LAPORAN MILESTONE 1 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

DAFTAR TABEL

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	Variabel yang terlibat	Deskripsi Implementasi	Alasan Implementasi
Prosedur	F01 - Login	userID-int, username - string, password - string, role - string, status_login - boolean	Digunakan untuk mengakses akun, dengan username serta password yang sesuai.	Prosedur digunakan untuk pengecekan kesesuaian antara password dan username.
Prosedur, ADT List	F02 - Register	username - string, password - string, status_register - boolean	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	status_login - boolean	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	username - string, kode_unik - string, newpassword - string	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	kapasitas_ruangan - integer	Digunakan untuk melihat denah	Denah berbentuk memanjang dan

		dokter_ruangan - string pasien_ruangan - string	ruangan di rumah sakit beserta detail dari ruangan	terurut
Sort	F07 - Lihat User	ID - integer nama_dokter - string nama_pasien - string jenis_penyakit_pasien - string jenis_penyakit_dokter - string	Digunakan untuk melihat data seluruh pengguna, baik dokter maupun pasien	Perlu melakukan pengurutan data pasien, baik *ascending* maupun *descending*
Binary Search	F08 - Cari User	ID - integer nama_dokter - string nama_pasien - string jenis_penyakit_pasien - string	Digunakan untuk mencari data pengguna (dokter, pasien) secara spesifik berdasarkan ID atau Nama	Melakukan pencarian data dengan jumlah yang besar
ADT Queue, Prosedur	F09 - Lihat Antrian	ID - integer kapasitas - integer nama_dokter - string nama_pasien - string jumlah_pasien_dalam_ruangan - integer jumlah_pasien_dalam_antrian - integer	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	ID - integer nama_dokter - string spesialisasi - string	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.
ADT List, Prosedur	F11 - Diagnosis	ID - integer nama_obat - string urutan_minum - integer penyakit_id - integer	Prosedur digunakan untuk mengolah data pasien berdasarkan template kasus yang ada, sehingga dapat terjadi diagnosis otomatis.	
ADT List, Prosedur	F12 - Ngobatin	ID - integer nama_penyakit - string	Prosedur bertugas mengotomatisasi pengobatan	Karena ada korelasi antara id obat dan id penyakit dalam

		id_penyakit - integer id_obat - integer nama_obat - string	berdasarkan pencocokan id obat dengan id penyakit	pengobatan otomatis yang dijalankan dalam fungsi ini.
ADT List	F13 - Aku boleh pulang ga, dok?	nama_obat - string	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	id_pasien - integer nama_pasien - string (kondisi_tubuh_pasien)	Digunakan untuk memilih dokter lalu *assign* 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!	nama_dokter - string ruangan - string antrian - integer	Digunakan untuk mendaftarkan antrian pasien	Karena pasien pertama akan dilayani dulu (FIFO)
ADT Stack	F16 - Minum Obat	nama_obat - string	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	nama_obat - string	Digunakan untuk menawarkan atau mengeluarkan obat yang terakhir kali diminum	Supaya obat yang terakhir diminum ada di urutan awal dan mudah untuk dikeluarkan
Prosedur	F18 - Exit	-	Keluar dari program	Mengakhiri program

PEMBAGIAN KERJA ANGGOTA

Tabel 2. Pembagian Kerja Anggota

Fitur	Implementasi *)	NIM Desainer **)	NIM Coder **)	NIM Tester **)
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
F10 - Tambah Dokter	Prosedur	18224064	18224064	18224016, 18224064, 18224070, 18224080, 18224094
F18 - Exit	Prosedur	18224070	18224070	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
F18 - Exit	v	v	v

SPESIFIKASI PROGRAM

F01 - Login

```
procedure loginAkun(input: UserList list, string usn, string pass,
output: boolean)
{berfungsi untuk login akun user}

KAMUS LOKAL
UserList    : array of User
usn, pass   : string
true, false : boolean

ALGORITMA
for (i = 0 ; i < nEff ; i ++)
    if list.elements[i].username = usn and
list.elements[i].password = pass then
        print ("Login berhasil!")
        if list.elements[i].role = "PASIEN" then
            print ("Selamat datang, pasien " + usn + "! Ada keluhan
apa?")
        else if list.elements[i].role = "DOKTER" then
            print ("Selamat datang, dokter " + usn + "!")
        else if list.elements[i].role = "MANAJER" then
            print ("Selamat datang, Manajer!")
        return true

    else if list.elements[i].username = usn then
        print ("Username atau password salah!")
        return false
print ("Maaf, tetapi tidak ada pengguna dengan username " + usn)
return false
```

F02 - Register Pasien

```
procedure registerAkun(input: UserList list, string usn, string
pass, output: boolean)
{berfungsi untuk register akun}

KAMUS LOKAL
UserList    : array of user
usn, pass   : string
```

```
true, false : boolean

ALGORITMA
for (i = 0 ; i < nEff ; i ++)
    if list.elements[i].username = usn then
        return false // Username sudah ada, registrasi gagal

User newUser
newUser.userID ← list.nEff + 1
newUser.username ← usn
newUser.password ← pass
newUser.role ← "PASIEN"  // Sudah pasti pasien

list.elements[list.nEff] ← newUser
list.nEff ← list.nEff + 1
return true
```

F03 - Logout

```
procedure logout(output Session : session)
{mengeluarkan output berupa pernyataan user telah logout}

KAMUS
isLoggedIn  : boolean

ALGORITMA
{ inisialisasi boolean }
isLoggedIn ← true

output(“>>> LOGOUT”)
if (!session->isLoggedIn)
    output(“Logout gagal!”)
    output(“Anda belum login, silahkan login terlebih dahulu
sebelum melakukan logout”)

else {kondisi jika boolean isLoggedIn bernilai true}
    session->isLoggedIn ← false
    session->userID[0] ← ‘\0’
    session->username[0] ← ‘\0’
    session->role[0] ← ‘\0’
    {mengosongkan list user yang berhasil logout yang terdiri dari
ID, username, dan role}
    output(“Sampai jumpa!”)
```

F04 - Lupa Password

```
procedure runLengthEncoding(input username : char, output rle :
char)
{menghasilkan kode unik dari username yang diinput oleh user}
{
    I.S Mengambil username pengguna
    F.S Menghasilkan kode unik dengan algoritma run length encoding
}
KAMUS LOKAL
currentChar : char
count       : integer
tempCount   : integer
temp        : char
countStr    : array of [1..12] of char
outIndex    : integer
usernameLen : integer
i           : integer
j           : integer
rle         : char
numDigits   : integer

ALGORITMA
{ inisialisasi variabel }
outIndex ← 0
usernameLen ← length(username)
i ← 0
j ← 0

while (i < usernameLen) do
    currenChar ← username[i]
    count ← 0

    while (i < usernameLen and username[i] = currentChar) do
       count ← count + 1
       i ← i + 1

    rle[outIndex++] ← currentChar
    if (count > 1) then
        numDigits ← 0
        tempCount ← count

        while (tempCount > 0) do
           countStr[numDigits++] ← (tempCount%10) + '0'
```

```
            tempCount ← tempCount/10

        while (j + 1 < numDigits / 2) do
            temp ← countStr[j]
            countStr[j] ← countStr[numDigits - 1 - j]
            countStr[numDigits - 1 - j] ← temp

        countStr[numDigits] ← '\0'
        rle + outIndex ← countStr
        outIndex ← outIndex + numDigits

rle[outIndex] ← '\0'

procedure lupaPassword(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
username : array of [1..50] of char
input    : array of [1..50] of char
kodeUnik : array of [1..50] of char
newPass  : array of [1..100] of char
found    : integer
i        : integer

ALGORITMA
{ inisialisasi variabel }
found ← -1

output(">>> LUPA_PASSWORD")
output("Username: ")
input(username)

while (i + 1 < users->nEff) do
    if(users->elements[i].username = username) then
        found ← i
        stop

runLengthEncoding(input username : char , output kodeUnik : char)

output("Kode Unik: ")
```

```
input(input)

if (found = -1) then
    output(“Username tidak terdaftar!”)
if (input != kodeUnik) then
    output(“Kode unik salah!”)

output(“Halo <users->elements[found].role> <username>, silahkan
daftarkan ulang password anda!”)
output(“Password Baru: “)

input(newPass)

newPass ← users->elements[found].password
```

F05 - Menu & Help

```
procedure displayMenu(input session : Session)
{
    I.S Mengambil role pengguna
    F.S Menampilkan layar menu dan help sesuai dengan role pengguna
}

KAMUS LOKAL
-

ALGORITMA
output(“>>>HELP”)
output(“=========== HELP ===========”)

if (!session.isLoggedIn) then
      output("Kamu belum login sebagai role apapun. Silahkan login
      terlebih dahulu.\n\n")
      output("LOGIN: Masuk ke dalam akun yang sudah terdaftar\n")
      output("REGISTER: Membuat akun baru\n")
else
       if (session.role = DOCTOR) then
             output("Halo Dokter <session.username>. Kamu memanggil
             command HELP. Kamu pasti sedang kebingungan. Berikut
             adalah hal-hal yang dapat kamu lakukan sekarang:\n\n")
             output("LOGOUT: Keluar dari akun yang sedang
             digunakan\n")
```

```
            output("DIAGNOSIS: Melakukan diagnosis penyakit pasien
            berdasarkan kondisi tubuh pasien\n")
      else if (session.role = PATIENT) then
            output("Selamat datang, <session.username>. Kamu
            memanggil command HELP. Kamu pasti sedang kebingungan.
            Berikut adalah hal-hal yang dapat kamu lakukan
            sekarang:\n\n")
            output("LOGOUT: Keluar dari akun yang sedang
            digunakan\n")
            output("DAFTAR_CHECKUP: Mendaftarkan diri untuk
            pemeriksaan dokter\n")
      else if (session.role = MANAGER) then
             output("Halo Manager <session.username>. Kenapa kamu
             memanggil command HELP? Kan kamu manager, tapi
             yasudahlah kamu pasti sedang kebingungan. Berikut
             adalah hal-hal yang dapat kamu lakukan sekarang:\n\n")
             output("LOGOUT: Keluar dari akun yang sedang
             digunakan\n")
             output("TAMBAH_DOKTER: Mendaftarkan dokter baru ke
             sistem\n")
             output("ASSIGN_DOKTER: Menugaskan dokter ke ruangan
             tertentu\n")
             output("LIHAT_DOKTER: Melihat daftar dokter\n")
             output("LIHAT_PASIEN: Melihat daftar nama-nama
             pasien\n")
             output("LIHAT_RUANGAN: Melihat detail ruangan
             tertentu\n")
             output("LIHAT_USER: Melihat daftar seluruh pengguna\n")
             output("LIHAT_PASIEN: Melihat daftar pasien\n")
             output("LIHAT_DOKTER: Melihat daftar dokter\n")
             output("CARI_USER: Mencari pengguna berdasarkan
             kriteria\n")
             output("CARI_PASIEN: Mencari pasien berdasarkan
             kriteria\n")
             output("CARI_DOKTER: Mencari dokter berdasarkan
             kriteria\n\n")

Footnote()
```

```
procedure printFootnote
{
    Menampilkan footnote
}
```

```
KAMUS LOKAL
-

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

F06 - Denah Rumah Sakit

```
procedure displayRoomDetails(input/output hospital: Hospital, input
session: Session)
{ Menampilkan detail ruangan berdasarkan kode input. }
---
KAMUS LOKAL:
  roomCode                  : string
  room                      : pointer to Room
  i, j                      : integer
  doctorFound, patientFound : boolean
  patientID                 : integer

ALGORITMA:
  if (!session.isLoggedIn) then
    output("Akses ditolak! Anda harus login terlebih dahulu.")
    return

  output("Masukkan kode ruangan: ")
  input(roomCode)

  // Cari ruangan berdasarkan kode
  room ← NULL
  i traversal [0..hospital.layout.rowEff - 1]
    j traversal [0..hospital.layout.colEff - 1]
      if (hospital.layout.elements[i][j].code = roomCode) then
        room ← address of hospital.layout.elements[i][j]
        break

  if (room = NULL) then
    output("Ruangan ", roomCode, " tidak ditemukan!")
    return

  { Tampilkan detail ruangan }
  output("\n--- Detail Ruangan ", roomCode, " ---")
  output("Kapasitas  : ", room.capacity)

  { Cari nama dokter  }
```

```
  output("Dokter     : ")
  if (room.doctorID = -1) then
    output("-")
  else
    doctorFound ← false
    i traversal [0..hospital.doctors.nEff - 1]
      if (hospital.doctors.elements[i].id = room.doctorID) then
        output("Dr. ", hospital.doctors.elements[i].username)
        doctorFound ← true
        break
    if (!doctorFound) then
      output("(Tidak ditemukan)")

  { Tampilkan pasien di ruangan }
  output("Pasien di dalam ruangan:")
  if (room.patientInRoom.nEff = 0) then
    output("  Tidak ada pasien di dalam ruangan saat ini.")
  else
    i traversal [0..room.patientInRoom.nEff - 1]
      patientID ← room.patientInRoom.patientID[i]
      patientFound ← false
      j traversal [0..hospital.patients.nEff - 1]
        if (hospital.patients.elements[j].id = patientID) then
          output("  ", i + 1, ". ",
hospital.patients.elements[j].username)
          patientFound ← true
          break
      if (!patientFound) then
        output("  ", i + 1, ". (Pasien tidak ditemukan)")
  output("------------------------------")
```

```
procedure displayLayout(input hospital: Hospital, input session:
Session)
{ Menampilkan denah rumah sakit dalam bentuk grid matrix. }
---
KAMUS LOKAL:
  i, j : integer

ALGORITMA:
  if (!session.isLoggedIn) then
    output("Akses ditolak! Anda harus login terlebih dahulu.")
    return

  if (hospital.layout.rowEff ≤ 0 OR hospital.layout.colEff ≤ 0)
then
    output("Denah rumah sakit kosong!")
    return

  // Tampilkan header kolom (1, 2, 3, ...)
  output("      ")
  j traversal [0..hospital.layout.colEff - 1]
```

```
    output(j + 1, "    ")  // Angka kolom dengan spasi 5 karakter

  // Gambar garis horizontal
  output("   +")
  j traversal [0..hospital.layout.colEff - 1]
    output("-----+")

  // Tampilkan baris dan kode ruangan
  i traversal [0..hospital.layout.rowEff - 1]
    output(" ", 'A' + i, " |")  // Header baris (A, B, C, ...)
    j traversal [0..hospital.layout.colEff - 1]
      output(" ", hospital.layout.elements[i][j].code, " |")  //
Kode ruangan
    output("   +")
    j traversal [0..hospital.layout.colEff - 1]
      output("-----+")
```

F07 - Lihat User

```
procedure validateInputInteger(input integer : input)
{melakukan validasi terhadap input agar sesuai nilai min dan maks
yang seharusnya}

KAMUS LOKAL
c                   : char
min                 : integer
maks                : integer

ALGORITMA
while (input < min or input > max) do
    output("Input tidak valid. Silakan masukan ulang.")
    output(">>> Pilihan: ")
    input(input)

function printViewOptions() → integer
{melakukan output untuk pilihan melihat data user, pasien, atau
dokter}

KAMUS LOKAL
choice              : integer

ALGORITMA
output("Melihat data apa?")
output("1. User")
output("2. Pasien")
```

```
output("3. Dokter")
output(">>> Pilihan: ")
input(choice)
ValidateInputInteger(choice, 1, 3)
output(">>> LIHAT")
depend on (choice)
    1 : output("User")
    2 : output("Pasien")
    3 : output("Dokter")
→ choice

function printSortBy() → integer
{melakukan output untuk pilihan pengurutan data, ID atau Nama}

KAMUS LOKAL
choice              : integer

ALGORITMA
output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")
input(choice)
ValidateInputInteger(choice, 1, 3)
→ choice

function printSortOrder() → integer
{melakukan output untuk pilihan pengurutan menaik atau menurun}

KAMUS LOKAL
choice              : integer

ALGORITMA
output("Urutkan sort?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")
input(choice)
ValidateInputInteger(choice, 1, 3)
→ choice

function compareUserByIdAsc(a : pointer to User, b : pointer to
User) → integer
```

```
{melakukan perbandingan untuk mendukung fungsi qsort secara
ascending}

KAMUS LOKAL

ALGORITMA
const User ua ← (const User*) a
const User ub ← (const User*) b
 → ua↑.id - ub↑.id

function compareUserByIdDesc(a : pointer to User, b : pointer to
User) → integer
{melakukan perbandingan untuk mendukung fungsi qsort secara
descending}

KAMUS LOKAL

ALGORITMA
const User ua ← (const User*) a
const User ub ← (const User*) b
 → ub↑.id - ua↑.id

function compareUserByNameAsc(a : pointer to User, b : pointer to
User) → integer
{melakukan perbandingan untuk mendukung fungsi qsort secara
ascending}

KAMUS LOKAL

ALGORITMA
const User ua ← (const User*) a
const User ub ← (const User*) b
 → compare ua↑.username to ub↑.username

function compareUserByNameDesc(a : pointer to User, b : pointer to
User) → integer
{melakukan perbandingan untuk mendukung fungsi qsort secara
descending}

KAMUS LOKAL

ALGORITMA
const User ua ← (const User*) a
const User ub ← (const User*) b
```

```
 → compare ua↑.username to ub↑.username

procedure sortUser(input U : pointer to userList, input sortBy :
integer, input sortOrder : integer)
{melakukan pengurutan berdasarkan sortBy dan sortOrder yang
dipilih}

KAMUS LOKAL

ALGORITMA
if (sortBy = 1) then
    if (sortOrder = 1) then
         qsort(userList↑.elements, userList↑.nEff, sizeof(User),
compareUserByIdAsc)
    else
         qsort(userList↑.elements, userList↑.nEff, sizeof(User),
compareUserByIdDesc)
else
    if (sortOrder = 1) then
         qsort(userList↑.elements, userList↑.nEff, sizeof(User),
compareUserByNameAsc)
    else
         qsort(userList↑.elements, userList↑.nEff, sizeof(User),
compareUserByNameDesc)

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

KAMUS LOKAL
sortBy        : integer
sortOrder     : integer
tempList      : UserList
i, j          : integer
roleStr       : string
disease       : string

ALGORITMA
if (not session.isLoggedIn or session.role ≠ MANAGER) then
    output("Akses ditolak! Hanya Manajer yang dapat melihat daftar
pengguna.")
     →
```

```
sortBy ← printSortBy()
sortOrder ← printSortOrder()

{ Inisialisasi tempList }
tempList.capacity ← hospital.users.capacity
tempList.nEff ← 0
alokasi (tempList.elements sebanyak tempList.capacity)

i traversal [0..hospital↑.users.nEff - 1]
    if (viewType = 1 or
        (viewType = 2 and hospital.users.elements[i].role =
PATIENT) or
        (viewType = 3 and hospital.users.elements[i].role =
DOCTOR)) then
        tempList.elements[tempList.nEff] ←
hospital.users.elements[i]

sortUser(tempList, sortBy, sortOrder)
output("Menampilkan")
depend on(viewType):
1 :
    output(" semua pengguna")
2 :
    output(" pasien")
3 :
    output(" dokter")

output(" dengan ")
if (sortBy = 1) then
output("ID")
else
output("nama")

output(" terurut ")
if (sortOrder = 1) then
output("ascending") else
output("descending")

depend on(viewType):
1 :
    output("ID | Nama     | Role     | Penyakit")
    output("-----------------------------------")
    I traversal [0..tempList.nEff - 1]
        depend on (tempList.elements[i].role)
```

```
        MANAGER:
            roleStr ← "Manager"
        DOCTOR:
            roleStr ← "Dokter"
        PASIEN:
            roleStr ← "Pasien"

        disease ← "-"
        if (tempList.elements[i].role = PATIENT) then
            j traversal [0..hospital.patients.nEff - 1]
                if (hospital.patients.elements[j].id =
tempList.elements[i].id) then
                    disease ← hospital.patients.elements[j].disease
                    break

        output(tempList.elements[i].id, " | ",
tempList.elements[i].username, " | ", roleStr, " | ", disease)

2 :
    output("ID | Nama     | Penyakit")
    output("-------------------------")
    i traversal [0..tempList.nEff - 1]
        disease ← "-"
        j traversal [0..hospital.patients.nEff - 1]
            if (hospital.patients.elements[j].id =
tempList.elements[i].id) then
                disease ← hospital.patients.elements[j].disease
                break
        output(tempList.elements[i].id, " | ",
tempList.elements[i].username, " | ", disease)

3 :
    output("ID | Nama")
    output("--------------")
    i traversal [0..tempList.nEff - 1]
        output(tempList.elements[i].id, " | ",
tempList.elements[i].username)

dealokasi (tempList.elements)
```

F08 - Cari User

```
function sequentialSearchByName(list : UserList, name : char)
```

```
KAMUS
i        : integer
index    : integer

ALGORITMA
index ← -1

i traversal [0..list.nEff-1] do
    if (list.elements[i].username = name) then
        index ← i
        →

function binarySearchById(list : UserList, id : integer, index :
integer)

KAMUS
left, right, mid : integer

ALGORITMA
left ← 0
right ← list.nEff - 1
index ← -1

while (left ≤ right) do
    mid ← (left + right) div 2
    depend on (list.elements[mid].id):
    list.elements[mid].id = id :
        index ← mid
        →
    list.elements[mid].id < id :
        left ← mid + 1
    list.elements[mid].id > id :
        right ← mid - 1

function printSearchResultById(hospital : Hospital, id : integer,
searchType : integer)

KAMUS
index            : integer
user             : User
disease, roleStr : char

ALGORITMA
index ← -1
binarySearchById(hospital.users, id, index)

if (index = -1) then
    output("User dengan ID ", id, " tidak ditemukan.")
    →
```

```
user ← hospital.users.elements[index]

if ((searchType = 2 and user.role ≠ PATIENT) or (searchType = 3 and
user.role ≠ DOCTOR)) then
    depend on (searchType and user.role)
        2 : output("Tidak ditemukan pasien dengan ID ", id, ".")
        3 : output("Tidak ditemukan dokter dengan ID ", id, ".")
    →

disease ← "-"
if (user.role = PATIENT) then
    i traversal [0..hospital.patients.nEff-1]
        if (hospital.patients.elements[i].id = id) then
            disease ← hospital.patients.elements[i].disease
            stop

depend on (user.role)
    MANAGER : roleStr ← "Manager"
    DOCTOR  : roleStr ← "Dokter"
    PATIENT : roleStr ← "Pasien"

depend on (searchType)
    1 :
        output("Menampilkan pengguna dengan ID ", id, "...")
        output("ID | Nama     | Role     | Penyakit")
        output("-------------------------------------")
        output(user.id, " | ", user.username, " | ", roleStr, " |
", disease)
    2 :
        output("Menampilkan pasien dengan ID ", id, "...")
        output("ID | Nama     | Penyakit")
        output("-------------------------")
        output(user.id, " | ", user.username, " | ", disease)
    3 :
        output("Menampilkan dokter dengan ID ", id, "...")
        output("ID | Nama")
        output("--------------")
        output(user.id, " | ", user.username)

function printSearchResultByName(hospital : Hospital, name : char,
searchType : integer)

KAMUS
index            : integer
user             : User
disease, roleStr : char

ALGORITMA
index ← -1
sequentialSearchByName(hospital.users, name, index)
```

```
if (index = -1) then
    output("Tidak ditemukan pengguna dengan nama ", name, "!")
    →

user ← hospital.users.elements[index]

if ((searchType = 2 and user.role ≠ PATIENT) or (searchType = 3 and
user.role ≠ DOCTOR)) then
    depend on (searchType)
        2 : output("Tidak ditemukan pasien dengan nama ", name,
".")
        3 : output("Tidak ditemukan dokter dengan nama ", name,
".")
    →

disease ← "-"
if (user.role = PATIENT) then
    i traversal [0..hospital.patients.nEff-1]
        if (hospital.patients.elements[i].id = user.id) then
            disease ← hospital.patients.elements[i].disease
            stop

depend on (user.role)
    MANAGER : roleStr ← "Manager"
    DOCTOR  : roleStr ← "Dokter"
    PATIENT : roleStr ← "Pasien"

depend on (searchType)
    1 :
        output("Menampilkan pengguna dengan nama ", name, "...")
        output("ID | Nama     | Role     | Penyakit")
        output("-------------------------------------")
        output(user.id, " | ", user.username, " | ", roleStr, " |
", disease)
    2 :
        output("Menampilkan pasien dengan nama ", name, "...")
        output("ID | Nama     | Penyakit")
        output("-------------------------")
        output(user.id, " | ", user.username, " | ", disease)
    3 :
        output("Menampilkan dokter dengan nama ", name, "...")
        output("ID | Nama")
        output("--------------")
        output(user.id, " | ", user.username)

function printSearchResultByDisease(hospital : Hospital, disease :
char, sortBy : integer, sortOrder : integer)

KAMUS
tempList : UserList
```

```
userId   : integer
i, j     : integer

ALGORITMA
tempList.capacity ← hospital.users.capacity
tempList.nEff ← 0
alokasi(tempList.elements)

i traversal [0..hospital.patients.nEff-1]
    if (hospital.patients.elements[i].disease = disease) then
        userId ← hospital.patients.elements[i].id
        j traversal [0..hospital.users.nEff-1]
            if (hospital.users.elements[j].id = userId) then
                tempList.elements[tempList.nEff] ←
hospital.users.elements[j]
                tempList.nEff ← tempList.nEff + 1
                stop

if (tempList.nEff = 0) then
    output("Tidak ditemukan pasien dengan penyakit ", disease, "!")
    hapus tempList
    →

sortUser(tempList, sortBy, sortOrder)

output("Menampilkan pasien dengan penyakit ", disease, " dengan ")
depend on (sortBy)
    1 : output("ID")
    2 : output("nama")
output(" terurut ")
depend on (sortOrder)
    1 : output("ascending...")
    2 : output("descending...")

output("ID | Nama     | Penyakit")
output("-------------------------")
i traversal [0..tempList.nEff-1]
    output(tempList.elements[i].id, " | ",
tempList.elements[i].username, " | ", disease)

dealokasi(tempList.elements)

function searchUser(hospital : Hospital, session : Session,
searchType : integer)

KAMUS
choice, id, sortBy, sortOrder : integer
name, disease                 : char

ALGORITMA
if (hospital = NULL or session = NULL) then
```

```
    output("Struktur rumah sakit atau sesi tidak valid!")
    →

if (!session->isLoggedIn or session->role ≠ MANAGER) then
    output("Akses ditolak! Hanya Manajer yang dapat mencari
pengguna.")
    →

output("Cari berdasarkan?")
output("1. ID")
output("2. Nama")
if (searchType = 2) then
    output("3. Penyakit")

output(">>> Pilihan: ")
input(choice)

if (searchType = 2) then
    validateInputInteger(choice, 1, 3)
else
    validateInputInteger(choice, 1, 2)

output("")

depend on (choice)
    1 :
        depend on (searchType)
            1 : output(">>> Masukkan nomor ID user: ")
            2 : output(">>> Masukkan nomor ID pasien: ")
            3 : output(">>> Masukkan nomor ID dokter: ")
        input(id)
        output("")
        printSearchResultById(hospital, id, searchType)
    2 :
        depend on (searchType)
            1 : output(">>> Masukkan nama user: ")
            2 : output(">>> Masukkan nama pasien: ")
            3 : output(">>> Masukkan nama dokter: ")
        input(name)
        output("")
        printSearchResultByName(hospital, name, searchType)
    3 :
        output(">>> Masukkan penyakit: ")
        input(disease)
        output("")
        sortBy ← printSortBy()
        sortOrder ← printSortOrder()
        printSearchResultByDisease(hospital, disease, sortBy,
sortOrder)
```

F10 - Tambah Dokter

```
function isUsernameTaken(input Hospital hospital, username : char)
→ boolean
{mengembalikan true apabila username yang sudah digunakan pengguna
lain, dan false bila kondisi tidak terjadi}

KAMUS
lowerUsername        : array of [1..100] of char
lowerElementUsername : array of [1..100] of char
c                    : char
i                    : integer
j                    : integer

ALGORITMA
// inisialisasi variabel
lowerUsername ← length(username) of char

while(i + 1 < length(username)do
    c ← username[i]
    if (c >= 'A' and c <= 'Z') then
        c = c + ('a' - 'A')
    lowerUsername[i] ← c
lowerUsername[length(username)] ← '\0'

while(i + 1 < hospital->users.nEff)do

lowerElementUsername[length(hospital->users.elements[i].username +
1]] of char
    while (j + 1 < length(hospital->user.elements[i].username) do
        c ← hospital->users.elements[i].username[j]
        if (c >= 'A' and c <= 'Z') then
            c = c + ('a' - 'A')
        lowerElementUsername[j] ← c

lowerElementUsername[length(hospital->users.elements[i].username ←
'\0'

    if (lowerUsername = lowerElementUsername) then
        return ← true

return ← false {kondisi jika tidak memenuhi}
```

```
function addDoctor(input/output Hospital hospital, Session session,
username : char, password : char, specialization : char) → boolean
{mengembalikan true apabila berhasil menambahkan dokter}

KAMUS
newUser   : char
newDoctor : char

ALGORITMA
if (hospital = NULL or session = NULL or username = NULL or
password = NULL or specialization = NULL) then
    output("Struktur rumah sakit, sesi, atau input tidak valid!")
    return ← false

if (!session->isLoggedIn or session->role != MANAGER) then
    output("Akses ditolak! Hanya Manajer yang dapat menambah
dokter.")
    return ← false

if (!isValidUsername(username)) then
    output("Username tidak valid! Gunakan huruf, angka, atau
underscore.")
    return ← false

if (!isValidUsername(username)) then
    output("Username tidak valid! Gunakan huruf, angka, atau
underscore.")
    return ← false

if (length(password) < 6) then
    output("Password harus minimal 6 karakter!")
    return ← false

if (isValidUsername(specialization)) then
    output("Spesialisasi tidak valid! Gunakan huruf, angka, atau
underscore.")
    return ← false

if (isUsernameTaken(hospital, username)) then
    output("Username sudah terdaftar!")
    return ← false

if (hospital->users.nEff >= hospital->users.capacity or
hospital->doctors.nEff >= hospital->doctors.capacity) then
```

```
    output("Kapasitas pengguna atau dokter penuh!")
    return ← false

// Menambahkan user baru ke UserList
newUser ← hospital->users.elements[hospital->users.nEff]
newUser->id ← hospital->users.nEff + 1
newUser->username ← username

if (!enigmaEncrypt(password,
newUser->password.encryptedContent,100))
    output("Gagal mengenkripsi password!")
    return ← false

newUser->role ← DOCTOR
// Menambahkan ke DoctorList
newDoctor ← newUser->id
newDoctor->username ← username
newDoctor->specialization ← specialization
newDoctor->aura ← 0
newDoctor->bananaRich ← 100.0f
newDoctor->room[0] ← '\0'

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

// Membuat pesan sukses
output("Dokter <username> berhasil ditambahkan!")
return ← true

function assignDoctor(input/output Hospital hospital, Session
session, username : char, roomCode : char, specialization : char) →
boolean
{mengembalikan true apabila berhasil mengalokasikan dokter ke room}

KAMUS
doctorIdx : integer

ALGORITMA
if (hospital = NULL or session = NULL or username = NULL or
roomCode = NULL) then
    output("Struktur rumah sakit, sesi, atau input tidak valid!")
    return ← false

if (!session->isLoggedIn or session->role != MANAGER) then
```

```
    output("Akses ditolak! Hanya Manajer yang dapat menugaskan
dokter.")
    return ← false

if (!isValidUsername(username)) then
    output("Username dokter tidak valid!")
    return ← false

if (!isValidRoomCode(hospital, roomCode)) then
    output("Kode ruangan tidak valid! Contoh: A1")
    return ← false

doctorIdx ← -1
while (i + 1 < hospital->doctors.nEff) do
    if (hospital->doctors.elements[i].username == username) then
        doctorIdx ← i
        stop
if (doctorIdx = -1) then
    output{"Dokter tidak ditemukan!")
    return ← false

room ← NULL
if (hospital->layout.rowEff > 0 and hospital->layout.colEff > 0)
then
    while (i + 1 < hospital->layuout.rowEff) do
        while (j + 1 < hospital->layout.colEff) do
           if (hospital->layout.elements[i][j].code = room code)
then
               room ← hospital->layout.elements[i][j]
               stop

if (room = NULL) then
    output("Ruangan tidak ditemukan!)"
    return ← false
if (room->doctorID != -1) then
    output("Ruangan sudan ditempati dokter lain!")
    return ← false

doctor ← hospital->doctors.elements[doctorIdx]
if (doctor->room[0] != '\0') then
    while (i + 1 < hospital->layuout.rowEff) do
        while (j + 1 < hospital->layout.colEff) do
           if (hospital->layout.elements[i][j].code = doctor->room)
then
```

```
                hospital->layout.elements[i][j].doctorID ← -1
                Stop

room->doctorID ← doctor->id
doctor->room ← roomCode

// Membuat pesan sukses
output("Dokter <username> ditugaskan ke ruangan <roomCode>")
return ← true
```

F18 - Exit

```
procedure exitProgram
{
   Menampilkan pesan keluar dan mengeluarkan pengguna dari program
}

KAMUS LOKAL
-

ALGORITMA
output("Anda telah keluar dari program, babai")
exit(0)
```

PROGRAM UTAMA

Main: **hospitalSystem.c**

```
// sebelum fungsi main, berikut adalah beberapa fungsi ataupun
prosedur yang digunakan untuk memudahkan pembacaan program dan
membatasi input.

procedure normalizeCommand(input/output command : string)
{ mengubah input command menjadi huruf kapital dan menghapus
spasi/underscore }

KAMUS LOKAL
i, j : integer

ALGORITMA
i ← 0
j ← 0
```

```
while (command[i] ≠ '\0')
    if (command[i] ≠ ' ' and command[i] ≠ '_')
        if (command[i] ∈ ['a'..'z'])
            command[j] ← toUpper(command[i])
        else
            command[j] ← command[i]
        j ← j + 1
    i ← i + 1

command[j] ← '\0'

function stringToInt(input str : string) → integer
{ mengubah string angka menjadi bilangan bulat (integer),
memperhitungkan tanda negatif }

KAMUS LOKAL
result : integer
i      : integer
sign   : integer

ALGORITMA
result ← 0
i ← 0
sign ← 1

if (str[0] = '-')
    sign ← -1
    i ← 1

while (str[i] ≠ '\0')
    if (str[i] ∈ ['0'..'9'])
        result ← result * 10 + (str[i] - '0')
    else
        return sign * result
    i ← i + 1

return sign * result

function readValidString(output str : string, input maxLength :
integer, input prompt : string, input allowSpaces : boolean) →
boolean
{ membaca string dari input, menampilkan prompt jika ada, dan
membatasi panjang input }

KAMUS LOKAL
str          : string
maxLength    : int
prompt       : string
allowsSpaces : boolean
inputStatus  : integer
```

```
valid        : boolean

ALGORITMA
if (prompt ≠ NULL)
    output(prompt)

if (scanf(str) ≠ sukses)
    return false

if (length(str) ≥ maxLength - 1)
    str[maxLength - 1] ← '\0'

return true
```

```
procedure main()
{ main file dari Nimons Hospital Management System, gabungan dari
ke-8 fungsi pertama, serta tambah dokter dan exit }

KAMUS MAIN
hospital         : Hospital
session          : Session
hospitalRows     : integer
hospitalCols     : integer
command          : string[50]
argc             : integer
argv             : array of string

ALGORITMA
{ inisialisasi denah rumah sakit }
if (argc >= 2 and argc <= 3)
    hospitalRows ← stringToInt(argv[1])
    if (argc = 3)
        hospitalCols ← stringToInt(argv[2])
    else
        hospitalCols ← hospitalRows
    if (hospitalRows <= 0 or hospitalCols <= 0)
        output("Ukuran denah tidak valid!")
        return
else
    hospitalRows ← 5
    hospitalCols ← 5

{ default layout }
initHospital(hospital, 100, 100, 100, hospitalRows, hospitalCols)

{ inisialisasi akun manajer }
hospital.users.elements[0].id ← 1
hospital.users.elements[0].username ← "nimonsslatte"
hospital.users.elements[0].password ← "nimonatutgajah23"
hospital.users.elements[0].role ← MANAGER
hospital.users.nEff ← hospital.users.nEff + 1
```

```
{ inisialisasi session }
session.isLoggedIn ← false
session.userID ← -1
session.username ← ""
session.role ← -1

// menu awal, berserta command input

output("========== Nimons Hospital Management System ==========")
output("Ketik 'HELP' untuk melihat daftar perintah")

repeat forever
    output(">>> ")
    readValid ← readValidString(command, 50, NULL, true)
    if (not readValid)
        output("Perintah tidak valid!")
        continue

    normalizeCommand(command)

    // pemanggilan fungsi untuk seluruh menu
    if (command = "LOGIN")
        login(hospital, session)
    else if(command = "REGISTER")
        registerPatient(hospital, session)
    else if (command = "LOGOUT")
        logout(session)
    else if (command = "LUPA_PASSWORD" or command = "LUPAPASSWORD")
        forgotPassword(hospital.users)
    else if (command = "HELP")
        displayMenu(session)
    else if (command = "LIHAT_DENAH" or command = "LIHATDENAH")
        displayLayout(hospital, session)
    else if (command = "LIHAT_RUANGAN" or command = "LIHATRUANGAN")
        displayRoomDetails(hospital, session)
    else if (command = "LIHAT_USER" or command = "LIHATUSER")
        displayUsers(hospital, session, 1)
    else if (command = "LIHAT_PASIEN" or command = "LIHATPASIEN")
        displayUsers(hospital, session, 2)
    else if (command = "LIHAT_DOKTER" or command = "LIHATDOKTER")
        displayUsers(hospital, session, 3)
    else if (command = "CARI_USER" or command = "CARIUSER")
        searchUser(hospital, session, 1)
    else if (command = "CARI_PASIEN" or command = "CARIPASIEN")
        searchUser(hospital, session, 2)
    else if (command = "CARI_DOKTER" or command = "CARIDOKTER")
        searchUser(hospital, session, 3)
    else if (command = "TAMBAH_DOKTER" or command = "TAMBAHDOKTER")
        addDoctor(hospital, session)
    else if (command = "ASSIGN_DOKTER" or command = "ASSIGNDOKTER")
```

```
        assignDoctor(hospital, session)
    else if (command = "EXIT")
        exitProgram()
        break
    else
        output("Perintah tidak dikenali! Ketik 'HELP' untuk
bantuan.")

    output("\n\n")

{ cleanup }
freeHospital(hospital)
```

TANGKAPAN LAYAR

F01 - Login

Kasus 1 : Login sebagai Manager

```
>>> LOGIN
Username: nimonsslatte
Password: nimonatutgajah23
Selamat pagi Manager nimonsslatte!
```

Kasus 2 : Login sebagai Dokter

```
>>> LOGIN
Username: abelgantenghehe
Password: tubesalproinisangatmenyenangkan
Selamat pagi Dokter abelgantenghehe!
```

Kasus 3 : Login sebagai Pasien

```
>>> LOGIN
Username: almerresing
Password: almerresingsambiltubesalpro
Selamat pagi almerresing! Ada keluhan apa?
```

Kasus 4 : Tidak ada username yang terdaftar

```
>>> LOGIN
Username: nisasabyan
Password: nisasabyan712
Tidak ada Manager, Dokter, atau pun Pasien yang bernama nisasabyan!
```

Kasus 5 : Kasus password salah

```
>>> LOGIN
Username: almerresing
Password: almergajadiresing
Username atau password salah untuk pengguna yang bernama almerresing!
```

F02 - Register Pasien

Kasus 1 : Pasien bernama **almerresing** belum ada

```
>>> REGISTER
Username: almerresing
Password: almerresingsambiltubesalpro
Pasien almerresing berhasil ditambahkan!
```

Kasus 2 : Pasien bernama **almerresing** sudah ada

```
>>> REGISTER
Username: almerresing
Password: lhokokresing
Registrasi gagal! Pasien dengan nama almerresing sudah terdaftar.
```

F03 - Logout

Kasus 1 : sedang dalam keadaan logged in

```
>>> LOGOUT
Keluar dari akun Manager nimonsslatte
Sampai jumpa!
```

Kasus 2 : sedang dalam keadaan belum logged in

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

F04 - Lupa Password

Kasus 1 : Manager, Dokter, atau Pasien bernama **nimonsslatte** ada

```
>>> LUPA_PASSWORD
Username: nimonsslatte
Kode Unik: nimons2lat2e
Halo Manager nimonsslatte, silakan daftarkan ulang password anda!
Password Baru: manusiamanusiakwat
Password berhasil diubah!
```

Kasus 2 : Manager, Dokter, atau Pasien bernama **akusukadia** tidak ada

```
>>> LUPA_PASSWORD
Username: akusukadia
Username tidak terdaftar!
```

Kasus 3 : Kode unik untuk username **zulfanurhuda** bukan **z2alsha** tetapi adalah **zulfanurhuda**

```
>>> LUPA_PASSWORD
Username: zulfanurhuda
Kode Unik: z2alsha
Kode unik salah!
```

F05 - Menu & Help

Kasus 1 : belum dalam keadaan logged in

```
>>> HELP
=========== HELP ===========

Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

LOGIN: Masuk ke dalam akun yang sudah terdaftar
REGISTER: Membuat akun baru

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

Kasus 2 : sudah login sebagai Dokter

```
>>> HELP
=========== HELP ===========

Halo Dokter abelgantenghehe. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang
dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan
DIAGNOSIS: Melakukan diagnosis penyakit pasien berdasarkan kondisi tubuh pasien

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

Kasus 3 : sudah login sebagai Pasien

```
>>> HELP
=========== HELP ===========

Selamat datang, almerresing. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang
dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan
DAFTAR_CHECKUP: Mendaftarkan diri untuk pemeriksaan dokter

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

Kasus 4 : sudah login sebagai Manager

```
>>> HELP
========== HELP ==========

Halo Manager nimonsslatte. Kenapa kamu memanggil command HELP? Kan kamu manager, tapi yasudahlah kamu pasti sedang ke
bingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan
TAMBAH_DOKTER: Mendaftarkan dokter baru ke sistem
ASSIGN_DOKTER: Menugaskan dokter ke ruangan tertentu
LIHAT_DENAH: Melihat denah rumah sakit
LIHAT_RUANGAN: Melihat detail ruangan tertentu
LIHAT_USER: Melihat daftar seluruh pengguna
LIHAT_PASIEN: Melihat daftar pasien
LIHAT_DOKTER: Melihat daftar dokter
CARI_USER: Mencari pengguna berdasarkan kriteria
CARI_PASIEN: Mencari pasien berdasarkan kriteria
CARI_DOKTER: Mencari dokter berdasarkan kriteria

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

F06 - Denah Rumah Sakit

Kasus 1 : ruangan terdapat dokter dan pasien

Test case ini tidak dapat ditunjukkan karena menu untuk **register checkup** belum ada, tidak ada pasien yang bisa masuk ke ruangan.

Kasus 2 : ruangan terdapat dokter dan tidak ada pasien

```
>>> LIHAT_RUANGAN B3
Masukkan kode ruangan:
--- Detail Ruangan B3 ---
Kapasitas  : 3
Dokter     : Dr. abelgantenghehe
Pasien di dalam ruangan:
  Tidak ada pasien di dalam ruangan saat ini.
-----------------------------
```

Kasus 3 : ruangan tidak terdapat dokter

```
>>> LIHAT_RUANGAN E4
Masukkan kode ruangan:
--- Detail Ruangan E4 ---
Kapasitas  : 3
Dokter     : -
Pasien di dalam ruangan:
  Tidak ada pasien di dalam ruangan saat ini.
------------------------------
```

F07 - Lihat User

Kasus 1 : Melihat data user (dokter/pasien)

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

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

Menampilkan semua pengguna dengan ID terurut ascending...

ID | Nama     | Role     | Penyakit
-------------------------------------
1  | nimonsslatte | Manager  | -
2  | zulfanurhuda | Dokter   | -
3  | abelgantenghehe | Dokter   | -
```

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

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

Menampilkan semua pengguna dengan ID terurut descending...

ID | Nama     | Role     | Penyakit
-------------------------------------
3  | abelgantenghehe | Dokter   | -
2  | zulfanurhuda | Dokter   | -
1  | nimonsslatte | Manager  | -
```

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

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

Menampilkan semua pengguna dengan nama terurut ascending...

ID | Nama     | Role     | Penyakit
-------------------------------------
3  | abelgantenghehe | Dokter   | -
1  | nimonsslatte | Manager  | -
2  | zulfanurhuda | Dokter   | -
```

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

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

Menampilkan semua pengguna dengan nama terurut descending...

ID | Nama     | Role     | Penyakit
-------------------------------------
2  | zulfanurhuda | Dokter   | -
1  | nimonsslatte | Manager  | -
3  | abelgantenghehe | Dokter   | -
```

Kasus 2 : Spesifik melihat data pasien

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

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

Menampilkan pasien dengan nama terurut descending...

ID | Nama     | Penyakit
-------------------------
5  | annnissahduar |
4  | almerresing |
```

Kasus 3 : Spesifik melihat data dokter

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

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

Menampilkan dokter dengan ID terurut descending...

ID | Nama
-------------
3  | abelgantenghehe
2  | zulfanurhuda
```

F08 - Cari User

Kasus 1 : Mencari data user (dokter/pasien) berdasarkan ID

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

>>> Masukkan nomor ID user: 3

Menampilkan pengguna dengan ID 3...

ID | Nama     | Role     | Penyakit
-------------------------------------
3  | abelgantenghehe | Dokter   | -
```

Kasus 2 : User yang dicari tidak ditemukan

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

>>> Masukkan nama user: almerpromex

Tidak ditemukan pengguna dengan nama almerpromex!
```

Kasus 3 : Mencari data pasien

```
>>> Masukkan nama pasien: annnissahduar

Menampilkan pasien dengan nama annnissahduar...

ID | Nama      | Penyakit
--------------------------
5  | annnissahduar |
```

Kasus 4 : Mencari data dokter

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

>>> Masukkan nama dokter: abelgantenghehe

Menampilkan dokter dengan nama abelgantenghehe...

ID | Nama
-------------
3  | abelgantenghehe
```

F10 - Tambah Dokter

Kasus 1 : Dokter bernama **zulfanurhuda** belum ada

```
>>> TAMBAH_DOKTER
Username: zulfanurhuda
Password: akusukatubesalprosangatseru
Spesialisasi: Bedah
Dokter zulfanurhuda berhasil ditambahkan!
```

Kasus 2 : Dokter bernama **zulfanurhuda** sudah ada

```
>>> TAMBAH_DOKTER
Username: zulfanurhuda
Password: wahkerensekalitubesalpro
Sudah ada Dokter bernama zulfanurhuda!
```

Kasus 1 : Ruangan kosong dan dokter belum di assign di ruang manapun

```
>>> ASSIGN_DOKTER
Username: zulfanurhuda
Ruangan: A1
Dokter zulfanurhuda berhasil diassign ke ruangan A1!
```

Kasus 2 : Ruangan kosong dan dokter sudah di assign di ruangan lain

```
>>> ASSIGN_DOKTER
Username: zulfanurhuda
Ruangan: B2
Dokter zulfanurhuda sudah diassign ke ruangan A1!
```

Kasus 3 : Ruangan tidak kosong dan dokter belum di assign di ruangan manapun

```
>>> ASSIGN_DOKTER
Username: abelgantenghehe
Ruangan: A1
Dokter zulfanurhuda sudah menempati ruangan A1!
Silakan cari ruangan lain untuk dokter abelgantenghehe.
```

Kasus 4 : Ruangan tidak kosong dan dokter sudah di assign di ruangan lain

```
>>> ASSIGN_DOKTER
Username: abelgantenghehe
Ruangan: A2
Dokter abelgantenghehe sudah diassign ke ruangan B3!
Ruangan A2 juga sudah diassign ke dokter azzainanotherlife!
```

F18 - Exit

Kasus 1 : Keluar program

```
>>> EXIT
Anda telah keluar dari program, babai!
SUCCESS: Memori rumah sakit berhasil dibebaskan!
```

Kasus 2 : Input tidak valid

```
>>> INICOMMAND
Perintah tidak dikenali! Ketik 'HELP' untuk bantuan.
```

LAMPIRAN

Formulir Asistensi 1: IF1210_FormAsistensiTB_1_K04-I.docx