: Obat berhasil diminum!

ERROR: Pasien kehabisan nyawa! Pasien akan dihapus.

SUKSES: Pasien dengan username 'Nisa' berhasil dihapus dari sistem!

## B06 - Mainin Antrian

### Kasus 1: Lewati antrian melalui manajer

```
>>> Masukkan perintah: login  
Masukkan username: NimonsDawg  
Masukkan password: admoontothemoon  
SUKSES: Teks berhasil dienkripsi!  
SUKSES: Login berhasil sebagai NimonsDawg! (Peran: Manajer)  
  
>>> Masukkan perintah: lewatiantrian  
Masukkan username pasien yang antriannya akan dilewati: Bunga  
SUKSES: Pasien 'Bunga' di ruangan A2 berhasil dipindahkan ke posisi pertama dalam antrian.
```

```
>>> Masukkan perintah: lihatantrian
```

Status Antrian

A1	A2	A3
B1	B2	B3
C1	C2	C3

Antrian Ruangan A2

Kapasitas	3 orang
Dokter	Zulfa

Pasien di dalam ruangan:

1. Nisa
2. Ziza
3. Alya

Pasien di antrian:

1. Bunga
2. Fira

### Kasus 2: Batalkan antrian melalui manajer

```
>>> Masukkan perintah: batalkanantrian  
Masukkan username pasien yang antriannya akan dibatalkan: Fira  
SUKSES: Antrian untuk pasien Fira di ruangan A2 berhasil dibatalkan.
```

```
>>> Masukkan perintah: lihatantrian
```

```
Status Antrian
```

A1	A2	A3
B1	B2	B3
C1	C2	C3

```
Antrian Ruangan A2
```

Kapasitas	3 orang
Dokter	Zulfa

Pasien di dalam ruangan:

1. Nisa
2. Ziza
3. Alya

Pasien di antrian:

1. Bunga

#### Kasus 3: Lewati antrian melalui pasien

```
>>> Masukkan perintah: lewatiantrian
```

```
SUKSES: Pasien 'Fira' di ruangan A2 berhasil dipindahkan ke posisi pertama dalam antrian.
```

#### Kasus 4: Batalkan antrian melalui pasien

```
>>> Masukkan perintah: batalkanantrian
```

```
SUKSES: Antrian untuk pasien Lona di ruangan A2 berhasil dibatalkan.
```

#### B07 - Search Suggestion

##### Kasus 1: Terdapat user yang dapat direkomendasikan

```
>>> Masukkan perintah: cariuser
```

No.	Cari berdasarkan
1.	ID
2.	Nama

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

```
>>> Masukkan nama user: fira
```

Hasil Pencarian Pengguna

Tidak ada hasil pencocokan pasti. Mungkin maksud Anda:

ID	Username	Role	Penyakit
5	Fira fura	Pasien	-

## LAMPIRAN

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

Nomor Asistensi : 1  
No. Kelompok/Kelas : I/04  
Tanggal asistensi : Rabu, 30 April 2025

Anggota kelompok	NIM / Nama (Hanya yang Hadir)
<u>1</u>	18224080 Anisa Aulia Alhaqi
<u>2</u>	18224064 Muhammad Zulfa Fauzan Nurhuda
<u>3</u>	18224094 Endda Tsa Azzahra Syaifur
<u>4</u>	18224016 Abel Gani
<u>5</u>	18224070 Almer Zain Farisseno

Asisten pembimbing	NIM / Nama
	13522129 Hugo Sabam Augusto

**Catatan Asistensi:**

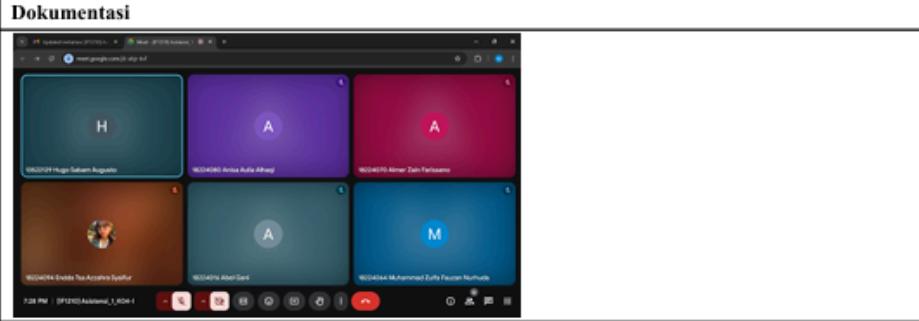
**Rangkuman Diskusi**

Diskusi ini membahas beberapa hal penting dalam mengelola proyek. Pertama, F18 itu wajib (untuk F1 hingga F18 wajib), tapi nggak perlu disimpan ke file (khusus untuk STI). Disarankan untuk merencanakan struktur ADT dengan baik supaya tidak ada kesalahan di step selanjutnya, termasuk memilih ADT dan atributnya dengan tepat. Lebih baik file-file untuk setiap fitur dipisah supaya lebih modular. Selain itu, penting untuk menguasai WSL dan Makefile, karena Makefile dipakai untuk menjalankan program, jadi tidak perlu ngetik perintah gcc berulang-ulang dan panjang. Program akan langsung menghandle input saat dijalankan, dengan satu file yang mengatur input dan output (sebagai manajer utama). Untuk perintah di program, kata-katanya bisa berbeda-beda asal masih relevan. Terakhir, repo proyek harus diupload pakai template dari repo labpro, dan jangan lupa untuk ngerjain bonus 1 buat tambahan nilai.

**Tindak Lanjut**

Membuat pembagian tugas untuk mencari tahu lebih dalam di tiap fitur memerlukan ADT apa saja (serta atributnya apa saja). Berikut adalah pembagian tugasnya:

- Azza (F01 - F04)
- Nisa (F05 - F8)
- Zulfa (F9 - F12)
- Abel (F13 - F16)
- Almer (F17, F18, S01, dan S02)



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

Nomor Asistensi	:	2
No. Kelompok/Kelas	:	1/04
Tanggal asistensi	:	22 Mei 2025
Anggota kelompok	<b>NIM / Nama (Hanya yang Hadir)</b>	
	1	18224080 Anisa Aulia Alhaqi
	2	18224064 Muhammad Zulfa Fauzan Nurhuda
	3	18224094 Endda Tsa Azzahra Syaiful
	4	18224016 Abel Gani
	5	18224070 Almer Zain Farissenno
Asisten pembimbing	<b>NIM / Nama</b>	
	13522129 Hugo Sabam Augusto	

**Catatan Asistensi:**

**Rangkuman Diskusi**

**Feedback penggerjaan Milestone 1:**

Sudah cukup baik, catatannya hanya pada format notasi algoritmik yang kurang selaras.

**Demonstrasi:**

Belum ada teknis yang lengkap, tetapi pelajari kode yang dikerjakan oleh masing-masing anggota.

**Tambahan:**

Akan ada revisi besar-besaran, jadi pantau spek atau dokumen IF1210 Spesifikasi Tugas Besar

**Tindak Lanjut**

- Menyelaraskan format notasi algoritmik
- Mempelajari kode masing-masing
- Memantau dokumen spesifikasi tugas besar
- Menyelesaikan progres penggerjaan milestone 2

**Dokumentasi**

