LAPORAN FINAL PROJECT
PRAKTIKUM STRUKTUR DATA

Disusun oleh :

Dosen Pengampu :

I Gusti Anom Cahyadi Putra, S.T., M.Cs.

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

2020

DAFTAR ISI

BAB I
LANDASAN TEORI

1.1. Queue

Queue merupakan sebuah struktur data yang menggunakan konsep FIFO (first in first out) dimana data yang dimasukkan pertama adalah data yang pertama keluar dari queue. Data yang ditambahkan dalam queue akan disimpan dalam sebuah tempat dengan nama atau posisi REAR yaitu pada akhir dari urutan data sedangkan data yang akan di keluarkan berada dalam tempat dengan nama atau posisi FRONT pada awal dari urutan data. Struktur data Queue dapat di implementasikan dengan menggunakan array ataupun linked list.

Operasi yang ada dalam implementasi Queue. Enqueue, dequeue, peek, dan display.

- Enqueue: menginputkan data kedalam queue pada posisi terakhir atau REAR.
- Dequeue: mengeluarkan data dari queue pada posisi terdepan atau FRONT.
- Peek: melihat data pertama dan terakhir pada queue.
- Display: menampilkan seluruh data yang ada pada queue.

1.2. Priority Queue

Priority queue merupakan sebuah struktur data yang didasarkan dari struktur data queue. Priority queue memiliki basis HPIFO (*Highest Priority in First Out*), berbeda dengan queue yang memiliki basis FIFO (*First in First Out*). Dalam priority queue, elemen yang memiliki prioritas tertinggi lah yang akan diambil atau dihapus.

Priority queue digunakan pada saat sebuah data dalam antrian perlu diprioritaskan. Contohnya pada saat kita berada di amusement park, pengunjung yang memiliki kartu vip akan dilayani terlebih dahulu. Contoh lainnya adalah pada antrian di rumah sakit, tidak mungkin pasien yang parah seperti kecelakaan

harus menunggu sampai pasien- pasien sebelumnya selesai untuk mendapat perawatan, tentunya pasien ini harus diprioritaskan. Untuk itu, kita dapat menggunakan priority queue.

Ada pun tipe-tipe dari priority queue, yaitu sebagai berikut:

- Ascending Priority Queue merupakan priority queue yang elemen-elemennya diurutkan dengan prioritas menaik, yakni dimulai dari elemen dengan prioritas terendah hingga prioritas tertinggi.
- Descending Priority Queue merupakan priority queue yang elemen-elemennya diurutkan dengan prioritas menurun, yakni dimulai dari elemen dengan prioritas tertinggi hingga prioritas terendah.

Priority queue bisa diimplementasikan dengan tree menggunakan binary heap dan juga dapat diimplementasikan menggunakan array. Dalam program ini, kami mengimplementasikannya menggunakan array, dengan cara menambah data pada akhir, lalu menempatkannya sesuai prioritas dan menghapus data pada awal yaitu data dengan prioritas tertinggi yang lebih awal ditambahkan.

1.3. Rekursif

Fungsi rekursif merupakan sebuah fungsi atau method yang memanggil dirinya sendiri baik secara langsung ataupun secara tak langsung (lewat fungsi lain). Proses pemanggilan inilah yang disebut rekursif.

Ada dua aspek yang harus dimiliki oleh fungsi rekursi. Sebuah fungsi rekursi harus mengetahui kapan harus berhenti (simple case) dan kapan harus memanggil dirinya kembali. Format penulisan isi dalam bahasa C umumnya menggunakan (if-else), dengan implementasi sebagai berikut :

if kasus sederhana then

solusi

else

mendefinisikan masalah kembali dengan recursion

Rekursi bisa digunakan sebagai pengganti perulangan, kita bisa saja menggunakan perulangan untuk menciptakan solusi yang sama, kelebihan

rekursif yaitu sangat mudah untuk melakukan perulangan dengan batasan yang luas dalam artian melakukan perulangan dalam skala yang besar, dan dapat melakukan perulangan dengan batasan fungsi.

BAB II
HASIL PROGRAM

2.1. Source Code

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

//nomor antrian dan size array
int no=0, size=-1;

//struct antrian pasien
struct pasien{
	int nomor_antrian;
	char nama[20];
	char alamat[50];
	int umur;
	char jk;
	int prioritas;
};

struct pasien *temp;
struct pasien *Table[10];

//struct riwayat pasien
struct riwayat{
	int nomor_antrian;
	char nama[20];
	char alamat[50];
	int umur;
	char jk;
	int prioritas;
```

```
        int total_tagihan;
        struct riwayat *next;
};

struct riwayat *tempR;
struct riwayat *front=NULL, *rear=NULL;

//menukar node input ke posisi sesuai melalui penukaran dengan node sebelumnya
void tukar(int i){
        int x=i;
        //apabila nilai prioritas node sebelumnya lebih kecil daripada proiritas node
        if(i>0 && Table[i-1]->prioritas < Table[i]->prioritas){
                temp=Table[i-1];
                Table[i-1]=Table[i];
                Table[i]=temp;

                //proses rekursi
                tukar(--i);
        }

}

//fungsi untuk menambah antrian
void tambah_antrian(){
        //indeks dalam array table
        size++;
        if(size==10){
                size--;
                printf("\nAntrian penuh!\n");
                return;
        }
```

```
int indeks=size,i=0;
char jk;

//alokasi memori untuk struct pasien
Table[indeks]= (struct pasien*)malloc(sizeof(struct pasien));

//masukkan data
NAMA:
printf("Masukkan nama pasien\t: ");
fflush(stdin);
gets(Table[indeks]->nama);
//cek input nama yang sama
while(i<size && strcmp(strlwr(Table[indeks]->nama), strlwr(Table[i]->nama))!=0){
	i++;
}

if(i<size){
	printf("Nama Sudah ada, Silahkan masukkan nama berbeda!\n\n");
	getch();
	goto NAMA;
}

printf("Masukkan alamat pasien\t: ");
fflush(stdin);
gets(Table[indeks]->alamat);
printf("Masukkan Umur Pasien\t: ");
scanf("%d", &Table[indeks]->umur);
printf("Masukkan jenis kelamin pasien <p/l>\t: ");
fflush(stdin);
scanf("%c", &jk);
Table[indeks]->jk=toupper(jk);
```

```
printf("---------------------------------------------------------------\n");
        printf("1. Ringan \n");
        printf("2. Sedang \n");
        printf("3. Berat \n");
        printf("Masukkan Tingkat Keparahan Pasien\t: ");
        scanf("%d", &Table[indeks]->prioritas);

        //menamilkan nomor antian

printf("---------------------------------------------------------------\n");
        no++;
        Table[indeks]->nomor_antrian=no;
        printf("Nomor Antrian Pasien\t: %d\n", Table[indeks]->nomor_antrian);

printf("---------------------------------------------------------------\n");
        getch();

        //update posisi
        if(indeks>0){
                tukar(indeks);
        }

}



//fungsi untuk menggeser indeksnya
void geser(int i){
        if(i<size){
                //swapping
```

```
                Table[i]=Table[i+1];
                //proses rekursi
                geser(++i);
        }
        else {
                Table[i]=NULL;
                size--;
        }
}

//menghapus satu antrian pasien
void hapus_antrian(){
        //menyimpan antrian yang ingin dihapus ke node riwayat
        //membuat node riwayat
        if(size==-1){
                printf("\nAntrian Kosong!!\n");
                getch();
                return;
        }
        tempR=(struct riwayat *)malloc(sizeof(struct riwayat));
        strcpy(tempR->nama,Table[0]->nama);
        strcpy(tempR->alamat,Table[0]->alamat);
        tempR->umur=Table[0]->umur;
        tempR->jk=Table[0]->jk;
        tempR->prioritas=Table[0]->prioritas;
        tempR->nomor_antrian=Table[0]->nomor_antrian;

        //swap dan hapus node
        if(size==0){
                Table[size]=NULL;
                size--;
        }
        else{
```

```
                geser(0);
        }



        //menampilkan isi node

printf("--------------------------------------------------------------------\n");
        printf("Nama Pasien\t: %s\n", tempR->nama);
        printf("Alamat Pasien\t: %s\n", tempR->alamat);
        printf("Umur Pasien\t: %d\n", tempR->umur);
        printf("Jenis Kelamin Pasien\t: %c\n", tempR->jk);
        printf("Tingkat Prioritas\t: %d\n", tempR->prioritas);
        printf("Nomor Antrian\t: %d\n", tempR->nomor_antrian);

printf("--------------------------------------------------------------------\n");
        printf("Masukkan Total Tagihan\t: Rp. ");
        scanf("%d", &tempR->total_tagihan);

        //menambahkan ke linked list Riwayat
        if(front==NULL){
                front=tempR;
                rear=tempR;
                tempR->next=NULL;
        }
        else{
                rear->next=tempR;
                rear=tempR;
                tempR->next=NULL;
        }
}

//fungsi riwayat antrian
```

```
void riwayat_antrian(){

printf("\n-----------------------------------------------------------------\n");
	//set pointer di front
	tempR=front;

	if(tempR==NULL){
		printf("Riwayat Kosong !!\n");
		getch();
		return;
	}

	//perulangan utnuk menampilkan riwayat antrian
	while(tempR!=NULL){

		printf("Nama Pasien\t: %s\n", tempR->nama);
		printf("Alamat Pasien\t: %s\n", tempR->alamat);
		printf("Umur Pasien\t: %d\n", tempR->umur);
		printf("Jenis Kelamin Pasien\t: %c\n", tempR->jk);
		printf("Tingkat Prioritas\t: %d\n", tempR->prioritas);
		printf("Nomor Antrian\t: %d\n", tempR->nomor_antrian);
		printf("Total Tagihan\t: Rp. %d\n", tempR->total_tagihan);


printf("-----------------------------------------------------------------\n");
		tempR=tempR->next;
		getch();
	}
}

//menu untuk resepsionis
```

```
void menu_resepsionis(){
    //deklarasi variable
    int menu;
    MENU:
    //judul

printf(".==========================================
====================.\n");
    printf("|                Antrian Rumah Sakit                    |\n");

printf("'==========================================
====================\n");
    printf("| 1. Tambah Antrian                                     |\n");
    printf("| 2. Hapus Antrian                                      |\n");
    printf("| 3. Riwayat Pasien                                     |\n");
    printf("| 4. Kembali                                            |\n");

printf("'==========================================
====================\n");
    printf("Masukkan Pilihan Anda : ");
    scanf("%d", &menu);

    //switch case menu
    switch (menu){
        case 1:{


printf("-----------------------------------------------------------------\n");
                tambah_antrian();
                break;
        }
        case 2:{
```

```
printf("--------------------------------------------------------------------\n");
                        hapus_antrian();
                        break;
                }
                case 3:{
                        riwayat_antrian();
                        break;
                }
                case 4:{
                        return;
                        break;
                }
        }
        system("cls");
        goto MENU;
}

//menu untuk pasien
void menu_pasien(){
        int i=0;
        char nama[20];
        //judul

printf(".============================================
==================.\n");
        printf("|                Antrian Rumah Sakit                |\n");

printf("'============================================
==================\n");
        printf("Masukkan Nama Anda : ");
        fflush(stdin);
        gets(nama);
```

```
        //menampilkan urutan antrian sampai pasien yang dicari
        if(size==-1){
                printf("Antrian Kosong!!\n");
                getch();
                return;
        }
        else if(size==0 && strcmp(strlwr(Table[i]->nama),strlwr(nama))
!= 0){
                printf("(%d) ", Table[i]->nomor_antrian);
                printf("\nData Tidak Ditemukan!\n");
                getch();
                return;
        }
        else if(strcmp(strlwr(Table[i]->nama),strlwr(nama)) == 0){
                printf("(%d) ",Table[i]->nomor_antrian);
                getch();
        }
        else{
                while(i<=size && strcmp(strlwr(Table[i]-
>nama),strlwr(nama)) != 0){
                        if(i==0){
                                printf("(%d) ", Table[i]-
>nomor_antrian);
                        }
                        else{
                                printf("<- (%d) ", Table[i]-
>nomor_antrian);
                        }
                        getch();
                        i++;
                }
                //menampilkan data dari indeks pasien yang dicari
```

```
                if(i>size){
                        printf("\nData Tidak Ditemukan!\n");
                        getch();
                        return;
                }
                else{
                        printf("<- (%d) ", Table[i]->nomor_antrian);
                }
                getch();
        }

        //menampilkan isi node

printf("\n-------------------------------------------------------------------\n");
        printf("Nama Pasien\t: %s\n", Table[i]->nama);
        printf("Alamat Pasien\t: %s\n", Table[i]->alamat);
        printf("Umur Pasien\t: %d\n", Table[i]->umur);
        printf("Jenis Kelamin Pasien\t: %c\n", Table[i]->jk);
        printf("Tingkat Prioritas\t: %d\n", Table[i]->prioritas);
        printf("Nomor Antrian\t: %d\n", Table[i]->nomor_antrian);

printf("-------------------------------------------------------------------\n");
        getch();
}

//menu melihat antrian untuk umum
void lihat_antrian(){
        int i;
        if(size==-1){
                printf("Antrian Kosong!!\n");
                getch();
        }
        else{
```

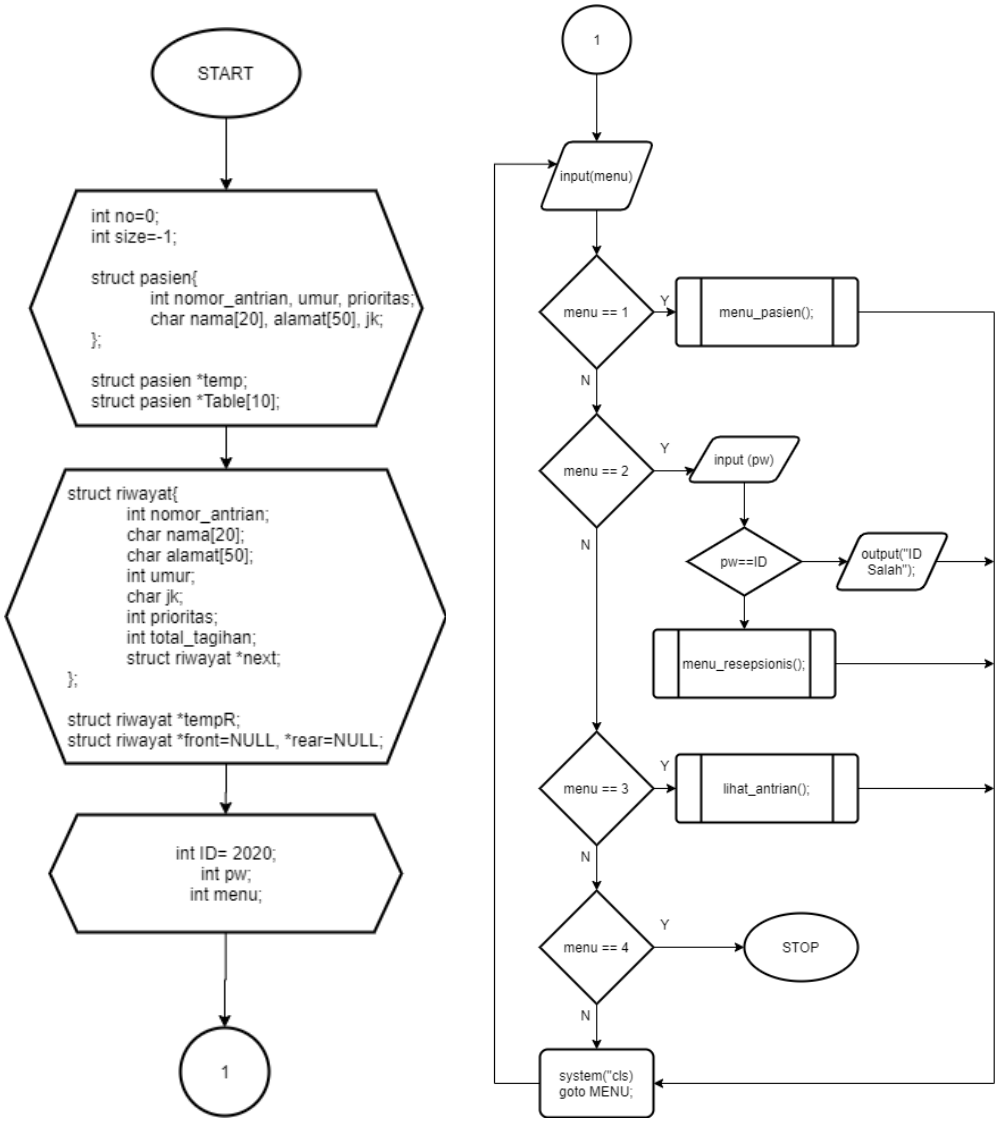
```
                for(i=0;i<=size;i++){
                        if(i==0){
                                printf("(%d) ", Table[i]->nomor_antrian);
                        }
                        else{
                                printf("<- (%d) ", Table[i]->nomor_antrian);
                        }

                }
                getch();

        }
}

int main(){
        //deklarasi variable
        int ID= 2020, pw;
        int menu;
        MENU:
        //judul

printf(".==========================================================.\n");
        printf("|          Antrian Rumah Sakit          |\n");

printf("'==========================================================\n");
        printf("| Masuk Sebagai :                       |\n");
        printf("| 1. Pasien                             |\n");
        printf("| 2. Resepsionis                        |\n");
        printf("|                                       |\n");
```

```
            printf("| Menu Lain :                                    |\n");
            printf("| 3. Lihat Antrian                               |\n");
            printf("| 4. Keluar                                      |\n");

        printf("'==============================================
        ==================\n");
            printf("Masukkan Pilihan Anda : ");
            scanf("%d", &menu);

            //switch case menu
            switch (menu){
                case 1:{
                    menu_pasien();
                    break;
                }
                case 2:{
                    printf("Masukkan ID Resepsionis : ");
                    scanf("%d", &pw);
                    //cek ID resepsionis
                    if(pw==ID){
                        system("cls");
                        menu_resepsionis();
                    }
                    else{
                        printf("ID yang anda masukkan
        salah !\n");
                        getch();
                    }
                    break;
                }
                case 3:{
                    lihat_antrian();
                    break;
```

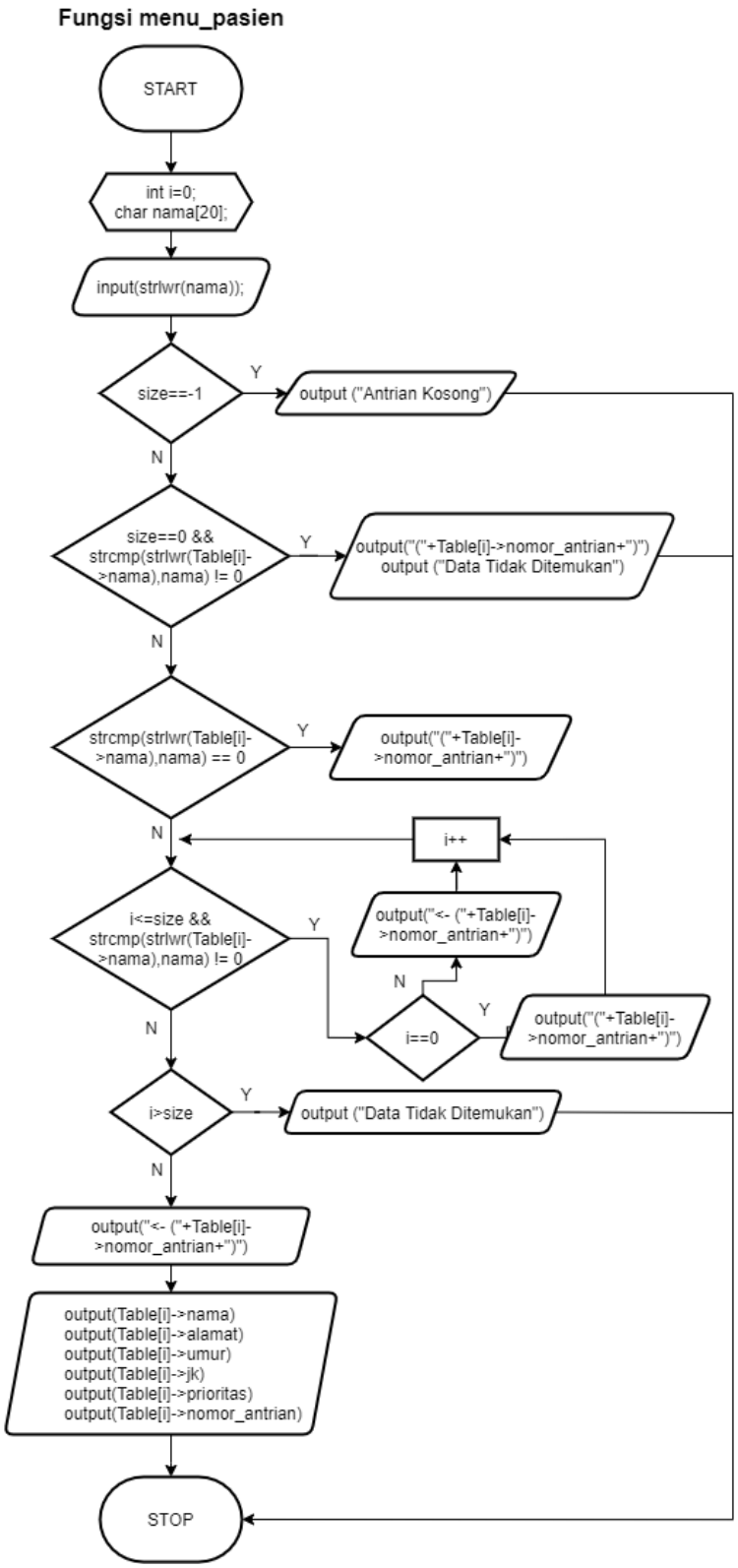
```
                }
                case 4:{
                        printf("Program Berakhir !!\n");
                        return 0;
                        break;
                }
        }
        system("cls");
        goto MENU;
        return 0;
}
```

2.2. Flowchart

1. Main Function

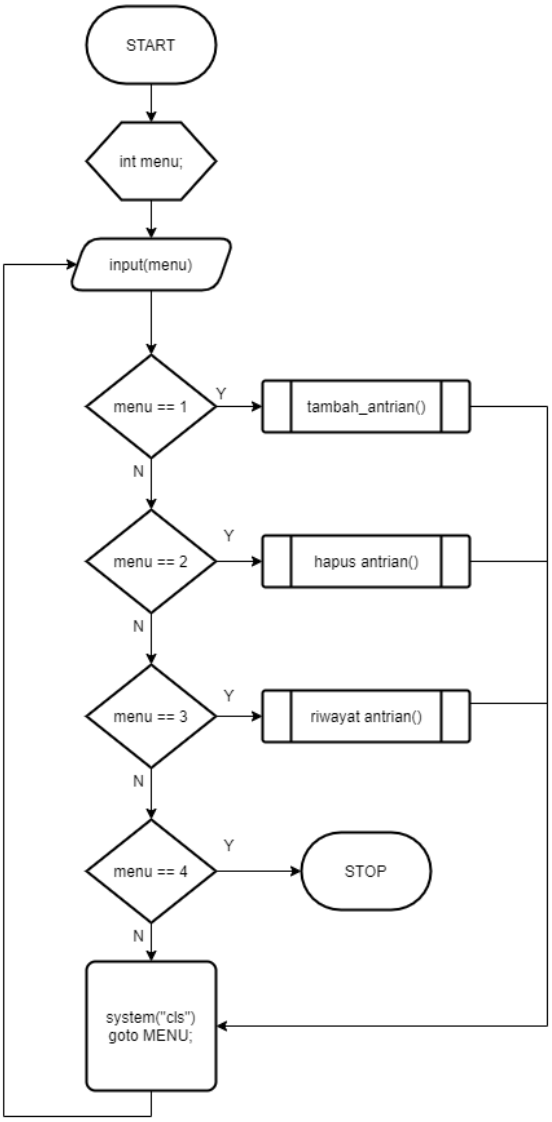


2. Menu Pasien



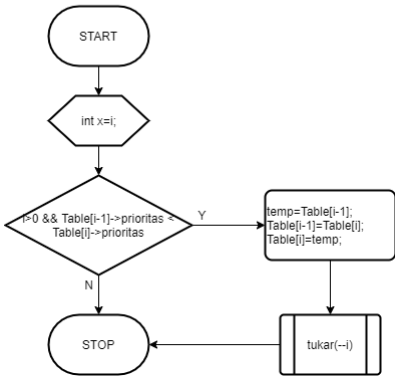
3. Menu Resepsionis

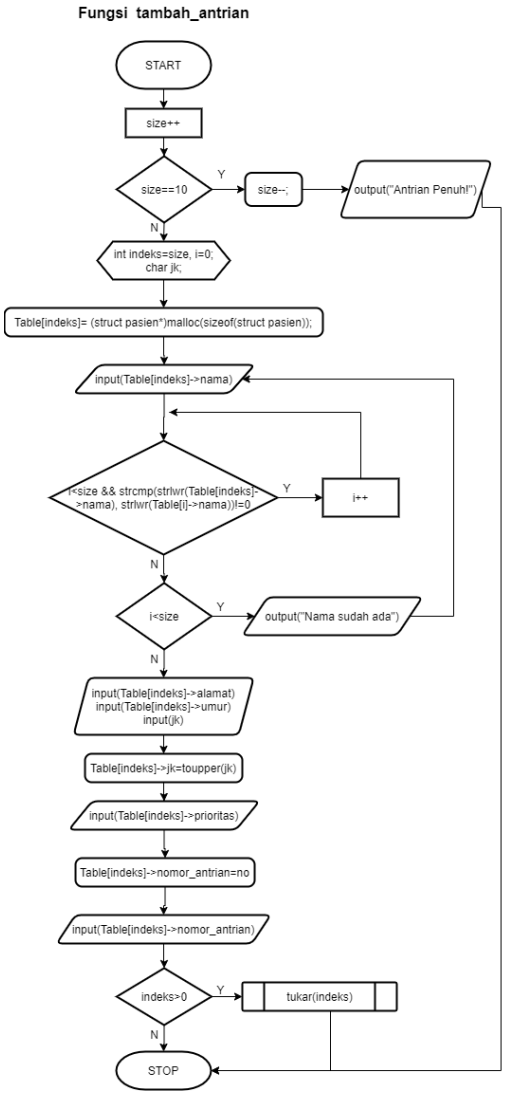
Fungsi menu_resepsionis



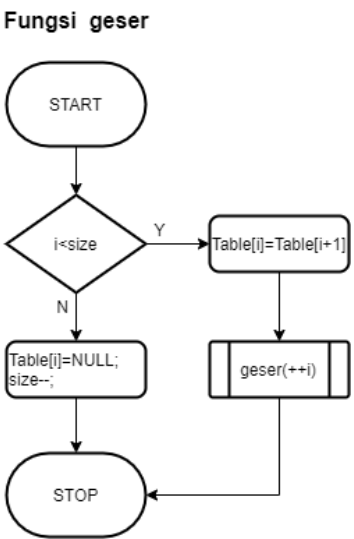
4. Tambah Antrian

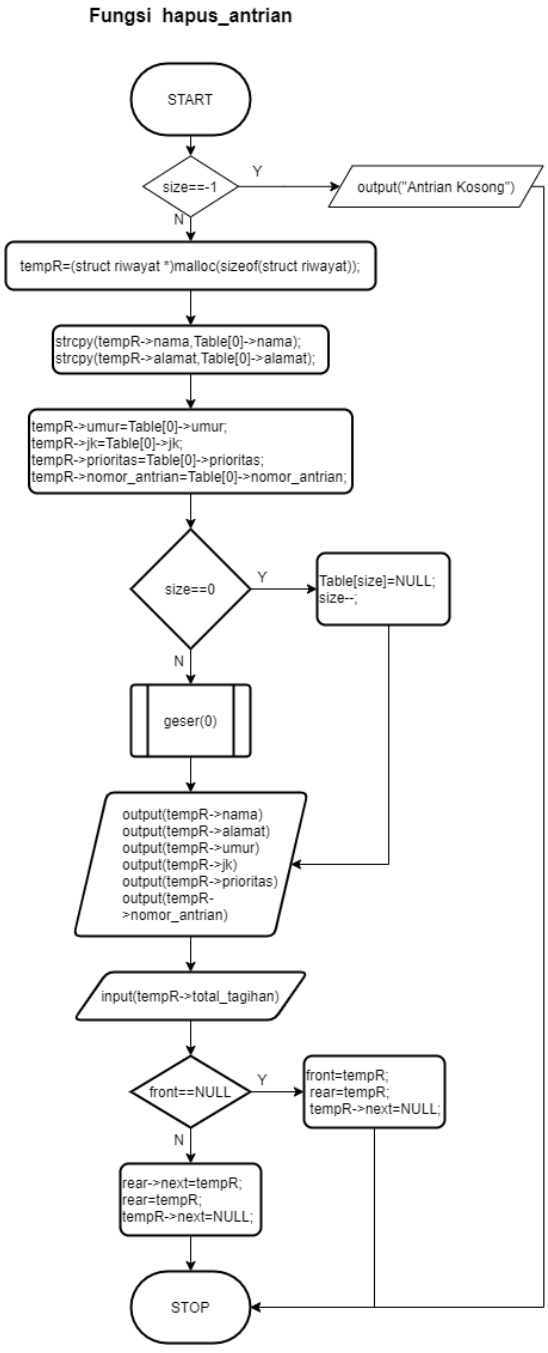
Fungsi tukar





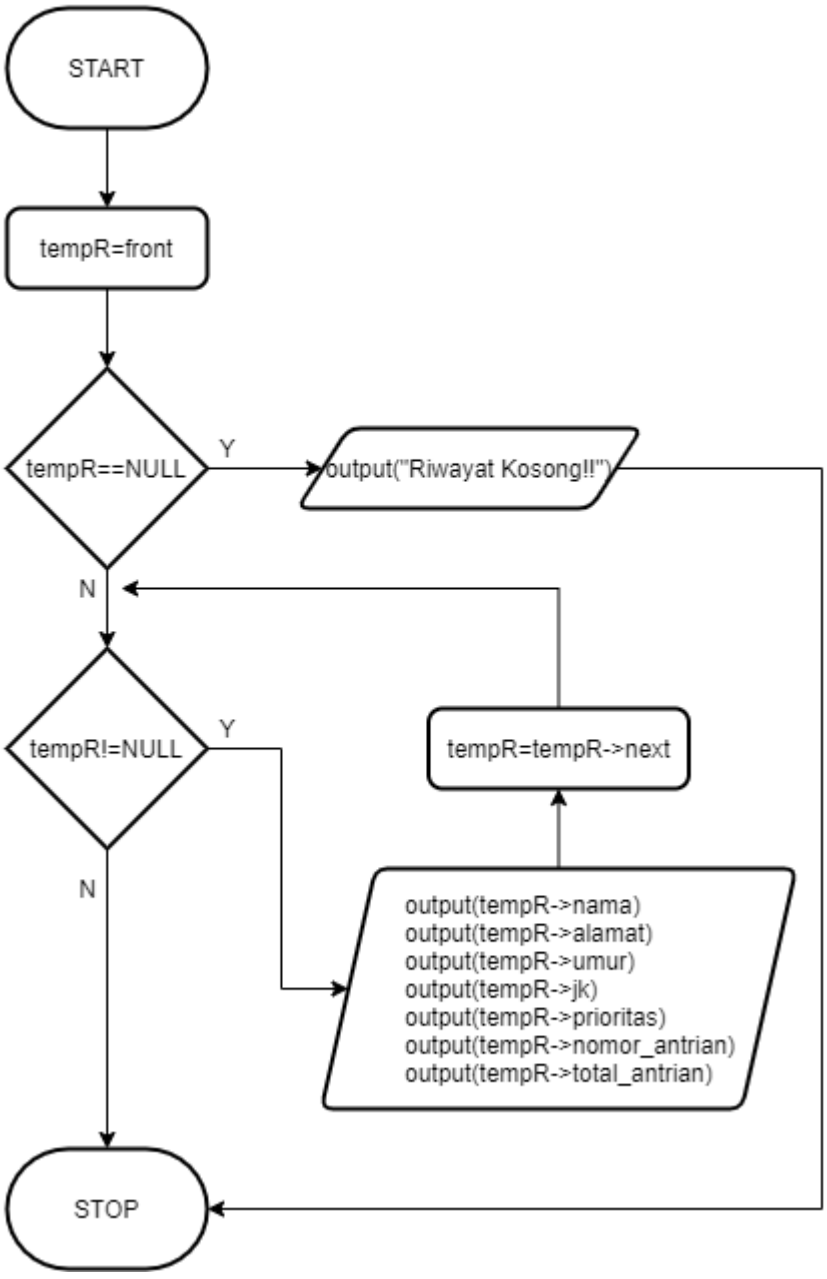
5. Hapus Antrian



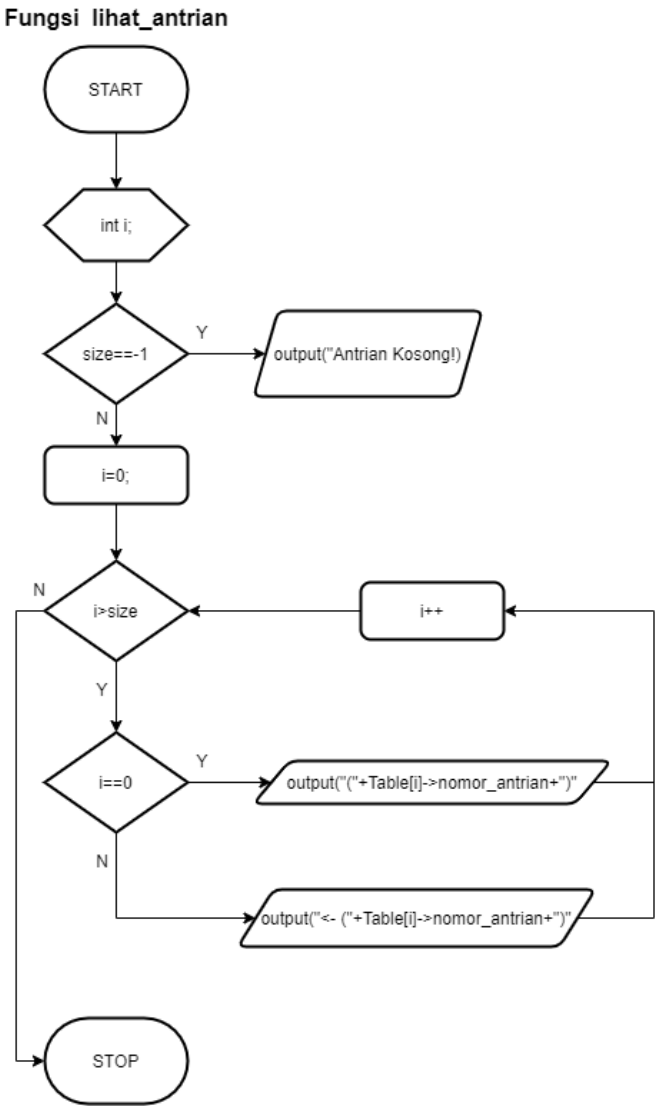


6. Riwayat Antrian

Fungsi riwayat_antrian



7. Lihat Antrian



2.3. Screenshot Program

1. Menu Awal

```
.================================================================.
|                      Antrian Rumah Sakit                       |
'================================================================'
| Masuk Sebagai :                                                |
| 1. Pasien                                                      |
| 2. Resepsionis                                                 |
|                                                                |
| Menu Lain :                                                    |
| 3. Lihat Antrian                                               |
| 4. Keluar                                                      |
'================================================================'
Masukkan Pilihan Anda :
```

2. Menu Resepsionis

```
.=================================================================.
|                     Antrian Rumah Sakit                         |
'=================================================================='
| Masuk Sebagai :                                                 |
| 1. Pasien                                                       |
| 2. Resepsionis                                                  |
|                                                                 |
| Menu Lain :                                                     |
| 3. Lihat Antrian                                                |
| 4. Keluar                                                       |
'=================================================================='
Masukkan Pilihan Anda : 2
Masukkan ID Resepsionis : 2020
```

Untuk dapat lanjut ke halaman menu resepsionis, kita harus memasukkan ID dari resepsionis agar yang dapat mengakses halaman hanya resepsionis saja.

```
.=================================================================.
|                     Antrian Rumah Sakit                         |
'=================================================================='
| 1. Tambah Antrian                                               |
| 2. Hapus Antrian                                                |
| 3. Riwayat Pasien                                               |
| 4. Kembali                                                      |
'=================================================================='
Masukkan Pilihan Anda : 1
```

3. Tambah Antrian

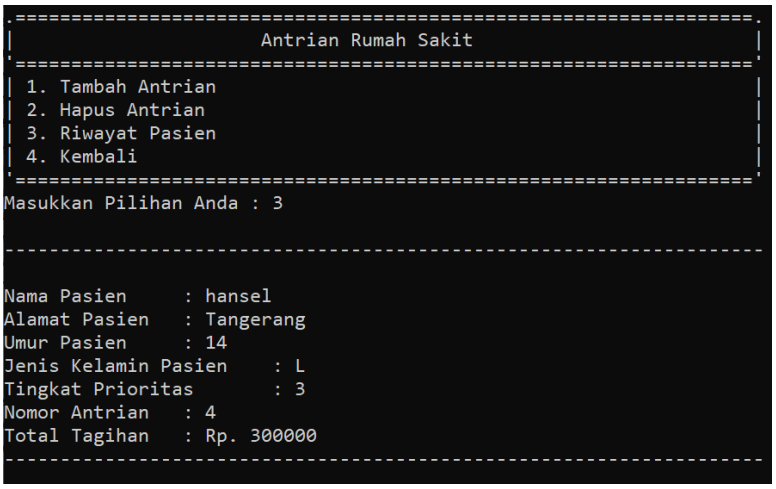
```
.=================================================================.
|                     Antrian Rumah Sakit                         |
'=================================================================='
| 1. Tambah Antrian                                               |
| 2. Hapus Antrian                                                |
| 3. Riwayat Pasien                                               |
| 4. Kembali                                                      |
'=================================================================='
Masukkan Pilihan Anda : 1
-------------------------------------------------------------------
Masukkan nama pasien    : Vinna
Masukkan alamat pasien  : Tangerang
Masukkan Umur Pasien    : 19
Masukkan jenis kelamin pasien <p/l>     : p
-------------------------------------------------------------------
1. Ringan
2. Sedang
3. Berat
Masukkan Tingkat Keparahan Pasien       : 1
-------------------------------------------------------------------
Nomor Antrian Pasien    : 1
-------------------------------------------------------------------
```

Kita akan menambahkan 4 orang dalam antrian yaitu Vinna dengan prioritas ringan, kayla dengan prioritas sedang, kevin dengan prioritas ringan, dan hansel dengan prioritas berat.

4. Hapus Antrian

Antrian yang akan dihapus adalah antrian yang berada di awal array yaitu antrian yang memiliki prioritas tertinggi dan juga lebih awal datang.

5. Riwayat Pasien



Riwayat pasien akan menampilkan pasien yang telah selesai ditangani, karena baru satu pasien yang telah terhapus antriannya maka disini program hanya menampilkan satu data.

6. Menu Pasien

```
.=====================================================================.
|                       Antrian Rumah Sakit                           |
'====================================================================='
| Masuk Sebagai :                                                     |
| 1. Pasien                                                           |
| 2. Resepsionis                                                      |
|                                                                     |
| Menu Lain :                                                         |
| 3. Lihat Antrian                                                    |
| 4. Keluar                                                           |
'====================================================================='
Masukkan Pilihan Anda : 1
.=====================================================================.
|                       Antrian Rumah Sakit                           |
'====================================================================='
Masukkan Nama Anda : vinna
(2) <- (1)
---------------------------------------------------------------------
Nama Pasien     : vinna
Alamat Pasien   : Tangerang
Umur Pasien     : 19
Jenis Kelamin Pasien    : P
Tingkat Prioritas       : 1
Nomor Antrian   : 1
---------------------------------------------------------------------
```

Menu Pasien menampilkan data diri dan sisa antrian sebelum pasien tersebut dengan memasukkan nama pasien.

7. Lihat Antrian

```
.=====================================================================.
|                       Antrian Rumah Sakit                           |
'====================================================================='
| Masuk Sebagai :                                                     |
| 1. Pasien                                                           |
| 2. Resepsionis                                                      |
|                                                                     |
| Menu Lain :                                                         |
| 3. Lihat Antrian                                                    |
| 4. Keluar                                                           |
'====================================================================='
Masukkan Pilihan Anda : 3
(2) <- (1) <- (3)
```

Menu lihat antrian menampilkan nomor antrian sesuai urutan antriannya.

2.4. Penjelasan Program

1. Struct Pasien

```c
//nomor antrian dan size array
int no=0, size=-1;

//struct antrian pasien
struct pasien{
    int nomor_antrian;
    char nama[20];
    char alamat[50];
    int umur;
    char jk;
    int prioritas;
};

struct pasien *temp;
struct pasien *Table[10];
```

Pada awal program, dilakukan pendeklarasian dan pengisian nilai terhadap variabel gloobal, yakni no=0 untuk nomor antrian dan size=-1 untuk ukuran array.

Lalu, dibuat struct pasien yang berisikan variabel dari data-data pasien, yaitu nomor_antrian yang bertipe integer, array of char untuk menampung nama pasien yang berukuran 20, array of char untuk menampung alamat pasien yang berukuran 50, umur yang bertipe int, jk yang bertipe char, dan prioritas yang bertipe integer.

Setelah itu, dibuat pointer baru yang bernama temp yang digunakan sebagai pointer bantuan dalam program dan Table[10] yang merupakan pointer yang menunjuk setiap data strcut yang ada pada indeks pointer tertentu sebagai penggambaran dari queue menggunakan array.

2. Struct Riwayat

```
//struct riwayat pasien
struct riwayat{
    int nomor_antrian;
    char nama[20];
    char alamat[50];
    int umur;
    char jk;
    int prioritas;
    int total_tagihan;
    struct riwayat *next;
};

struct riwayat *tempR;
struct riwayat *front=NULL, *rear=NULL;
```

Struct dengan nama riwayat ini menyimpan 8 data dengan tipe: 4 data bertipe integer, 3 data bertipe char dan 1 data bertipe struct dimana dati ini dimaksudkan untuk menunjuk next pada linked list.

Tipe data int memiliki variabel berupa nomor_antrian, umur, prioritas, dan total_tagihan. Nomor antrian akan didapatkan dari fungsi tambah antrian. Umur adalah sebuah variabel yang menyimpan umur pasien yang akan didapatkan oleh program dari sebuah tipe data struct dengan nama pasien. Prioritas merupakan sebuah variabel yang digunakan unntuk menyimpan skala prioritas dari pasien yang terdaftar. Total_tagihan merupakan sebuah vaariabel yang akan menyimpan total tagihan biaya yang harus ditanggung pasien

Tipe data char memiliki variabel berupa nama, alamat, dan jk. Semua variabel ini akan dipanggil dari sebuah data struct bernama pasien.

3. Menu Awal

```
MENU:
//judul
printf(".============================================================.\n");
printf("|                      Antrian Rumah Sakit                   |\n");
printf("'============================================================'\n");
printf("| Masuk Sebagai :                                            |\n");
printf("| 1. Pasien                                                  |\n");
printf("| 2. Resepsionis                                             |\n");
printf("|                                                            |\n");
printf("| Menu Lain :                                                |\n");
printf("| 3. Lihat Antrian                                           |\n");
printf("| 4. Keluar                                                  |\n");
printf("'============================================================'\n");
```

Didalam menu awal atau main function, ditampilkan terlebih dahulu menu atau fungsi apa saja yang dimiliki program menggunakan fungsi printf. Pada program ini terdapat menu untuk masuk sebagai pasien atau resepsionis, terdapat juga menu lain seperti lihat antrian dan keluar dari program.

```
printf("Masukkan Pilihan Anda : ");
scanf("%d", &menu);
```

Setelah itu pengguna akan diminta menginputkan sebuah nilai menggunakan fungsi scanf() yang disimpan di variabel menu sebagai menu yang dipilih oleh pengguna.

```
//switch case menu
switch (menu){
    case 1:{
        menu_pasien();
        break;
    }
    case 2:{
        printf("Masukkan ID Resepsionis : ");
        scanf("%d", &pw);
        //cek ID resepsionis
        if(pw==ID){
            system("cls");
            menu_resepsionis();
        }
        else{
            printf("ID yang anda masukkan salah !\n");
            getch();
        }
        break;
    }
    case 3:{
        lihat_antrian();
        break;
    }
    case 4:{
        printf("Program Berakhir !!\n");
        return 0;
        break;
    }
}
```

Lalu akan dilakukan switch case sesuai isi dari variable menu, pada case pertama program akan memanggil fungsi menu_pasien(), pada case kedua program akan memanggul fungsi menu_resepsionis(). Namun, pemanggilan menu resepsionis memiliki syarat yang diwakilkan dengan if else yaitu kita harus memasukkan password atau ID terlabih dahulu, jika ID benar, maka program akan memanggil fungsi menu_resepsionis tapi jika ID salam, maka akan diberikan keterangan bahwa ID yang dimasukkan salah. Pada case 3, program akan memanggil fungsi lihat_antrian() dan pada case 4, akan diberi keterangan bahwa program berakhir lalu ada statement return 0 untuk mengakhiri program.

```
system("cls");
goto MENU;
return 0;
```

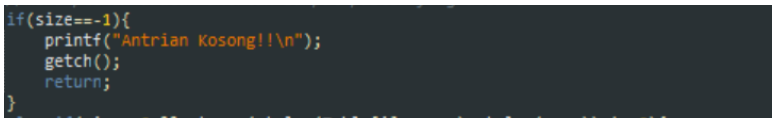
Setelah dilakukan switch case, program akan mengulang dengan menghapus layar terlebih dahulu, lalu kembali ke bagian menu dengan menggunakan statement goto MENU;

4. Menu Pasien

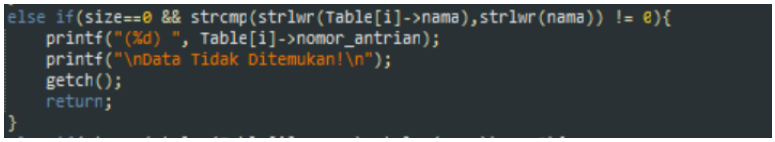
Fumgsi menu_pasien adalah sebuah fungsi yang akan dieksekusi saat user memilih 1 pada menu utama, fungsi ini menampilkan jalur nomor antrian yang ada sebelum nama pasien yang telah dimasukkan.

Didalam fungsi ini terdapat deklarasi dari 2 variabel yang akan digunakkan dalam fungsi ini, variabel i dengan tipe integer digunakan dalam perulangan dan variabel nama dengan array bernilai maksimum 20 bertipe char digunakkan untuk menyimpan nama pasien yang akan dicari data diri serta urutannya didalam antrian.

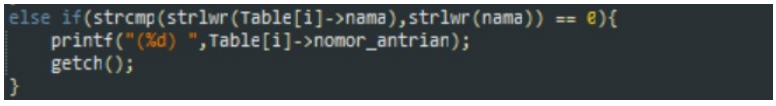
Selanjutnya program akan menerima nama pasien dan menyimpannya dalam variabel nama.



lalu program akan melakukan pengecekan untuk menentukan apakah antrian kosong atau tidak, jika antrian kosong maka program akan mencetak pernyataan bahwa antrian kosong lalu akan dilakukan return ke menu awal.



Jika antrian memiliki jumlah elemen satu atau nilai size=0, dan data yang ada tidak sema dengan yang dicari, maka program akan mencerak isi antrian dan memberikan pernyataan bahwa data nama tersebut tidak ditemukan.



Jika antrian memiliki elemen lebih dari satu dan data nama ditemukan pada indeks pertama atau indeks 0, maka program akan mencetak nomor antriannya, namun tidak dilakukan return ke menu awal karena akan dilakukan pencetakan dara diri.

```
while(i<=size && strcmp(strlwr(Table[i]->nama),strlwr(nama)) != 0){
    if(i==0){
        printf("(%d) ", Table[i]->nomor_antrian);
    }
    else{
        printf("<- (%d) ", Table[i]->nomor_antrian);
    }
    getch();
    i++;
}
```

Selain kondisi diatas, maka akan dilakukan perulangan yang akan berakhir ketika data nama pada indeks i sama dengan nama yang dicari dan i masih lebih kecil sama dengan size arraynya. pada setiap perulangan akan dicetak nomor pasiennya sehingga membentuk sebuah jalur antrian.

```
if(i>size){
    printf("\nData Tidak Ditemukan!\n");
    getch();
    return;
}
else{
    printf("<- (%d) ", Table[i]->nomor_antrian);
}
```

Setelah perulangan berhenti, maka akan dilakukan pengecekan, jika perulangan berhenti karena nilai i sudah lebih besar dari size, berarti data tidak ditemukan dalam antrian. Namun jika tiak, maka akan dicetak nomot antrian untuk data yang dicari.

Setelah itu akan dilakukan pencetakan data diri yang berada pada Tabel[i] menggunakan fungsi printf().

5. Menu Resepsionis

```
intf(".============================================================.\n");
intf("|                    Antrian Rumah Sakit                     |\n");
intf("'============================================================'\n");
intf("| 1. Tambah Antrian                                          |\n");
intf("| 2. Hapus Antrian                                           |\n");
intf("| 3. Riwayat Pasien                                          |\n");
intf("| 4. Kembali                                                 |\n");
intf("'============================================================'\n");
```

Didalam menu resepsionis, ditampilkan terlebih dahulu menu atau fungsi apa saja yang dimiliki program menggunakan fungsi printf. Pada program ini terdapat menu untuk menambah antrian, menghapus antrian dan melihat riwayat pasien.

```
printf("Masukkan Pilihan Anda : ");
scanf("%d", &menu);
```

Setelah itu pengguna akan diminta menginputkan sebuah nilai menggunakan fungsi scanf() yang disimpan di variabel menu sebagai menu yang dipilih oleh pengguna.

```
//switch case menu
switch (menu){
    case 1:{
        printf("--------------------------------------------------------------------------\n");
        tambah_antrian();
        break;
    }
    case 2:{
        printf("--------------------------------------------------------------------------\n");
        hapus_antrian();
        break;
    }
    case 3:{
        riwayat_antrian();
        break;
    }
    case 4:{
        return;
        break;
    }
}
```

Lalu akan dilakukan switch case sesuai isi dari variable menu, pada case pertama program akan memanggil fungsi tambah_antrian(), pada case kedua program akan memanggul fungsi hapus_antrian(), Pada case 3, program akan memanggil fungsi riwayat_antrian() dan pada case 4, ada statement return yang berfungsi untuk kembali ke menu awal.

```
system("cls");
goto MENU;
return 0;
```

Setelah dilakukan switch case, program akan mengulang dengan menghapus layar terlebih dahulu, lalu kembali ke bagian menu pasien dengan menggunakan statement goto MENU;

6. Fungsi Tambah Antrian

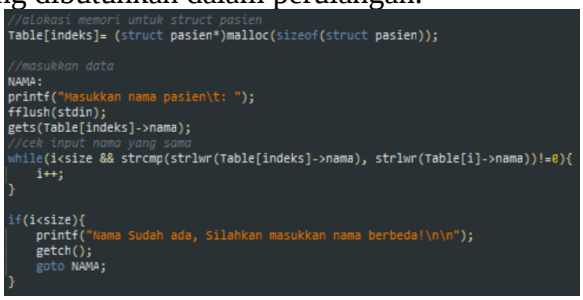
Fungsi ini berfungsi untuk menambah antrian pasien dan menempatkannya di indeks paling akhir dari antrian.

```
//indeks dalam array table
size++;

//cek antrian penuh
if(size==10){
    size--;
    printf("\nAntrian penuh!\n");
    return;
}
```

Nilai size terlebih dahulu ditambah satu, tapi sebelum menambah antrian ke array, dilakukan pengecekan terlebih dahulu apakah antrian sudah penuh atau belum, jika sudah penuh maka akan diberi keterangan dan program akan kembali ke menu pasien.

Jika antrian belum penuh, program akan mendeklarasikan indeks dalam array, yang nilainya adalah size, variabel jk yang bertipe char

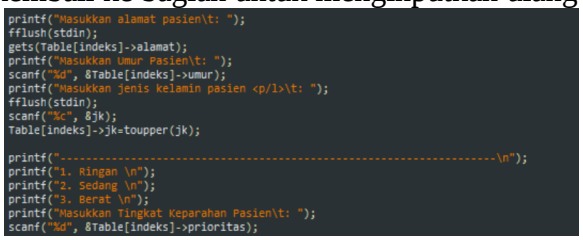
yang menyimpan jenis kelamin, dan variabel i bertipe char yang diisi oleh nilai 0 yang dibutuhkan dalam perulangan.



```
//alokasi memori untuk struct pasien
Table[indeks]= (struct pasien*)malloc(sizeof(struct pasien));

//masukkan data
NAMA:
printf("Masukkan nama pasien\t: ");
fflush(stdin);
gets(Table[indeks]->nama);
//cek input nama yang sama
while(i<size && strcmp(strlwr(Table[indeks]->nama), strlwr(Table[i]->nama))!=0){
    i++;
}

if(i<size){
    printf("Nama Sudah ada, Silahkan masukkan nama berbeda!\n\n");
    getch();
    goto NAMA;
}
```

Lalu, dilakukan pengalokasian memori untuk Table[indeks]. Kita akan menginputkan nama pasien yang datanya disimpan dalam Table[indeks]->nama. Lalu, dilakukan penelusuran terhadap data nama yang ada pada Tabel dengan indeks i dengan melakukan perulangan selama data nama pada tabel tidak sama dengan nama yang dicari, pada setiap perulangan nilai i bertambah.

Selanjutnya, dilakukan pengecekan kondisi apakah i<size untuk mengecek apakah nama tersebut sudah terdaftar atau belum. Jika iya, maka program akan mencetak bahwa nama sudah ada dan program akan kembali kembali ke bagian untuk menginputkan ulang nama.

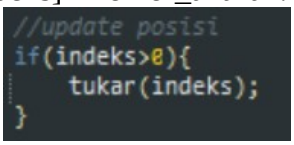


```
printf("Masukkan alamat pasien\t: ");
fflush(stdin);
gets(Table[indeks]->alamat);
printf("Masukkan Umur Pasien\t: ");
scanf("%d", &Table[indeks]->umur);
printf("Masukkan jenis kelamin pasien <p/l>\t: ");
fflush(stdin);
scanf("%c", &jk);
Table[indeks]->jk=toupper(jk);

printf("-------------------------------------------------------------------------\n");
printf("1. Ringan \n");
printf("2. Sedang \n");
printf("3. Berat \n");
printf("Masukkan Tingkat Keparahan Pasien\t: ");
scanf("%d", &Table[indeks]->prioritas);
```

Setelah keluar dari pengecekan kondisi, lanjut untuk menginputkan alamat yang akan disimpan di Table[indeks]->alamat, menginputkan umur yang akan disimpan di Table[indeks]->umur, menginputkan jenis kelamin pasien yang akan disimpan dalam Table[indeks]->jk dan tingkat keparahan yang akan disimpan dalam Table[indeks]->prioritas.

```
//menamilkan nomor antian
printf("-------------------------------------------------------------------------\n");
no++;
Table[indeks]->nomor_antrian=no;
printf("Nomor Antrian Pasien\t: %d\n", Table[indeks]->nomor_antrian);
printf("-------------------------------------------------------------------------\n");
getch();
```

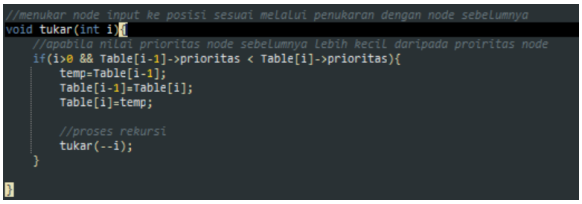
Dilanjutkan dengan menampilkan nomor antrian. Nilai dari variabel no akan bertambah, Table[indeks]->nomor_antrian akan diisi oleh nilai no, setelah itu program akan mencetak nomor antrian yang tersimpan dalam Table[indeks]->nomor_antrian.



```
//update posisi
if(indeks>0){
    tukar(indeks);
}
```

Untuk membuat antrian berdasarkan prioritas, dilakukan update posisi. Dilakukan pengecekan apakah indeks>0 yang berarti data antrian lebih dari satu. Jika iya, maka fungsi tukar akan dipanggil dengan parameter indeks.

7. Fungsi Tukar



fungsi tukar bertipe void yang menerima parameter variabel i bertipe integer yang merupakan indeks akhir antian yang akan dipindah posisinya. Fungsi ini berfungsi untuk menukar node input ke posisi sesuai melalui penukaran dengan node sebelumnya berdasarkan tingkat keparahan yang dialami pasien. Dalam fungsi ini akan dicek kondisi apakah nilai prioritas node indeks sebelumnya lebih kecil daripada prioritas node. Jika iya, maka akan dilakukan proses pertukaran nilai struct yang ditunjuk oleh pointer Table[i-1] dengan nilai yang ditunjuk oleh pointer Table[i]. Setelah proses pertukaran dilakukan, maka akan dilakukan proses rekursi untuk memanggil fungsi itu sendiri dengan perintah tukar(--i). Untuk melakukan pengecekan dan pertukaran kembali terhadap indeks indeks sebelumnya.

8. Fungsi hapus_antrian

Fungsi ini merupakan dungsi yang digunakan untuk mengeluarkan data pada awal antrian dan memasukkannya kedalam data riwayat antrian.

Sebelum dilakukan penghapusan data antrian, akan dicek terlebih apakah antrian kosong atau tidak karena jika antrian kosong maka tidak aka nada data yang bisa dihapus, maka program akan kembali ke menu pasien.

```
//menyimpan antrian yang ingin dihapus ke node riwayat
//membuat node riwayat
tempR=(struct riwayat *)malloc(sizeof(struct riwayat));
strcpy(tempR->nama,Table[0]->nama);
strcpy(tempR->alamat,Table[0]->alamat);
tempR->umur=Table[0]->umur;
tempR->jk=Table[0]->jk;
tempR->prioritas=Table[0]->prioritas;
tempR->nomor_antrian=Table[0]->nomor_antrian;
```

Hapus antrian dilakukan jika nomor antrian pasien tersebut telah dipanggil atau telah tiba gilirannya. Maka data pasien tersebut akan dihapus namun tak menghilang, melainkan disimpan dalam riwayat sebagai arsip. Pertama kita mengalokasikan memori terlebih dahulu untuk membuat struct riwayat yang memiliki pointer berupa tempR Lalu dengan strcpy atau stringcopy kita menyalin elemen data dalam antrian yang telah dihapus meliputi nama, alamat. Sedangkan umur, jenis kelamin, prioritas, dan nomor antrian akan disalin ke struct antrian menggunakan penugasan.

```
//swap dan hapus node
if(size==0){
    Table[size]=NULL;
    size--;
}
else{
    geser(0);
}
```

Lalu ada percabangan untuk fungsi swap dan hapus node, jika antrian kosong maka tak ada yang terjadi namun jika sebaliknya maka antrian setelah yang dihapus akan menggantikan posisi yang telah dihapus dengan memanggil fungsi geser() dan parameternya bernilai 0.

9. Fungsi Geser

```
//fungsi untuk menggeser indeksnya
void geser(int i){
    if(i<size){
        //swapping
        Table[i]=Table[i+1];
        //proses rekursi
        geser(++i);
    }
    else {
        Table[i]=NULL;
        size--;
    }
}
```

Fungsi ini digunakan untuk menggeser urutan nomor antrian nantinya. Dalam fungsi ini terdapat percabangan kondisi if yang jika variabel i lebih kecil daripada ukuran antrian, maka elemen antrian akan di swap sehingga data pada tabel indeks ke- i+1 akan berubah menjadi terletak pada indeks i lalu akan terjadi proses rekursi dengan memanggil kembali fungsi geser untuk menggeser indeks i+1 dan seterusnya sampai i==size, apabila i==size maka akan dijalankan simple casenya yaitu Table[i]=NULL, atau datanya dihapus dan sizenya dikurang satu.

10. Fungsi Riwayat Antrian/pasien

Fungsi riwayat_antrian adalah sebuah fungsi yang akan dipanggil saat user memilih nomor 3 pada menu resepsionis, fungsi ini mengembalikan data data yang sudah disimpan dalam sistem untuk menampilkan biaya yang harus ditanggung oleh pasien serta data data yang terdaftar atas nama pasien.

```
if(tempR==NULL){
    printf("Riwayat Kosong !!\n");
    getch();
    return;
}
```

Fungsi ini diawali dengan melakukan pengecekan terhadap variabel tempR, diamana jika variabel tersebut bernilai NULL maka program akan mengembalikan output berupa sebuah message “Antrian Kosong” dan program akan kembali ke menu resepsionis.

```
//perulangan utnuk menampilkan riwayat antrian
while(tempR!=NULL){

    printf("Nama Pasien\t: %s\n", tempR->nama);
    printf("Alamat Pasien\t: %s\n", tempR->alamat);
    printf("Umur Pasien\t: %d\n", tempR->umur);
    printf("Jenis Kelamin Pasien\t: %c\n", tempR->jk);
    printf("Tingkat Prioritas\t: %d\n", tempR->prioritas);
    printf("Nomor Antrian\t: %d\n", tempR->nomor_antrian);
    printf("Total Tagihan\t: Rp. %d\n", tempR->total_tagihan);
    printf("-------------------------------------------------------------------------\n");
    tempR=tempR->next;
    getch();
}
```

Jika tempR memiliki nilai maka program akan melakukan proses perulangan untuk menampilkan semua data riwayat antrian. Selama nilai dari tempR tidak NULL maka program akan mengembalikan output berupa:

1. Nama Pasien yang diambil dari pointer tempR->nama
2. Alamat Pasien yang diambil dari pointer tempR->alamat
3. Umur Pasien yang diambil dari pointer tempR->umur
4. Kelamin Pasien yang diambil dari pointer tempR->jk
5. Tingkat Prioritas yang diambil dari pointer tempR->prioritas
6. Nomor Antrian yang diambil dari pointer tempR->nomor_antrian
7. Total Tagihan yang diambil dari pointer tempR->total_tagihan

Setelah itu program akan menggeser pointer tempR sehingga menunjuk node selanjutnya.

11. Fungsi Lihat Antrian

Fungsi ini digunakan untuk melihat antrian yang terdaftar pada sistem. Fungsi ini mendeklarasikan sebuah variabel bertipe integer dengan nama i sebagai penunjuk indek dalam perulangan.

```
if(size==-1){
    printf("Antrian Kosong!!\n");
    getch();
}
```

Pertama program akan melakukan proses pengecekan kondisi dimana jika variabel size bernilai -1 maka program akan mengembalikan output berupa message “Antrian Kosong”.

```
else{
    for(i=0;i<=size;i++){
        if(i==0){
            printf("(%d) ", Table[i]->nomor_antrian);
        }
        else{
            printf("<- (%d) ", Table[i]->nomor_antrian);
        }

    }
    getch();

}
```

Jika tidak maka program akan melakukan perulangan untuk menampilkan antrian yang ada. Untuk nilai i = 0 sampai dengan nilai kurang dari sama dengan size dan nilai bertambah di setiap perulangan maka program akan melakukan pengecekan kembali, dimana jika nilai i saat perulangan adalah 0, program akan mengembalikan nomor anttrian pertama dari linked list nomor_antrian jika tidak program akan menampilkan message “<-” sebelum nomor antrian yang ada pada sistem.

BAB III

KESIMPULAN

Queue merupakan sebuah struktur data yang menggunakan konsep FIFO (first in first out) dimana data yang dimasukkan pertama adalah data yang pertama keluar dari queue. Struktur data ini dapat diimplementasikan untuk membuat program seperti antrian bank, antrian kasir restoran dan antrian lainnya. Namun, ada kalanya dimana sebuah data dalam antrian tersebut perlu diprioritaskan. Contohnya pada saat kita berada di *amusement park*, pengunjung yang memiliki kartu vip akan dilayani terlebih dahulu. Contoh lainnya adalah pada antrian di rumah sakit, tidak mungkin pasien yang parah seperti kecelakaan harus menunggu sampai pasien- pasien sebelumnya selesai untuk mendapat perawatan, tentunya pasien ini harus diprioritaskan. Untuk itu, kita dapat menggunakan priority queue.

Priority queue merupakan sebuah struktur data yang didasarkan dari struktur data queue. Priority queue memiliki basis HPIFO (*Highest Priority in First Out*), berbeda dengan queue yang memiliki basis FIFO (*First in First Out*). Dalam priority queue, elemen yang memiliki prioritas tertinggi lah yang akan diambil atau dihapus. Agar data dengan prioritas lebih tinggi dapat berada didepan atau lebih dulu dari data dengan prioritas lebih rendah kita dapat melakukan pertukaran data secara berulang dengan memanfaatkan proses rekursi.

Fungsi rekursif merupakan sebuah fungsi atau method yang memanggil dirinya sendiri baik secara langsung ataupun secara tak langsung (lewat fungsi lain). Ada dua aspek yang harus dimiliki oleh fungsi rekursi. Sebuah fungsi rekursi harus mengetahui kapan harus berhenti (simple case) dan kapan harus memanggil dirinya kembali.

Dalam program ini, kita mengimplementasikan struktur data yaitu priority queue untuk membuat program antrian rumah sakit yang memiliki tiga tingkat keparahan atau tingkat prioritas yaitu ringan, sedang, dan berat dan juga memanfaatkan rekusrif untuk memindahkan datanya.

DAFTAR PUSTAKA

Irfani, Miftakhul. 2013. Pengimplementasian Rekursif. Jurnal Informatika Universitas Bengkulu. 8(2), 43-47.

Nurcholis, Aditya., Simon Batara N, Mohamad Octamanullah. Penerapan struktur data Heap Priority Queue pada algoritma Djikstra untuk mendapatkan kompleksitas O((n + m)log n). Makalah STMIK Institut Teknologi Bandung.

Thareja, R. 2014, Data Structure Using C second edition, Oxford University Press, India