

UTS ALGORITMA DAN PEMROGRAMAN LANJUT
“PENERAPAN METODE QUICK SORT DAN LINEAR SEARCH
DALAM PENGELOMPOKAN DATA PASIEN
DI RUMAH SAKIT”



Disusun Oleh :

KELOMPOK BLOKCHAIN

Chintia Liu Wintin / 2109106008

Reihan Al Sya'ban / 2109106051

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK UNIVERSITAS MULAWARMAN
2022

KATA PENGANTAR

Puji syukur kehadiran Tuhan Yang Maha Esa atas berkat, rahmat, serta limpahan karunia-Nya sehingga penulis dapat menyelesaikan tugas proposal yang berjudul “Penerapan Metode Quick Sort dan Linear Search dalam Pengelompokkan Data Pasien di Rumah Sakit” dengan tepat waktu. Proposal ini disusun untuk memenuhi Tugas Ujian Tengah Semester Mata Kuliah Algoritma dan Pemrograman Lanjut. Selain itu, proposal ini juga bertujuan untuk menambah wawasan dan mengasah kemampuan penulis mengenai cara membuat program yang dapat bermanfaat bagi masyarakat luas khususnya bagi tenaga kesehatan, salah satunya dalam mempermudah pekerjaan seorang admin di rumah sakit. Dalam menyusun makalah ini, tidak sedikit kesulitan dan hambatan yang penulis hadapi.

Oleh karena itu, penulis mengucapkan terima kasih sebesar-besarnya kepada Ibu Ir. Novianti Puspitasari, M. Eng, selaku dosen pengampu mata kuliah Algoritma dan Pemrograman Lanjut yang telah bersedia memberikan arahan, bimbingan, dan saran, sehingga penulis mampu menyelesaikan proposal ini dengan baik. Dan ucapan terima kasih juga disampaikan kepada semua pihak yang telah membantu penulis dalam proses penyelesaian makalah ini.

Penulis menyadari bahwa masih banyak kekurangan dan kesalahan dalam makalah ini. Oleh karena itu, segala kritikan dan saran yang bersifat membangun sangat penulis harapkan dan tentunya akan penulis terima dengan baik.

Samarinda, 26 Maret 2022

Penulis

TAKARIR

Berikut ini adalah padanan kata bahasa asing dalam bahasa Indonesia yang digunakan dalam menyusun makalah kami adalah sebagai berikut :

<i>Admin</i>	: Administrator yang bertugas untuk mengurus permasalahan administrasi
<i>Ascending</i>	: Pengurutan data yang dilakukan dari data terendah hingga data tertinggi
<i>Create</i>	: Buat
<i>Delete</i>	: Hapus
<i>Descending</i>	: Pengurutan data yang dilakukan dari data tertinggi hingga data terendah
<i>Enter</i>	: Suatu perintah untuk membuat baris baru
<i>Indeks</i>	: Angka yang digunakan untuk mengakses di dalam deret bilangan
<i>Input</i>	: Masukkan
<i>Key</i>	: Kunci atau kata kunci
<i>Kodingan</i>	: Pengolahan kode yang dituliskan menggunakan bahasa pemrograman tertentu
<i>Main Menu</i>	: Menu utama
<i>Manage Menu</i>	: Menu kelola
<i>Online</i>	: Dalam Jaringan
<i>Output</i>	: Keluaran
<i>Pivot</i>	: Ringkasan sebuah data

<i>Platform</i>	: Tempat atau wadah untuk menjalankan perangkat lunak
<i>Powerfull</i>	: Sangat kuat
<i>Programmer</i>	: Sebutan profesi bagi seorang pemrogram komputer
<i>Read</i>	: Baca
<i>Rekursif</i>	: Proses pemanggilan dirinya sendiri
<i>Searching</i>	: Pencarian
<i>Sintaksis</i>	: Peraturan dalam membuat sebuah kalimat meliputi tata bahasa
<i>Sorting</i>	: Pengurutan
<i>Source Code</i>	: Kode sumber yang berisi uatu rangkaian perintah yang ditulis menggunakan bahasa pemrograman
<i>Successfull</i>	: Berhasil
<i>Unsuccessfull</i>	: Gagal
<i>Update</i>	: Memperbaharui
<i>User</i>	: Pengguna
<i>Variable</i>	: Penamaan item di dalam sebuah data

DAFTAR ISI

KATA PENGANTAR	ii
TAKARIR.....	iii
DAFTAR ISI.....	v
DAFTAR TABEL.....	vii
DAFTAR GAMBAR	viii
BAB I PENDAHULUAN.....	1
1.1 Deskripsi Masalah	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah	2
1.4 Tujuan Pembuatan	3
BAB II TINJAUAN PUSTAKA	4
2.1 Algoritma	4
2.2 Flowchart	4
2.3 Pseudocode	4
2.4 Array	4
2.5 List	5
2.6 Python	5
2.7 Prosedur atau Fungsi.....	5
2.8 Algoritma Searching	5
2.9 Algoritma Sorting	6
2.10 Metode Quick Sort.....	6
2.11 Metode Linear Search.....	6
BAB III METODE PENELITIAN	7
3.1 Tahapan Pembuatan Program	7
3.2 Pengumpulan Data.....	7
3.3 Perancangan Algoritma Sorting dan Searching	7
3.5 Perancangan Program	28
3.5.1 Main Menu.....	28
3.5.2 Manage Menu	28

3.5.3	Sort Menu.....	30
3.5.4	Search Menu	30
BAB IV HASIL DAN PEMBAHASAN		32
4.1	Ilustrasi Sorting.....	32
4.2	Ilustrasi Searching	35
4.3	Tampilan Program	36
4.3.1	Tampilan Awal Program.....	36
4.3.2	Tampilan Manage Menu	37
4.3.3	Tampilan Sort Menu	39
4.3.4	Tampilan Search Menu	41
4.3.5	Tampilan Error Handling.....	43
BAB V KESIMPULAN DAN SARAN		44
5.1	Kesimpulan	44
5.2	Saran	44
DAFTAR PUSTAKA		45

DAFTAR TABEL

Table 3.3.1 Pseudocode Algoritma Sorting dan Searching	21
Table 3.2.2 Source Code Algoritma Sorting dan Searching	28

DAFTAR GAMBAR

Gambar 3.3.1 Flowchart Algoritma Sorting dan Searching	10
Gambar 4.2.1 Ilustrasi Metode Quick Sort Angka (1).....	32
Gambar 4.1.3 Ilustrasi Metode Quick Sort Angka (2).....	32
Gambar 4.1.4 Ilustrasi Metode Quick Sort Angka (3).....	32
Gambar 4.1.5 Ilustrasi Metode Quick Sort Angka (4).....	33
Gambar 4.1.6 Ilustrasi Metode Quick Sort Angka (5).....	33
Gambar 4.1.7 Ilustrasi Metode Quick Sort Angka (6).....	33
Gambar 4.1.8 Ilustrasi Metode Quick Sort Angka (7).....	33
Gambar 4.1.9 Ilustrasi Metode Quick Sort Kata (1).....	34
Gambar 4.1.10 Ilustrasi Metode Quick Sort Kata (2).....	34
Gambar 4.1.11 Ilustrasi Metode Quick Sort Kata (3).....	34
Gambar 4.1.12 Ilustrasi Metode Quick Sort Kata (4).....	34
Gambar 4.1.13 Ilustrasi Metode Quick Sort Kata (5).....	35
Gambar 4.2.1 Ilustrasi Metode Linear Search (1).....	34
Gambar 4.2.2 Ilustrasi Metode Linear Search (2).....	35
Gambar 4.2.3 Ilustrasi Metode Linear Search (3).....	35
Gambar 4.2.4 Ilustrasi Metode Linear Search (4).....	35
Gambar 4.3.1.1 Tampilan Awal Program	36
Gambar 4.3.1.2 Tampilan Awal Program Menu Exit.....	36
Gambar 4.3.2.1 Tampilan Manage Menu (Lihat Data)	37
Gambar 4.3.2.2 Tampilan Manage Menu (Ubah Data)	37
Gambar 4.3.2.3 Tampilan Manage Menu (Tambah Data).....	37
Gambar 4.3.2.4 Tampilan Manage Menu (Hapus Data).....	38
Gambar 4.3.3.1 Tampilan Sort Menu (Nama)	39
Gambar 4.3.3.2 Tampilan Sort Menu (Usia)	39
Gambar 4.3.3.3 Tampilan Sort Menu (NIK).....	40
Gambar 4.3.3.4 Tampilan Sort Menu (Tanggal).....	40
Gambar 4.3.4.1 Tampilan Search Menu (Nama).....	41
Gambar 4.3.4.2 Tampilan Search Menu (Usia)	41

Gambar 4.3.4.3 Tampilan Search Menu (NIK)	42
Gambar 4.3.4.4 Tampilan Search Menu (Tanggal)	42

BAB I PENDAHULUAN

1.1 Deskripsi Masalah

Sebagaimana yang kita ketahui bahwasannya rumah sakit merupakan sebuah institusi atau lembaga yang bergerak di bidang kesehatan masyarakat yang mana dapat dikatakan lembaga ini menyediakan berbagai pelayanan kesehatan perorangan secara lengkap diantaranya pelayanan rawat inap, rawat jalan, gawat darurat, serta konsultasi bersama dokter di bidangnya atau biasa disebut dengan praktik dokter. Hal ini didukung dengan tujuan dari rumah sakit itu sendiri yakni untuk mempermudah akses masyarakat dalam mendapatkan pelayanan kesehatan secara lengkap, serta memberikan sebuah perlindungan terutama dalam hal keselamatan pasiennya.

Rekam medis merupakan sebuah bukti tertulis mengenai proses pelayanan yang diberikan kepada pasien oleh dokter dan tenaga kesehatan lainnya. Dengan adanya bukti tertulis tersebut, maka data rekam medis yang diberikan dapat dipertanggungjawabkan dengan tujuan sebagai penunjang tertib administrasi dalam upaya peningkatan pelayanan kesehatan rekam medis, yang mana setiap harinya rekam medis pada rumah sakit selalu mengalami penambahan data.

Namun, di era pandemi covid-19 seperti sekarang ini yang mana mengharuskan kita semua untuk selalu menjaga jarak satu sama lain dan berusaha untuk tidak menciptakan sebuah keramaian membuat rumah sakit sedikit kesulitan dalam beroperasi secara maksimal untuk menyediakan pelayanan kesehatan bagi masyarakat luas, sehingga hampir semua kegiatan di rumah sakit khususnya pada bagian konsultasi dokter dialihkan secara *online* guna memutus penyebaran rantai covid-19 ini. Hal ini didukung dengan kemajuan teknologi dan informasi yang belakangan ini berkembang dengan cukup pesat dan signifikan khususnya dalam bidang teknologi komputer, yang mana untuk saat ini komputer memegang peranan penting dalam mempermudah penyelesaian suatu pekerjaan, meningkatkan efisiensi kerja, kreatifitas, dan aktivitas.

Berdasarkan deskripsi masalah tersebut, maka penulis menyusun sebuah proposal yang berjudul “Penerapan Metode Quick Sort dan Linear Search dalam Pengelompokkan Data Pasien di Rumah Sakit”, khususnya dalam lingkup praktik dokter guna mempermudah pekerjaan seorang *admin* dalam pengurutan dan pencarian data pasien secara cepat dan

tepat, serta mengurangi adanya resiko kemungkinan data pasien untuk hilang secara tidak sengaja. Hal ini didukung juga dengan perkembangan kebutuhan data dan informasi yang lebih baik agar setiap pencapaian dapat dilaksanakan secara optimal. Pelayanan pendataan pemeriksaan pasien di rumah sakit pada saat konsultasi dengan dokter menjadi peranan penting untuk memenuhi kebutuhan tersebut, maka dibentuklah sebuah sistem digital yang dapat menjadikan pelayanan menjadi lebih cepat dan efektif dalam hal pengurutan dan pencarian data pasien atau rekam medis khususnya pada saat pandemi seperti sekarang ini.

Selain itu dilatarbelakangi, dengan pengalaman penulis yang menyaksikan seorang *admin* di beberapa rumah sakit masih menggunakan cara manual dalam hal pengurutan dan pencarian data pasien dengan menggunakan warna yang berbeda di setiap kartu rekam medis untuk membedakan usia dan pengurutan abjad yang disusun secara urut pada kumpulan rak besar yang mana tentu hal ini membuat seorang *admin* kadang terlihat kesulitan dalam proses pencarian data dari seorang pasien. Dalam program ini, penulis ingin memfokuskan permasalahan pada masalah yang lebih khusus, yaitu untuk mencari dan mengurutkan nama pasien berdasarkan abjad, usia pasien, nomor induk kependudukan (NIK), dan tanggal pasien berobat.

1.2 Rumusan Masalah

1. Bagaimana cara melakukan pengurutan data pasien atau rekam medis di suatu rumah sakit dengan menggunakan metode quick sort?
2. Bagaimana cara melakukan pencarian data pasien atau rekam medis di suatu rumah sakit dengan menggunakan metode linear search?

1.3 Batasan Masalah

1. Program ini hanya dapat mengurutkan dan mencari data pasien melalui nama, usia, nomor induk kependudukan (NIK), dan tanggal pasien berobat saja.
2. Program ini hanya akan mengeluarkan output berupa hasil dari pengurutan dan pencarian yang telah dimasukkan sebelumnya oleh *user*.
3. Program ini akan lebih difokuskan dalam pencarian dan pengurutan data pasien saja tanpa melihat isi dari data pasien tersebut, seperti riwayat penyakitnya.

1.4 Tujuan Pembuatan

1. Memberikan kemudahan kepada para *admin* di rumah sakit dalam mencari dan mengurutkan data pasien yang masuk.
2. Mengurangi risiko terjadinya data pasien yang hilang ataupun tertukar dengan data pasien lain.

BAB II

TINJAUAN PUSTAKA

2.1 Algoritma

Algoritma merupakan kumpulan langkah-langkah logis yang disusun secara sistematis dan urut tentang bagaimana jalannya suatu program dari awal sampai akhir atau dapat dikatakan isi dari algoritma ini akan menyatakan suatu tugas dalam menyelesaikan suatu permasalahan yang terjadi. Oleh karena itu, untuk setiap aksi atau tugas yang dijalankan oleh program komputer atau aplikasi dapat dipastikan memiliki algoritma di dalamnya (Windra, 2018).

2.2 Flowchart

Flowchart atau diagram alir merupakan suatu langkah-langkah dalam menyelesaikan permasalahan yang terdiri dari sekumpulan simbol, yang mana setiap simbolnya menggambarkan suatu kegiatan tertentu yang dihubungkan dengan garis atau arah panah dan harus diikuti secara urut oleh pemroses. Bentuk dari setiap simbol di dalam suatu flowchart memiliki aturan penggunaan telah disepakati bersama (Windra, 2018).

2.3 Pseudocode

Pseudocode merupakan langkah-langkah yang digunakan untuk menyelesaikan proses algoritma yang masing-masing dijelaskan dengan algoritma yang lebih kecil dan lebih sederhana dari proses secara keseluruhan, serta kode yang terdapat di dalam *pseudocode* hampir sama dengan instruksi kode program yang sebenarnya sehingga lebih tepat digunakan untuk menggambarkan algoritma yang akan dikomunikasikan kepada *programmer* (Yan, Imam, Mustika, dan Vivin, 2018).

2.4 Array

Array merupakan sebuah struktur data yang menyimpan kumpulan beberapa data atau nilai yang mana setiap nilai di dalamnya memiliki tipe data yang sama dan disimpan ke dalam satu *variable* yang sama (Falentino, 2021). Dalam mengakses nilai yang terdapat di dalam array diperlukan spesifikasi dalam penamaan array dan nomor posisi atau *indeks* setiap elemen yang ingin diakses di dalamnya dan perlu diketahui bahwasannya di dalam

array, *indeks* setiap elemen bukanlah dimulai dari angka 1 melainkan dimulai dari angka 0 (Vivian dan Rismon, 2020).

2.5 List

List merupakan tipe data yang paling serbaguna dalam menyimpan kumpulan data atau nilai yang berisi jenis data campuran dan dapat menampung kumpulan data yang memiliki tipe data yang berbeda di setiap elemennya yang mana dari setiap elemen di dalamnya akan diberi nomor indeks yang berbeda dan dimulai dari angka 0 (Syafrial dan Esi, 2020).

2.6 Python

Python merupakan salah satu bahasa pemrograman tingkat tinggi yang sangat populer untuk saat ini, dikarenakan python dianggap *powerfull* dan mendekati bahasa manusia sehingga dirasa cukup mudah untuk digunakan dan diterapkan ke dalam *pseudocode* (Windra, 2018). Selain itu, python juga memiliki struktur bahasa atau *sintaksis* yang tidak terlalu rumit untuk dipahami atau dapat dikatakan paling sederhana dibandingkan dengan bahasa pemrograman lain. Didukung dengan kemampuannya dalam melakukan eksekusi sejumlah instruksi secara langsung (interpretatif) yang dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai *platform* sistem operasi (Yulita, Dewan, dan Said, 2021).

2.7 Prosedur atau Fungsi

Prosedur atau fungsi merupakan kumpulan perintah dari pernyataan pemrograman yang berbentuk sebuah potongan kode di dalamnya dan dapat digunakan berkali-kali tanpa perlu menyusun ulang setiap perintah di dalam program tersebut untuk mengolah data yang baru, yang mana prosedur atau fungsi ini bertujuan untuk menjalankan tugas khusus yang spesifik (Maulida dan Muhammad, 2021).

2.8 Algoritma Searching

Searching atau pencarian merupakan sebuah proses yang fundamental dalam pemrograman khususnya dalam menjalankan proses pengolahan data guna menemukan suatu nilai tertentu atau memvalidasi (mencocokkan) nilai di dalam sebuah kumpulan nilai yang memiliki tipe data yang sama (Windra, 2018).

Algoritma *searching* diartikan sebagai algoritma yang menerima sebuah argumen kunci dengan langkah-langkah tertentu yang akan mencari rekaman dengan kunci tersebut. Setelah

proses pencarian dijalankan, maka akan diperoleh dua kemungkinan yaitu data yang dicari ditemukan (*successfull*) atau tidak ditemukan (*unsuccessfull*) (Anita, 2019).

2.9 Algoritma Sorting

Sorting atau pengurutan merupakan sebuah proses untuk mengatur sekumpulan data menurut urutan atau susunan tertentu, dapat secara menaik atau *ascending* dan menurun atau *descending*. Proses searching di sini sangat erat kaitannya dengan proses searching atau pencarian sebuah data (Windra, 2018).

Algoritma sorting diartikan sebagai suatu proses untuk menyusun kembali himpunan objek menggunakan aturan tertentu (Anita, 2019). Menurut Microsoft Bookshelf, definisi algoritma pengurutan adalah algoritma untuk meletakkan kumpulan elemen data ke dalam urutan tertentu berdasarkan satu atau beberapa kunci dalam setiap elemen.

2.10 Metode Quick Sort

Metode Quick Sort merupakan metode pengurutan elemen struktur data yang menggunakan pendekatan *rekursif* dengan cara pembagian. Proses pengurutan dilakukan dengan memecah kumpulan data menjadi dua bagian berdasarkan nilai *pivot* yang dipilih, dengan meletakkan data terbesar di kanan dan data terkecil di kiri untuk *ascending* dan sebaliknya untuk *descending* (Ajay, 2012).

2.11 Metode Linear Search

Metode Linear Search merupakan sebuah metode pencarian elemen target yang dimulai dari awal daftar data dan akan dibandingkan dengan setiap elemen dalam daftar data secara berurutan. Metode ini memiliki caranya sendiri dalam meningkatkan pencarian yaitu dengan menghentikan proses pencarian apabila elemen yang diinginkan telah ditemukan (Nell, Daniel, dan Chip, 2006).

Metode ini juga sering digunakan dalam penulisan program assembly language atau bahasa rakitan untuk mencari byte yang diberikan dalam array, yang mana mengharuskan metode ini untuk dapat menampilkan bidang alamat, elemen pencarian, dan posisi dimana elemen tersebut ditemukan. Jika elemen pencarian tidak ditemukan, maka posisi elemen tersebut akan ditunjukkan sebagai 00 (Udaya, 2008).

BAB III

METODE PENELITIAN

3.1 Tahapan Pembuatan Program

Adapun tahapan yang harus dilakukan dalam pembuatan program pencarian dan pengurutan data pasien antara lain :

1. Mengumpulkan data pasien yang diperlukan seperti nama, usia, nomor induk kependudukan (NIK), dan kapan tanggal pasien tersebut melakukan pengobatan
2. Menentukan metode *sorting* dan *searching*
3. Pembuatan flowchart
4. Pembuatan variabel untuk mengelompokkan data-data pasien yang telah diinput *user*
5. Pembuatan fungsi untuk menampilkan, menghapus, memperbaiki, dan menambah data
6. Pembuatan fungsi untuk mengurutkan dan mencari data
7. Pembuatan menu *user interface* untuk *user*
8. Menjalankan semua fungsi

3.2 Pengumpulan Data

1. Data diperoleh dari input pengguna (petugas rumah sakit), yang berupa :
 - Nama pasien
 - Usia pasien
 - Nomor Induk Kependudukan (NIK) pasien
 - Tanggal pasien melakukan pengobatan
2. Pengumpulan data berlokasi di rumah sakit

3.3 Perancangan Algoritma Sorting dan Searching

- **Algoritma Metode Quick Sort**

Terdapat 11 tahap yang harus dijalankan terlebih dahulu oleh program dalam mengurutkan data pasien dengan menggunakan metode quick sort yaitu, sebagai berikut :

1. Program akan mulai berjalan
2. *User* akan diarahkan untuk memasukkan angka 2 agar dapat melakukan proses *sorting* (pengurutan)
3. Lalu *user* akan diminta untuk memasukkan angka kembali sesuai dengan proses pengurutan yang diinginkan

4. Jika *user* menginputkan angka 1, maka program akan mulai mengurutkan seluruh data berdasarkan nama pasien dan akan langsung lanjut ke tahap 9
5. Jika *user* menginputkan angka 2, maka program akan mulai mengurutkan seluruh data berdasarkan usia pasien dan akan langsung lanjut ke tahap 9
6. Jika *user* menginputkan angka 3, maka program akan mulai mengurutkan seluruh data berdasarkan NIK pasien dan akan langsung lanjut ke tahap 9
7. Jika *user* menginputkan angka 4, maka program akan mulai mengurutkan seluruh data pasien berdasarkan tanggal pasien tersebut berobat ke rumah sakit dan akan langsung lanjut ke tahap 9
8. Jika *user* menginputkan angka 5, maka program akan kembali ke tampilan awal tanpa mengurutkan apapun dan akan langsung lanjut ke tahap 10
9. Lalu program akan meminta *user* untuk menekan tombol *enter* agar dapat kembali ke menu *sorting* atau pengurutan dan akan kembali ke tahap 3
10. Program akan berhenti melakukan pengurutan
11. Program akan kembali ke tampilan awal

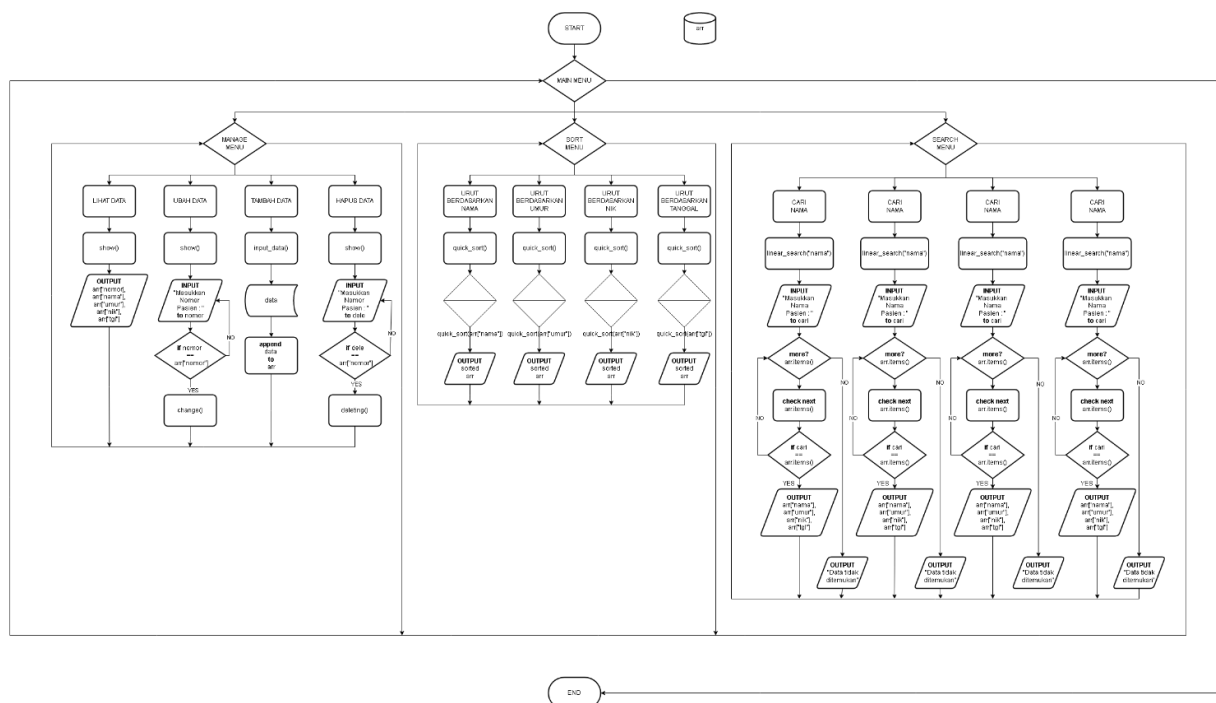
- **Algoritma Metode Linear Search**

Terdapat 16 tahap yang harus dijalankan terlebih dahulu oleh program dalam mencari data pasien dengan menggunakan metode linear search, yaitu sebagai berikut :

1. Program akan mulai berjalan
2. *User* akan diarahkan untuk memasukkan angka 3 untuk melakukan proses *searching* (pencarian)
3. Lalu *user* akan diminta untuk memasukkan angka kembali sesuai dengan proses pencarian yang diinginkan
4. Jika *user* menginputkan angka 1, maka program akan meminta *user* untuk memasukkan nama dari pasien yang ingin dilihat rincian datanya
5. Lalu program akan mengeluarkan output berupa data yang mencakup nama, usia, NIK, dan tanggal pasien berobat berdasarkan nama pasien yang diinputkan tadi dan lanjut ke tahap 13
6. Jika *user* menginputkan angka 2, maka program akan meminta *user* untuk memasukkan usia dari pasien yang ingin dilihat rincian datanya

7. Lalu program akan mengeluarkan output berupa data yang mencakup nama, usia, NIK, dan tanggal pasien berobat berdasarkan usia pasien yang diinputkan tadi dan lanjut ke tahap 13
8. Jika *user* menginputkan angka 3, maka program akan meminta *user* untuk memasukkan NIK dari pasien yang ingin dilihat rincian datanya
9. Lalu program akan mengeluarkan output berupa data yang mencakup nama, usia, NIK, dan tanggal pasien berobat berdasarkan NIK pasien yang diinputkan tadi dan lanjut ke tahap 13
10. Jika *user* menginputkan angka 4, maka program akan meminta *user* untuk memasukkan tanggal pasien tersebut berobat dengan urutan (dd-mm-yyyy)
11. Lalu program akan mengeluarkan output berupa data yang mencakup nama, usia, NIK, dan tanggal pasien berobat berdasarkan NIK pasien yang diinputkan tadi dan lanjut ke tahap 13
12. Jika *user* menginputkan angka 5, maka program akan kembali ke tampilan awal tanpa mengurutkan apapun dan akan langsung lanjut ke tahap 14
13. Lalu program akan meminta *user* untuk menekan tombol *enter* agar dapat kembali ke menu *searching* atau pencarian dan akan kembali ke tahap 3
14. Program akan berhenti melakukan pengurutan
15. Program akan kembali ke tampilan awal

Berikut disajikan juga bentuk algoritma *sorting* dan *searching* dalam bentuk *flowchart* atau diagram alir :



Gambar 3.3.1 Flowchart Algoritma Sorting dan Searching

Berikut disajikan juga bentuk algoritma *sorting* dan *searching* dalam bentuk *pseudo code* :

Program :

Pengelompokkan_Data_Pasien_di_Rumah_Sakit

Deklarasi :

nama, umur, nik, tgl : string

IMPORT sys

IMPORT os

DEFINE FUNCTION clear():

os.system("cls" IF os.name=="nt" else "clear")

SET arr TO [

{ "nama" : "Saban", "umur" : "25", "nik" : "6472040210970005",
"tgl" : "24-05-2020"},

{ "nama" : "Liu", "umur" : "18", "nik" : "6309061905030006",

```

    "tgl" : "23-06-2021"},
    {"nama" : "Alfi", "umur" : "12", "nik" : "6489010202100002",
    "tgl" : "23-01-2022"}
]

```

```

DEFINE FUNCTION show():

```

```

    FOR y, x IN enumerate(arr):
        x.update({"nomor" : int(y)+1})
        OUTPUT("Nomor Pasien      : ",x["nomor"])
        output(x)
        OUTPUT("")
    INPUT("Tekan Enter untuk lanjut")
    OUTPUT("")

```

```

DEFINE FUNCTION change():

```

```

    show()
    WHILE True:
        TRY:
            SET nomor TO int(INPUT("Masukkan Nomor Pasien      :
"))
            break
        EXCEPT:
            OUTPUT("Input anda salah !!!")
    clear()

    FOR data IN arr:
        IF data["nomor"] EQUALS nomor:
            WHILE True:

```

```

        TRY:
            SET nama TO str(INPUT("Masukkan Nama
: "))
            SET umur TO str(INPUT("Masukkan Umur
: "))
            SET nik TO str(INPUT("Masukkan NIK
: "))
            SET tgl TO str(INPUT("Tanggal Masuk RS
(dd-mm-yyyy) : "))

            SET data["nama"] TO nama
            SET data["umur"] TO umur
            SET data["nik"] TO nik
            SET data["tgk"] TO tgl
            break
        EXCEPT:
            OUTPUT("Input anda salah !!!")

DEFINE FUNCTION INPUT_data():
    SET data TO {}
    WHILE True:
        TRY:
            SET data["nama"] TO str(INPUT("Masukkan Nama
: "))
            SET data["umur"] TO str(INPUT("Masukkan Umur
: "))
            SET data["nik"] TO str(INPUT("Masukkan NIK
: "))
            SET data["tgl"] TO str(INPUT("Tanggal Masuk RS
(dd-mm-yyyy) : "))

```

```

        arr.append(data)
        OUTPUT("")
        OUTPUT("Data berhasil ditambahkan")
        break
    EXCEPT:
        OUTPUT("Masukkan Input yang benar")
    clear()

DEFINE FUNCTION deleting():
    show()
    WHILE True:
        TRY:
            SET dele TO int(INPUT("Data Nomor Pasien berapa
yang ingin dihapus : "))
            break
        EXCEPT:
            OUTPUT("Input anda salah !!!")
    clear()

    FOR data IN arr:
        IF data["nomor"] EQUALS dele:
            OUTPUT("Nomor Pasien      : ", data["nomor"])
            output(data)
            SET confirm TO str(INPUT("Anda yakin ingin
menghapus data ini (y/n) : "))

            IF confirm EQUALS "y":
                FOR index IN range(len(arr)):
                    IF arr[index]["nomor"] EQUALS dele:
                        del arr[index]

```

```

                                break
                                OUTPUT("Data telah dihapus")
                                clear()

DEFINE FUNCTION output(z):
    OUTPUT("Nama           : ", z["nama"])
    OUTPUT("Umur           : ", z["umur"])
    OUTPUT("NIK            : ", z["nik"])
    OUTPUT("Tanggal Masuk RS : ", z["tgl"])

DEFINE FUNCTION manage_menu():
    clear()
    OUTPUT("""
    =====MANAGEMENT MENU=====

    1. Lihat data
    2. Ubah data
    3. Tambah data
    4. Hapus data
    5. Kembali""")

    WHILE True:
        TRY:
            OUTPUT("")
            SET pilihan TO int(INPUT("Pilih : "))
            break
        EXCEPT:
            OUTPUT("Input sesuai dengan menu !!!")

    IF pilihan EQUALS 1:
        show()

```

```

        manage_menu()
    ELSEIF pilihan EQUALS 2:
        change()
        manage_menu()
    ELSEIF pilihan EQUALS 3:
        INPUT_data()
        manage_menu()
    ELSEIF pilihan EQUALS 4:
        deleting()
        manage_menu()
    ELSEIF pilihan EQUALS 5:
        main_menu()
clear()

```

```

DEFINE FUNCTION partition(array, low, high, value):
    SET pivot TO array[low]
    SET x TO low + 1
    SET y TO high
    WHILE True:
        WHILE x <= y and array[x][value] <= pivot[value]:
            SET x TO x + 1
        WHILE x <= y and array[y][value] >= pivot[value]:
            SET y TO y - 1
        IF x <= y:
            SET array[x], array[y] TO array[y], array[x]
        ELSE:
            break

    SET array[low], array[y] TO array[y], array[low]
    RETURN y

```



```

DEFINE FUNCTION quick_sort(array, low, high, value):
    IF low >= high:
        RETURN
    SET p TO partition(array, low, high, value)
    quick_sort(array, low, p - 1, value)
    quick_sort(array, p + 1, high, value)

DEFINE FUNCTION sort_menu():
    clear()
    OUTPUT("""
    =====SORT MENU=====
    1. Urutkan berdasarkan Nama
    2. Urutkan berdasarkan Umur
    3. Urutkan berdasarkan NIK
    4. Urutkan berdasarkan Tanggal Masuk RS
    5. Kembali""")

    WHILE True:
        TRY:
            OUTPUT("")
            SET pilih TO int(INPUT("Pilih : "))
            break
        EXCEPT:
            OUTPUT("Input sesuai dengan menu !!!")
    IF pilih EQUALS 1:
        quick_sort(arr, 0, len(arr)-1, "nama")

    FOR x IN arr:
        output(x)
    OUTPUT("")

```

```

        INPUT("Tekan Enter untuk lanjut")
        sort_menu()
    ELSEIF pilih EQUALS 2:
        quick_sort(arr, 0, len(arr)-1, "umur")
        FOR x IN arr:
            output(x)
            OUTPUT("")
        INPUT("Enter Untuk melanjutkan")
        sort_menu()
    ELSEIF pilih EQUALS 3:
        quick_sort(arr, 0, len(arr)-1, "nik")
        FOR x IN arr:
            output(x)
            OUTPUT("")
        INPUT("Enter Untuk melanjutkan")
        sort_menu()
    ELSEIF pilih EQUALS 4:
        quick_sort(arr, 0, len(arr)-1, "tgl")
        FOR x IN arr:
            output(x)
            OUTPUT("")
        INPUT("Tekan Enter untuk lanjut")
        sort_menu()
    ELSEIF pilih EQUALS 5:
        clear()
        main_menu()

DEFINE FUNCTION lin_search(arr, find):
    SET founded TO []
    FOR element IN arr:

```

```

        FOR nilai IN element.items():
            IF find.lower() IN str(nilai).lower():
                founded.append(element)
        RETURN founded

DEFINE FUNCTION search_menu():
    clear()
    OUTPUT("""
    =====SEARCH MENU=====
    1. Cari Nama
    2. Cari Umur
    3. Cari NIK
    4. Cari Tanggal Masuk RS
    5. Kembali""")

    WHILE True:
        TRY:
            OUTPUT("")
            SET pilih TO int(INPUT("Pilih : "))
            break
        EXCEPT:
            OUTPUT("Input sesuai dengan menu !!!")
    IF pilih EQUALS 1:
        OUTPUT("")
        SET carikan TO str(INPUT("Masukkan Nama yang dicari :
"))

        OUTPUT("")
        SET find TO lin_search(arr, carikan)
        IF find != []:
            FOR x IN find:

```

```

        output(x)
        OUTPUT("")
        break
    ELSE:
        OUTPUT("Data tidak ditemukan")
        INPUT("Tekan Enter untuk lanjut")
        search_menu()
    ELSEIF pilih EQUALS 2:
        OUTPUT("")
        SET carikan TO str(INPUT("Masukkan Umur yang dicari :
"))
        OUTPUT("")
        SET find TO lin_search(arr, carikan)
        IF find != []:
            FOR x IN find:
                output(x)
                OUTPUT("")
        ELSE:
            OUTPUT("Data tidak ditemukan")
            INPUT("Tekan Enter untuk lanjut")
            search_menu()
    ELSEIF pilih EQUALS 3:
        OUTPUT("")
        SET carikan TO str(INPUT("Masukkan NIK yang dicari :
"))
        OUTPUT("")
        SET find TO lin_search(arr, carikan)
        IF find != []:
            FOR x IN find:
                output(x)

```

```

        OUTPUT("")

    ELSE:
        OUTPUT("Data tidak ditemukan")
        INPUT("Tekan Enter untuk lanjut")
        search_menu()
ELSEIF pilihh EQUALS 4:
    OUTPUT("")
    SET carikan TO str(INPUT("Tanggal masuk RS (dd-mm-
yyyy) : "))
    OUTPUT("")
    SET find TO lin_search(arr, carikan)
    IF find != []:
        FOR x IN find:
            output(x)
            OUTPUT("")
    ELSE:
        OUTPUT("Data tidak ditemukan")
        INPUT("Tekan Enter untuk lanjut")
        search_menu()
ELSEIF pilihh EQUALS 5:
    clear()
    main_menu()

DEFINE FUNCTION main_menu():
    OUTPUT("=====")
    =====MAIN MENU=====
    1. MANAGE MENU
    2. SORT MENU
    3. SEARCH MENU
    4. exit"")

```

```

SET loop TO True
WHILE loop EQUALS True:
    OUTPUT("")
    SET pilih TO int(INPUT("Pilih : "))
    IF pilih EQUALS 1:
        clear()
        manage_menu()
    ELSEIF pilih EQUALS 2:
        clear()
        sort_menu()
    ELSEIF pilih EQUALS 3:
        clear()
        search_menu()
    ELSEIF pilih EQUALS 4:
        clear()
        SET loop TO False
        sys.exit()
    ELSE:
        OUTPUT("Input sesuai dengan menu !!!")
main_menu()

```

Table 3.3.1 Pseudocode Algoritma Sorting dan Searching

sehingga berdasarkan penulisan *pseudo code* di atas, didapat susunan *source code* pada program menjadi :

```

import sys
import os
def clear():
    os.system("cls" if os.name=="nt" else "clear")

arr = [
    {"nama" : "Saban", "umur" : "25", "nik" : "6472040210970005", "tgl" : "24-05-2020"},

```

```

        {"nama" : "Liu", "umur" : "18", "nik" : "6309061905030006", "tgl" : "23-06-2021"},
        {"nama" : "Alfi", "umur" : "12", "nik" : "6489010202100002", "tgl" : "23-01-2022"}
    ]

def show():
    for y, x in enumerate(arr):
        x.update({"nomor" : int(y)+1})
        print("Nomor Pasien      : ",x["nomor"])
        output(x)
        print("")
    input("Tekan Enter untuk lanjut")
    print("")

def change():
    show()
    while True:
        try:
            nomor = int(input("Masukkan Nomor Pasien      : "))
            break
        except:
            print("Input anda salah !!!")
    clear()
    for data in arr:
        if data["nomor"] == nomor:
            while True:
                try:
                    nama = str(input("Masukkan Nama                               : "))
                    umur = str(input("Masukkan Umur                               : "))
                    nik = str(input("Masukkan NIK                               : "))
                    tgl = str(input("Tanggal Masuk RS (dd-mm-yyyy) : "))
                    data["nama"] = nama
                    data["umur"] = umur
                    data["nik"] = nik
                    data["tgk"] = tgl
                    break
                except:
                    print("Input anda salah !!!")

def input_data():
    data = {}
    while True:
        try:
            data["nama"] = str(input("Masukkan Nama                               : "))

```

```

        data["umur"] = str(input("Masukkan Umur : "))
        data["nik"] = str(input("Masukkan NIK : "))
        data["tgl"] = str(input("Tanggal Masuk RS (dd-mm-yyyy) : "))
        arr.append(data)
        print("")
        print("Data berhasil ditambahkan")
        break
    except:
        print("Masukkan Input yang benar")
clear()

def deleting():
    show()
    while True:
        try:
            dele = int(input("Data Nomor Pasien berapa yang ingin dihapus : "))
            break
        except:
            print("Input anda salah !!!")
    clear()
    for data in arr:
        if data["nomor"] == dele:
            print("Nomor Pasien : ", data["nomor"])
            output(data)
            confirm = str(input("Anda yakin ingin menghapus data ini (y/n) : "))

            if confirm == "y":
                for index in range(len(arr)):
                    if arr[index]["nomor"] == dele:
                        del arr[index]
                        break
                print("Data telah dihapus")
    clear()

def output(z):
    print("Nama : ", z["nama"])
    print("Umur : ", z["umur"])
    print("NIK : ", z["nik"])
    print("Tanggal Masuk RS : ", z["tgl"])

def manage_menu():
    clear()
    print("""
=====MANAGEMENT MENU=====
1. Lihat data

```



```

2. Ubah data
3. Tambah data
4. Hapus data
5. Kembali"""')
while True:
    try:
        print("")
        pilihan = int(input("Pilih : "))
        break
    except:
        print("Input sesuai dengan menu !!!")
if pilihan == 1:
    show()
    manage_menu()
elif pilihan == 2:
    change()
    manage_menu()
elif pilihan == 3:
    input_data()
    manage_menu()
elif pilihan == 4:
    deleting()
    manage_menu()
elif pilihan == 5:
    main_menu()
clear()

def partition(array, low, high, value):
    pivot = array[low]
    x = low + 1
    y = high
    while True:
        while x <= y and array[x][value] <= pivot[value]:
            x = x + 1
        while x <= y and array[y][value] >= pivot[value]:
            y = y - 1
        if x <= y:
            array[x], array[y] = array[y], array[x]
        else:
            break
    array[low], array[y] = array[y], array[low]
    return y

def quick_sort(array, low, high, value):
    if low >= high:
        return

```

```

p = partition(array, low, high, value)
quick_sort(array, low, p - 1, value)
quick_sort(array, p + 1, high, value)

def sort_menu():
    clear()
    print("""
=====SORT MENU=====
1. Urutkan berdasarkan Nama
2. Urutkan berdasarkan Umur
3. Urutkan berdasarkan NIK
4. Urutkan berdasarkan Tanggal Masuk RS
5. Kembali""")
    while True:
        try:
            print("")
            pilih = int(input("Pilih : "))
            break
        except:
            print("Input sesuai dengan menu !!!")
    if pilih == 1:
        quick_sort(arr, 0, len(arr)-1, "nama")
        for x in arr:
            output(x)
            print("")
        input("Tekan Enter untuk lanjut")
        sort_menu()
    elif pilih == 2:
        quick_sort(arr, 0, len(arr)-1, "umur")
        for x in arr:
            output(x)
            print("")
        input("Enter Untuk melanjutkan")
        sort_menu()
    elif pilih == 3:
        quick_sort(arr, 0, len(arr)-1, "nik")
        for x in arr:
            output(x)
            print("")
        input("Enter Untuk melanjutkan")
        sort_menu()
    elif pilih == 4:
        quick_sort(arr, 0, len(arr)-1, "tgl")
        for x in arr:
            output(x)

```

```

        print("")
        input("Tekan Enter untuk lanjut")
        sort_menu()
    elif pilih == 5:
        clear()
        main_menu()

def lin_search(arr, find):
    founded = []
    for element in arr:
        for nilai in element.items():
            if find.lower() in str(nilai).lower():
                founded.append(element)
    return founded

def search_menu():
    clear()
    print("""
    =====SEARCH MENU=====
    1. Cari Nama
    2. Cari Umur
    3. Cari NIK
    4. Cari Tanggal Masuk RS
    5. Kembali""")
    while True:
        try:
            print("")
            pilihh = int(input("Pilih : "))
            break
        except:
            print("Input sesuai dengan menu !!!")
    if pilihh == 1:
        print("")
        carikan = str(input("Masukkan Nama yang dicari : "))
        print("")
        find = lin_search(arr, carikan)
        if find != []:
            for x in find:
                output(x)
                print("")
                break
        else:
            print("Data tidak ditemukan")
            input("Tekan Enter untuk lanjut")
            search_menu()

```

```

elif pilihh == 2:
    print("")
    carikan = str(input("Masukkan Umur yang dicari : "))
    print("")
    find = lin_search(arr, carikan)
    if find != []:
        for x in find:
            output(x)
            print("")
    else:
        print("Data tidak ditemukan")
        input("Tekan Enter untuk lanjut")
        search_menu()
elif pilihh == 3:
    print("")
    carikan = str(input("Masukkan NIK yang dicari : "))
    print("")
    find = lin_search(arr, carikan)
    if find != []:
        for x in find:
            output(x)
            print("")
    else:
        print("Data tidak ditemukan")
        input("Tekan Enter untuk lanjut")
        search_menu()
elif pilihh == 4:
    print("")
    carikan = str(input("Tanggal masuk RS (dd-mm-yyyy) : "))
    print("")
    find = lin_search(arr, carikan)
    if find != []:
        for x in find:
            output(x)
            print("")
    else:
        print("Data tidak ditemukan")
        input("Tekan Enter untuk lanjut")
        search_menu()
elif pilihh == 5:
    clear()
    main_menu()

def main_menu():
    print("")

```

```

=====MAIN MENU=====
1. MANAGE MENU
2. SORT MENU
3. SEARCH MENU
4. exit"")

loop = True
while loop == True:
    print("")
    pilih = int(input("Pilih : "))
    if pilih == 1:
        clear()
        manage_menu()
    elif pilih == 2:
        clear()
        sort_menu()
    elif pilih == 3:
        clear()
        search_menu()
    elif pilih == 4:
        clear()
        loop = False
        sys.exit()
    else:
        print("Input sesuai dengan menu !!!")

main_menu()

```

Table 3.2.2 Source Code Algoritma Sorting dan Searching

3.5 Perancangan Program

3.5.1 Main Menu

1. Program akan mengeluarkan output berupa “*Main Menu*” yang berisi 4 pilihan menu yang tersedia di dalam program yang penulis buat
2. Jika *user* menginputkan “4” pada pilihan “*Main Menu*”, maka program otomatis akan berhenti dan *user* akan keluar dari program tersebut

3.5.2 Manage Menu

3. Jika *user* menginputkan angka “1” pada pilihan “*Main Menu*”, maka *user* akan masuk ke dalam menu “*Manage Menu*”, program yang didesain menggunakan metode CRUD (Create, Read, Update, dan Delete), yang mana program akan

mengeluarkan output pilihan menu yang terdiri dari [1] Lihat Data, [2] Ubah Data, [3] Tambah Data, [4] Hapus Data, dan [0] Kembali.

4. Apabila *user* menginputkan angka “1” yang berarti melihat data, maka program akan mengeluarkan output berupa rincian seluruh data yang tersimpan di dalam program secara acak
5. Apabila *user* menginputkan angka “2” yang berarti mengubah data, maka program akan mengeluarkan output berupa rincian seluruh data yang tersimpan beserta nomor pasiennya
6. Lalu *user* akan diminta untuk memasukkan nomor pasien yang ingin diubah datanya
7. Kemudian program akan meminta *user* untuk memasukkan ulang mengenai rincian data pasien terbaru dari nama, usia, NIK, dan tanggal pasien berobat
8. Setelah itu program akan menyimpan secara otomatis dan *user* akan diarahkan kembali menuju “*Manage Menu*”
9. Apabila *user* menginputkan angka “3” yang berarti menambahkan data, maka program akan meminta *user* untuk memasukkan nama pasien terbaru dilanjutkan dengan usia, NIK, dan tanggal pasien berobat pada saat itu
10. Setelah itu program akan menyimpan secara otomatis dan *user* akan diarahkan kembali menuju “*Manage Menu*”
11. Apabila *user* menginputkan angka “4” yang berarti hapus data, maka program akan mengeluarkan output berupa rincian seluruh data pasien yang tersimpan beserta nomor pasiennya
12. Kemudian *user* akan diminta untuk memasukkan nomor pasien mana yang ingin dihapus datanya
13. Lalu program akan menanyakan kepada *user* “Anda yakin ingin menghapus data ini?”. Jika *user* yakin, maka *user* cukup memasukkan huruf “y” pada program
14. Setelah itu program akan mengeluarkan output berupa “Data telah dihapus” dan *user* akan diarahkan kembali menuju “*Manage Menu*”
15. Namun, jika *user* kurang yakin untuk menghapus data tersebut, maka *user* dapat memasukkan huruf “n” pada program

16. Setelah itu program akan menyimpan secara otomatis dan *user* akan diarahkan kembali menuju “*Manage Menu*”
17. Apabila *user* menginputkan angka “5”, maka program akan langsung kembali ke “*Main Menu*”

3.5.3 Sort Menu

18. Jika *user* menginputkan “2” pada pilihan “*Main Menu*”, maka *user* akan masuk ke dalam menu “Sort Menu”
19. Apabila *user* menginputkan angka “1”, maka program akan mulai mengurutkan seluruh data yang terkumpul berdasarkan nama pasien dan *user* akan diarahkan untuk menekan *enter* agar program dapat berjalan kembali
20. Apabila *user* menginputkan angka “2”, maka program akan mulai mengurutkan seluruh data yang terkumpul berdasarkan usia pasien dan *user* akan diarahkan untuk menekan *enter* agar program dapat berjalan kembali
21. Apabila *user* menginputkan angka “3”, maka program akan mulai mengurutkan seluruh data berdasarkan NIK pasien dan *user* akan diarahkan untuk menekan *enter* agar program dapat berjalan kembali
22. Apabila *user* menginputkan angka “4”, maka program akan mulai mengurutkan seluruh data pasien berdasarkan tanggal pasien tersebut berobat ke rumah sakit dan *user* akan diarahkan untuk menekan *enter* agar program dapat berjalan kembali
23. Apabila *user* menginputkan angka “5”, maka program akan kembali ke “*Main Menu*” tanpa mengurutkan apapun

3.5.4 Search Menu

24. Jika *user* menginputkan “3” pada pilihan “*Main Menu*”, maka *user* akan masuk ke dalam menu “Search Menu”
25. Apabila *user* menginputkan angka “1”, maka program akan meminta *user* untuk memasukkan nama dari pasien yang ingin dilihat rincian datanya
26. Lalu program akan mengeluarkan output berupa rincian data berdasarkan nama pasien yang diinputkan tadi
27. Apabila *user* menginputkan angka “2”, maka program akan meminta *user* untuk memasukkan usia dari pasien yang ingin dilihat rincian datanya

28. Lalu program akan mengeluarkan output berupa rincian data berdasarkan usia pasien yang diinputkan tadi
29. Apabila *user* menginputkan angka “3”, maka program akan meminta *user* untuk memasukkan NIK dari pasien yang ingin dilihat rincian datanya
30. Lalu program akan mengeluarkan output berupa rincian data berdasarkan NIK pasien yang diinputkan tadi
31. Apabila *user* menginputkan angka “4”, maka program akan meminta *user* untuk memasukkan tanggal pasien tersebut berobat dengan format (dd-mm-yyyy)
32. Lalu program akan mengeluarkan output berupa rincian data berdasarkan tanggal pasien tersebut berobat seperti yang diinputkan tadi
33. Apabila *user* menginputkan angka “5”, maka program akan kembali ke “*Main Menu*” tanpa mencari apapun

BAB IV

HASIL DAN PEMBAHASAN

4.1 Ilustrasi Sorting

Berikut penyajian ilustrasi dari metode quick sort dalam melakukan pengurutan data para pasien di rumah sakit :

- Usia dan NIK pasien, serta tanggal pasien berobat

Langkah pertama, susun seluruh data yang terkumpul seperti pada ilustrasi di bawah ini, untuk mempermudah dalam menentukan sisi kanan, kiri, dan juga *pivot* (tengah) dari setiap elemen.

0	1	2	3	4	5	6	7	8	9	Indeks
80	19	70	22	60	50	40	20	75	6	Data

Gambar 4.2.1 Ilustrasi Metode Quick Sort Angka (1)

Langkah kedua, di sini 60 akan berperan sebagai *pivot* dan akan melakukan pertukaran tempat antara sisi ujung kanan (6) dengan sisi ujung kiri (80).

80	19	70	22	60	50	40	20	75	6
6	19	70	22	60	50	40	20	75	80

Gambar 4.1.3 Ilustrasi Metode Quick Sort Angka (2)

Langkah ketiga, 60 tetap berperan sebagai *pivot* dan lakukan pertukaran yang sama seperti pada langkah kedua dengan menempatkan data terkecil di kanan dan data terbesar di kiri.

6	19	70	22	60	50	40	20	75	80
6	19	70	22	60	50	40	20	75	80

Gambar 4.1.4 Ilustrasi Metode Quick Sort Angka (3)

Langkah keempat, ulangi pertukaran yang sama seperti langkah sebelumnya.

6	19	70	22	60	50	40	20	75	80
6	19	20	22	60	50	40	70	75	80

Gambar 4.1.5 Ilustrasi Metode Quick Sort Angka (4)

Langkah kelima, lakukan pertukaran sekali lagi dengan menukar angka 12 dan 40.

6	19	20	22	60	50	40	70	75	80
6	19	20	22	60	50	40	70	75	80

Gambar 4.1.6 Ilustrasi Metode Quick Sort Angka (5)

Langkah keenam, 60 bukan lagi sebagai *pivot* dan kali 50 akan berperan sebagai *pivot*.

Di sini 60 akan bertukar tempat dengan 40.

6	19	20	22	60	50	40	70	75	80
6	19	20	22	40	50	60	70	75	80

Gambar 4.1.7 Ilustrasi Metode Quick Sort Angka (6)

Langkah ketujuh, seluruh data yang terkumpul telah terurut secara ascending yaitu dari data terkecil hingga data terbesar.

0	1	2	3	4	5	6	7	8	9	Indeks
80	19	70	22	60	50	40	20	75	6	Data sebelum sorting
6	19	20	22	40	50	60	70	75	80	Data setelah sorting

Gambar 4.1.8 Ilustrasi Metode Quick Sort Angka (7)

- Nama pasien

Langkah pertama, susun seluruh data yang terkumpul terlebih dahulu untuk mempermudah dalam menentukan sisi kanan, kiri, dan juga *pivot* (tengah) dari setiap elemen

Langkah kedua, di sini Burhan akan berperan sebagai *pivot* dan akan melakukan pertukaran tempat dengan elemen selanjutnya

Burhan	Anto	Zidan	Bagas	Budi	Sita	Ragil	Ghozi
--------	------	-------	-------	------	------	-------	-------

Gambar 4.1.9 Ilustrasi Metode Quick Sort Kata (1)

Langkah ketiga, Anto dan Zidan akan berperan sebagai *pivot* dan melakukan pertukaran yang sama seperti pada langkah kedua dengan menempatkan data terbesar di kiri dan data terkecil di kanan

Anto	Bagas	Budi	Burhan	Zidan	Sita	Ragil	Ghozi
------	-------	------	--------	-------	------	-------	-------

Gambar 4.1.10 Ilustrasi Metode Quick Sort Kata (2)

Langkah keempat, Sita akan berperan sebagai *pivot* dan melakukan pertukaran yang sama seperti pada langkah sebelumnya

Anto	Bagas	Budi	Burhan	Sita	Ragil	Ghozi	Zidan
------	-------	------	--------	------	-------	-------	-------

Gambar 4.1.11 Ilustrasi Metode Quick Sort Kata (3)

Langkah kelima, Ragil akan berperan sebagai *pivot* terakhir dan melakukan pertukaran yang sama seperti pada langkah sebelumnya

Anto	Bagas	Budi	Burhan	Ragil	Ghozi	Sita	Zidan
------	-------	------	--------	-------	-------	------	-------

Gambar 4.1.12 Ilustrasi Metode Quick Sort Kata (4)

Langkah keenam, seluruh data yang terkumpul telah terurut secara ascending yaitu dari data terkecil hingga data terbesar.

Anto	Bagas	Budi	Burhan	Ragil	Ghozi	Sita	Zidan
------	-------	------	--------	-------	-------	------	-------

Gambar 4.1.13 Ilustrasi Metode Quick Sort Kata (5)

4.2 Ilustrasi Searching

Berikut penyajian ilustrasi dari metode linear search dalam melakukan pencarian nama dan usia para pasien di rumah sakit :

- Usia dan NIK pasien, serta tanggal pasien berobat

Langkah pertama, dimisalkan ingin mencari angka 25 pada seluruh data yang telah terkumpul sebagai berikut :


0	1	2	3	4	5	6	7	8	9	Indeks
2	7	6	9	60	50	40	25	7	24	Data

Gambar 4.2.1 Ilustrasi Metode Linear Search (1)

Langkah kedua, program akan terus mencari dari indeks 0 hingga 10 dimanakah letak dari angka 25 tersebut

Langkah ketiga, program akhirnya menemukan bahwa angka 25 terdapat pada indeks ke-7 dari 10 data yang terkumpul. Apabila data telah ditemukan maka program akan langsung berhenti.

0	1	2	3	4	5	6	7	8	9
2	7	6	9	60	50	40	25	7	24



Gambar 4.2.2 Ilustrasi Metode Linear Search (2)

- Nama pasien


Langkah pertama, dimisalkan ingin mencari nama Daffa pada seluruh data yang telah terkumpul sebagai berikut :

0	1	2	3	Indeks
Anto	Cahyo	Daffa	Cahya	Data

Gambar 4.2.3 Ilustrasi Metode Linear Search (3)

Langkah kedua, program akan terus mencari dari indeks 0 hingga 3 dimanakah letak dari nama Andi tersebut

0	1	2	3
Anto	Cahyo	Daffa	Cahya



Gambar 4.2.4 Ilustrasi Metode Linear Search (4)

Langkah ketiga, program akhirnya menemukan bahwa nama Daffa terdapat pada indeks ke-2 dari 3 data yang terkumpul. Apabila data telah ditemukan maka program akan langsung berhenti.

4.3 Tampilan Program

4.3.1 Tampilan Awal Program

Ketika program dijalankan, maka tampilan awal program yang akan keluar adalah seperti gambar di bawah ini yaitu, program akan langsung mengarahkan user ke menu utama atau *main menu*, yang berisi “*Manage Menu*”, “*Sort Menu*”, “*Search Menu*”, dan “*Exit*”. Lalu *user* akan diminta untuk menginputkan pilihan menu yang dipilih sesuai dengan pilihan yang tersedia.

```

=====MAIN MENU=====
1. MANAGE MENU
2. SORT MENU
3. SEARCH MENU
4. exit
Pilih :

```

Gambar 4.3.1.1 Tampilan Awal Program

Jika *user* ingin keluar dari program atau menghentikan program tersebut, maka pada pilihan menu utama *user* cukup menginputkan angka 4 saja dan program akan otomatis berhenti dan *user* sudah pasti langsung keluar dari program, serta mengeluarkan output seperti di bawah ini.

```

=====MAIN MENU=====
1. MANAGE MENU
2. SORT MENU
3. SEARCH MENU
4. exit

Pilih : 4
PS D:\#KULIAH\SEMESTER 2\ALPRO LANJUT\UTS\KODING>

```

Gambar 4.3.1.2 Tampilan Awal Program Menu Exit

4.3.2 Tampilan Manage Menu

Jika *user* menginputkan angka “1” yakni memilih manage menu pada pilihan menu utama, maka program akan mengarahkan *user* menuju program CRUD (*Create*, *Read*, *Update*, dan *Delete*). Selain itu pada manage menu di sini *user* akan diminta untuk menginputkan pilihan menu yang diinginkan kembali.

```

=====MANAGEMENT MENU=====
1. Lihat data
2. Ubah data
3. Tambah data
4. Hapus data
5. Kembali

Pilih : 1
Nomor Pasien : 1
Nama : Saban
Umur : 22
NIK : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Nomor Pasien : 2
Nama : Liu
Umur : 18
NIK : 6309061905030006
Tanggal Masuk RS : 23-06-2021

Nomor Pasien : 3
Nama : Alfi
Umur : 12
NIK : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Tekan Enter untuk lanjut

```

Gambar 4.3.2.1 Tampilan Manage Menu (Lihat Data)

Jika *user* ingin melihat data saja, maka angka 1 yang harus diinputkan ke dalam program. Kemudian program akan mengeluarkan output seperti gambar di atas.

```

Masukkan Nama      : Liu
Masukkan Umur      : 19
Masukkan NIK       : 6472010091000104
Tanggal Masuk RS (dd-mm-yyyy) : 03-04-2022

```

Gambar 4.3.2.2 Tampilan Manage Menu (Ubah Data)

Jika *user* ingin mengubah data pada data yang telah tersimpan sebelumnya, maka angka 2 yang harus diinputkan ke dalam program. Kemudian program akan menampilkan kumpulan data pasien beserta nomor pasien tersebut, lalu *user* akan diminta untuk menginputkan nomor pasien yang mana yang ingin diubah datanya dan program akan mengeluarkan output seperti gambar di atas. Setelah itu program akan otomatis menyimpan data perubahan setelah *user* mengklik tombol *enter*.

```

Pilih : 3
Masukkan Nama      : Mail
Masukkan Umur      : 6
Masukkan NIK       : 6473017845009801
Tanggal Masuk RS (dd-mm-yyyy) : 09-04-2021

```

Gambar 4.3.2.3 Tampilan Manage Menu (Tambah Data)

Jika *user* ingin menambahkan data pasien baru pada program, maka angka 3 yang harus diinputkan ke dalam program. Kemudian program akan menampilkan beberapa data yang harus diisi sebelum data berhasil ditambahkan. Jika *user* telah selesai menginputkan seluruh data pasien baru, maka *user* cukup menhkklik tombol *enter* dan program akan mengeluarkan output “Data berhasil ditambahkan”. Kemudian program akan kembali pada tampilan awal manage menu.

```

Nomor Pasien      : 4
Nama              : Mail
Umur              : 6
NIK               : 6473017845009801
Tanggal Masuk RS  : 09-04-2021
Anda yakin ingin menghapus data ini (y/n) : y

```

Gambar 4.3.2.4 Tampilan Manage Menu (Hapus Data)

Jika *user* ingin menghapus data pasien pada program, maka angka 4 yang harus diinputkan ke dalam program. Kemudian program akan menampilkan kumpulan data yang tersimpan di dalam program dengan nomor pasiennya. *User* cukup menginputkan nomor pasien mana yang ingin dihapus, lalu akan muncul sebuah pertanyaan dalam

program “Anda yakin ingin menghapus data ini”. Jika iya, maka *user* hanya perlu menginputkan huruf “y” dan program akan mengeluarkan output “Data berhasil dihapus”. Namun, jika tidak maka *user* diharuskan untuk menginputkan huruf “n”. Kemudian program akan kembali pada tampilan awal manage menu.

Jika *user* ingin kembali ke main menu atau menu utama, maka *user* cukup menginputkan angka 5 saja ke dalam program dan program akan otomatis membawa *user* kembali ke menu utama.

4.3.3 Tampilan Sort Menu

Jika *user* menginputkan angka “2” yakni memilih sort menu pada pilihan menu utama, maka program akan mengarahkan *user* menuju program sorting atau pengurutan data. Selain itu pada sort menu di sini *user* akan diminta untuk menginputkan pilihan menu yang diinginkan kembali.

```

=====SORT MENU=====
1. Urutkan berdasarkan Nama
2. Urutkan berdasarkan Umur
3. Urutkan berdasarkan NIK
4. Urutkan berdasarkan Tanggal Masuk RS
5. Kembali

Pilih : 1
Nama      : Alfi
Umur      : 12
NIK       : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Nama      : Saban
Umur      : 22
NIK       : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Tekan Enter untuk lanjut

```

Gambar 4.3.3.1 Tampilan Sort Menu (Nama)

Jika *user* ingin mengurutkan data berdasarkan nama pasien, maka *user* perlu menginputkan angka 1, maka program akan mulai mengurutkan seluruh data berdasarkan nama pasien.


```

Pilih : 2
Nama      : Alfi
Umur      : 12
NIK       : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Nama      : Saban
Umur      : 22
NIK       : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Enter Untuk melanjutkan

```

Gambar 4.3.3.2 Tampilan Sort Menu (Usia)

Jika *user* ingin mengurutkan data berdasarkan usia pasien, maka *user* perlu memasukkan angka 2, maka program akan mulai mengurutkan seluruh data berdasarkan usia pasien.

```

Pilih : 3
Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Nama      : Saban
Umur      : 22
NIK       : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Nama      : Alfi
Umur      : 12
NIK       : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Enter Untuk melanjutkan

```

Gambar 4.3.3.3 Tampilan Sort Menu (NIK)

Jika *user* ingin mengurutkan data berdasarkan NIK pasien, maka *user* perlu memasukkan angka 3, maka program akan mulai mengurutkan seluruh data berdasarkan NIK pasien.

```

Pilih : 4
Nama      : Alfi
Umur      : 12
NIK       : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Nama      : Saban
Umur      : 22
NIK       : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Tekan Enter untuk lanjut

```

Gambar 4.3.3.4 Tampilan Sort Menu (Tanggal)

Jika *user* ingin mengurutkan data berdasarkan tanggal pasien berobat, maka *user* perlu menginputkan angka 4, maka program akan mulai mengurutkan seluruh data berdasarkan tanggal pasien berobat.

Jika *user* ingin kembali ke main menu atau menu utama, maka *user* cukup menginputkan angka 5 saja ke dalam program dan program akan otomatis membawa *user* kembali ke menu utama.

4.3.4 Tampilan Search Menu

Jika *user* menginputkan angka “3” yakni memilih search menu pada pilihan menu utama, maka program akan mengarahkan *user* menuju program searching atau pencarian data. Selain itu pada search menu di sini *user* akan diminta untuk menginputkan pilihan menu yang diinginkan kembali.

```

=====SEARCH MENU=====
1. Cari Nama
2. Cari Umur
3. Cari NIK
4. Cari Tanggal Masuk RS
5. Kembali

Pilih : 1

Masukkan Nama yang dicari : Liu

Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Tekan Enter untuk lanjut

```

Gambar 4.3.4.1 Tampilan Search Menu (Nama)

Jika *user* ingin mencari data berdasarkan nama pasien, maka *user* perlu menginputkan angka 1, lalu program akan meminta *user* untuk memasukkan nama pasien yang ingin dicari. Kemudian, program akan mulai mencari nama tersebut dan menampilkan rincian datanya seperti gambar di atas.

```
Pilih : 2

Masukkan Umur yang dicari : 12

Nama      : Alfi
Umur      : 12
NIK       : 6489010202100002
Tanggal Masuk RS : 23-01-2022

Tekan Enter untuk lanjut
```

Gambar 4.3.4.2 Tampilan Search Menu (Usia)

Jika *user* ingin mencari data berdasarkan usia pasien, maka *user* perlu menginputkan angka 2, lalu program akan meminta *user* untuk memasukkan usia pasien yang ingin dicari. Kemudian, program akan mulai mencari usia tersebut dan menampilkan rincian datanya seperti gambar di atas.

```
Pilih : 3

Masukkan NIK yang dicari : 6472040210200005

Nama      : Saban
Umur      : 22
NIK       : 6472040210200005
Tanggal Masuk RS : 24-05-2020

Tekan Enter untuk lanjut
```

Gambar 4.3.4.3 Tampilan Search Menu (NIK)

Jika *user* ingin mencari data berdasarkan NIK pasien, maka *user* perlu menginputkan angka 3, lalu program akan meminta *user* untuk memasukkan NIK pasien yang ingin dicari. Kemudian, program akan mulai mencari NIK tersebut dan menampilkan rincian datanya seperti gambar di atas.

```

Pilih : 4

Tanggal masuk RS (dd-mm-yyyy) : 23-06-2021

Nama      : Liu
Umur      : 19
NIK       : 6472010091000104
Tanggal Masuk RS : 23-06-2021

Tekan Enter untuk lanjut

```

Gambar 4.3.4.4 Tampilan Search Menu (Tanggal)

Jika *user* ingin mencari data berdasarkan tanggal pasien berobat, maka *user* perlu menginputkan angka 4, lalu program akan meminta *user* untuk memasukkan tanggal pasien berobat yang ingin dicari. Kemudian, program akan mulai mencari tanggal berobat pasien tersebut dan menampilkan rincian datanya seperti gambar di atas.

Jika *user* ingin kembali ke main menu atau menu utama, maka *user* cukup menginputkan angka 5 saja ke dalam program dan program akan otomatis membawa *user* kembali ke menu utama.

4.3.5 Tampilan Error Handling

Adapun error handling di dalam program ini berupa `else` dan `try, except ValueError` agar memudahkan user jika salah menginputkan pilihan pada program dan memberikan arahan letak salahnya dimana.

Pilihan menu di dalam program dibuat menjadi tipe data string dan jika user menginputkan string yang salah, maka error handling berupa `else` akan menangani error yang disebabkan oleh inputan user yang salah atau kurang sesuai dengan apa yang telah dituliskan di program.

Sedangkan untuk penginputan indeks di dalam program adalah tipe data integer dan jika user menginputkan integer yang salah, maka error handling berupa `try, except ValueError` akan menangani error yang disebabkan oleh inputan user yang salah atau kurang sesuai dengan apa yang telah dituliskan di program.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dengan selesainya penulisan proposal dengan judul “Penerapan Metode Quick Sort dan Linear Search dalam Pengelompokkan Data Pasien di Rumah Sakit” ini diharapkan dapat meringankan tugas dari tenaga kesehatan khususnya para admin di rumah sakit dalam melakukan pencarian dan pengurutan data pasien yang meningkat setiap harinya, serta membantu mengurangi adanya risiko data pasien yang hilang ataupun tertukar dengan data pasien yang lain. Dengan pembuatan program yang didesain mirip dengan penyusunan kartu rekam medis pasien dan menerapkan penggunaan metode CRUD (Create, Read, Update, Delete) juga diharapkan dapat mendukung kinerja *admin* di suatu rumah sakit dapat menjadi lebih baik lagi dalam mencari dan mengurutkan data pasien yang terus bertambah setiap harinya.

5.2 Saran

Dalam pembuatan program ini, penulis menyadari bahwa program yang dibuat masih jauh dari kata sempurna, seperti susunan kodongan (*source code*) yang masih terbilang kurang rapi, keluaran berupa tampilan yang dihasilkan masih kurang menarik, program yang hanya bisa mencari dan mengurutkan hanya dari nama, usia, NIK, dan tanggal pasien berobat saja, serta menu *update* atau ubah yang tidak bisa menggunakan indeks yang ingin diubah saja untuk diganti, sehingga mengharuskan pengguna untuk menginputkan seluruh datanya kembali.

DAFTAR PUSTAKA

- Dale, N., Joyce, T. D., & Weems, C. (2006). *Object-oriented Data Structures Using Java*. Ontario: Jones and Bartlett Publishers.
- Khaituddin, M., & Muhammad. (2021). *MEMBERANTAS BUTA PROGRAM DENGAN BAHASA PYTHON*. Lampung: UPPM Universitas Malahayati.
- Kumar, A. (2012). *Data Structure for "C" Programming*. New Delhi: FIREWALL MEDIA.
- Kumar, K. U. (2008). *The 8085 Microprocessor : Architecture, Programming, and Interfacing*. New Delhi: Pearson Education India.
- Pane Esi, S. F., & Rahmadani, E. V. (2020). *Big Data : Forecasting Menggunakan Python*. Bandung: Kreatif Industri Nusantara.
- Rangkuti, Y. M., Al Idrus, S. I., & Tarigan, D. D. (2021). *Pengantar Pemrograman Python*. Bandung: CV. MEDIA SAINS INDONESIA.
- RMS, A. S. (2019). *STRUKTUR DATA DAN ALGORITMA DENGAN C++*. Medan: CV. AA. RIZKY.
- Sembiring, F. (2021). *Buku Ajar Dasar Pemrograman (Python)*. Sukabumi: NUSAPUTRA PRESS.
- Siahaan, V., & Sianipar, R. H. (2020). *SIX BOOKS IN ONE : Belajar Pemrograman C/C++/Java/MATLAB/Visual Basic/Visual C#*. Jakarta: SPARTA PUBLISHING.
- Skiena, S. S. (1998). *The Algorithm Design Manual : Text*. New York: Springer Science & Business Media.
- Swastika, W. (2018). *Pengantar Algoritma dan Penerapannya pada Python*. Malang: Ma Chung Press.
- Syaifudin, Y. W., Rozi, I. F., Mentari, M., & Lestari, V. A. (2018). *Dasar Pemrograman*. Malang: UPT Percetakan dan Penerbitan Polinema.