

**IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM Mencari  
JARAK TERPENDEK RUMAH SAKIT DI KOTA MEDAN**

**SKRIPSI**

**SURYA ANDIKA RAMADANI GINTING**

**171401030**



**PROGRAM STUDI S1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM Mencari  
JARAK TERPENDEK RUMAH SAKIT DI KOTA MEDAN**

**SKRIPSI**

**SURYA ANDIKA RAMADANI GINTING**

**171401030**



**PROGRAM STUDI S1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN**

**2024**

## PERSETUJUAN

**Judul** : IMPLEMENTASI ALGORITMA FLOYD  
WARSHALL DALAM MENCARI JARAK  
TERPENDEK RUMAH SAKIT DI KOTA MEDAN

**Kategori** : SKRIPSI

**Nama** : SURYA ANDIKA RAMADANI GINTING

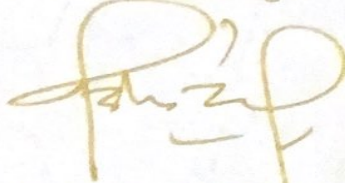
**Nomor Induk Mahasiswa** : 171401030

**Program Studi** : SARJANA (S1) ILMU KOMPUTER

**Fakultas** : ILMU KOMPUTER DAN TEKNOLOGI  
INFORMASI UNIVERSITAS SUMATERA UTARA

**Komisi Pembimbing** :

**Dosen Pembimbing II**



Amer Sharif, S.Si, M.Kom  
NIP. 196910212021011001

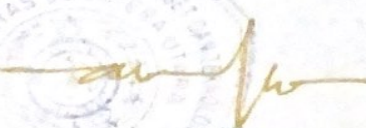
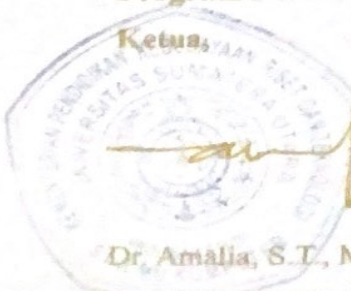
**Dosen Pembimbing I**



Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M  
NIP. 197510082008011011

**Diketahui/Disetujui oleh**  
**Program studi S1 Ilmu Komputer**

**Ketua,**

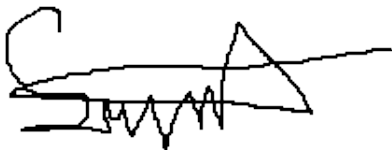
  


Dr. Amalia, S.T., M.T  
NIP. 19781221201404200

**PERNYATAAN****IMPLEMENTASI ALGORITMA FLOYD WARSHALL DALAM MENCARI  
JARAK TERPENDEK RUMAH SAKIT DI KOTA MEDAN****SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 8 Juni 2024



Surya Andika Ramadani Ginting  
171401030

## UCAPAN TERIMA KASIH

Segala puji syukur, hormat, serta kemuliaan bagi Tuhan, Allah Bapa Yang Maha Kuasa, atas berkat dan karunia-Nya lah penulis dapat menyelesaikan Tugas Akhir Mahasiswa S-1 Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara.

Dalam penelitian ini penulis tidak luput dari bantuan dan dukungan berbagai pihak yang terlibat dalam segala proses pengerjaan skripsi ini. Oleh karena itu, penulis secara khusus ingin mengucapkan rasa terima kasih yang sebesar-besarnya kepada:

1. Bapak Dr. Muryanto Amin, S.Sos., M.Si selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia, M.Sc., selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia, S.T., M.T selaku Ketua Program Studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Bapak Dr. M. Andri Budiman, S.T., M.Comp.Sc., M.E.M. selaku dosen pembimbing I yang senantiasa memberikan arahan, bimbingan serta dukungan kepada penulis dalam pengerjaan skripsi ini.
5. Bapak Amer Sharif, S.Si, M.Kom selaku dosen pembimbing II yang senantiasa memberikan nasihat serta bimbingan kepada penulis dalam menyelesaikan skripsi ini.
6. Seluruh Bapak dan Ibu dosen tenaga pengajar dan sivitas akademika Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
7. Ayah, Ibu, dan Kakak Anisa Fitriana yang tiada hentinya memberikan doa dan semangat kepada penulis dalam kehidupan ini..
8. Sahabat seperjuangan penulis dari NBSO yaitu Jeremy Michael Sinaga, Egi Wahyu Rasmamana Dakhi, dan Donisius Martin Sirait yang ganteng.

9. Serta teman-teman Stambuk 2017 S-1 Ilmu Komputer USU terkhusus Kom C sebagai teman sekelas penulis.
10. Seluruh sahabat online penulis siapa dan dimana pun mereka berada, serta semua pihak yang telah mendukung dan membantu penulis yang tidak dapat disebutkan satu-persatu.
11. Teman-teman NBSO yang selalu mendukung dan membimbing saya untuk mengerjakan skripsi ditengah kemalasan saya.

## ABSTRAK

Rumah sakit memainkan peran penting dalam memberikan pelayanan kesehatan kepada masyarakat, terutama di kota-kota besar seperti Medan. Penelitian ini bertujuan untuk mengatasi masalah penting dalam memilih rumah sakit, yaitu menemukan jarak terpendek dari lokasi pengguna ke rumah sakit terdekat, terutama dalam situasi darurat. Algoritma Floyd Warshall dipilih sebagai metode utama dalam penelitian ini untuk mencari jarak terpendek antar rumah sakit di Kota Medan. Implementasi penelitian menggunakan bahasa pemrograman Python dan aplikasi berbentuk desktop. Tujuan utama penelitian ini adalah untuk menyediakan solusi praktis bagi pengguna dalam menemukan rumah sakit terdekat, serta sebagai referensi untuk penelitian selanjutnya dalam bidang yang sama. Melalui analisis, implementasi sistem, dan pengujian, penelitian ini menyimpulkan bahwa algoritma Floyd Warshall berhasil dalam mencari jarak terpendek antar rumah sakit di Kota Medan, dengan hasil yang akurat dan efisiensi waktu yang baik. Selain itu, penelitian ini menunjukkan bahwa algoritma ini dapat diadaptasi untuk berbagai kota atau jaringan lainnya dengan sedikit penyesuaian pada data input, menunjukkan fleksibilitas dan skalabilitas yang luas. Implikasi dari penelitian ini adalah meningkatnya aksesibilitas dan efektivitas pelayanan kesehatan, serta potensi penggunaannya dalam pengelolaan rute dan respons darurat di berbagai konteks.

**Kata Kunci :** Desain dan Analisis Algoritma, Floyd-Warshall, Shortest Path, Python

**IMPLEMENTATION OF THE FLOYD WARSHALL ALGORITHM TO FIND  
THE SHORTEST PATH OF HOSPITAL IN MEDAN CITY**

**ABSTRACT**

*Hospitals play a crucial role in providing healthcare services to the community, especially in large cities like Medan. This research aims to address an important issue in hospital selection, which is finding the shortest distance from the user's location to the nearest hospital, particularly in emergency situations. The Floyd Warshall algorithm was chosen as the primary method in this study to find the shortest distance between hospitals in the city of Medan. The implementation of the research uses the Python programming language and a desktop application. The main objective of this research is to provide a practical solution for users to find the nearest hospital and serve as a reference for further research in the same field. Through analysis, system implementation, and testing, this research concludes that the Floyd Warshall algorithm successfully finds the shortest distance between hospitals in the city of Medan, with accurate results and good time efficiency. Additionally, this research demonstrates that the algorithm can be adapted for various cities or networks with minor adjustments to input data, showing broad flexibility and scalability. The implications of this research are increased accessibility and effectiveness of healthcare services, as well as the potential for use in route management and emergency response in various contexts.*

**Keyword :** *Design and Analysis Algorithm, Floyd-Warshall, Shortest Path, Python.*



## DAFTAR ISI

PERSETUJUAN .....	ii
PERNYATAAN.....	iii
UCAPAN TERIMA KASIH .....	iv
ABSTRAK.....	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
BAB 1 .....	1
PENDAHULUAN .....	1
<b>1.1    Latar Belakang</b> .....	1
<b>1.2    Rumusan Masalah</b> .....	2
<b>1.3    Batasan Masalah</b> .....	2
<b>1.4    Tujuan Penelitian</b> .....	3
<b>1.5    Manfaat Penelitian</b> .....	3
<b>1.6    Sistematika Penulisan</b> .....	3
BAB 2 .....	5
TINJAUAN PUSTAKA.....	5
<b>2.1    Rumah Sakit</b> .....	5
<b>2.2    Jalur Terpendek (<i>shortest path</i>)</b> .....	5
<b>2.3    Graf</b> .....	6
<b>2.4    Algoritma Floyd Warshall</b> .....	7
<b>2.5    Pemrograman Python</b> .....	14
BAB 3 .....	17
ANALISIS DAN PERANCANGAN.....	17
<b>3.1    Rancangan Penelitian</b> .....	17
<b>3.2    Flowchart Sistem</b> .....	18

3.2.1	<i>Flowchart</i> Konversi Koordinat Menjadi Matrix .....	19
3.2.2	Flowchart Algoritma Floyd Warshall.....	21
BAB 4	.....	23
HASIL DAN PEMBAHASAN.....		23
4.1	Implementasi Sistem .....	23
4.1.1	Ruang Lingkup Penerapan .....	23
4.2	Tampilan Program .....	23
4.2.1	Tampilan Perhitungan Jarak Terpendek Rumah Sakit .....	25
4.2.2	Tampilan Peta Rumah Sakit .....	26
4.2.3	Tampilan Hasil Perhitungan Titik Peta Rumah Sakit .....	26
BAB 5	.....	26
KESIMPULAN DAN SARAN.....		26
5.1	Kesimpulan .....	26
5.2	Saran .....	26
Daftar Pustaka.....		28

## BAB 1

### PENDAHULUAN

#### 1.1 Latar Belakang

Rumah sakit merupakan fasilitas kesehatan yang memberikan perawatan medis kepada masyarakat secara menyeluruh termasuk rawat inap, rawat jalan, dan gawat darurat (Putra et al., 2020). Keberadaan rumah sakit sangat penting bagi masyarakat kota Medan. Sebab, Rumah Sakit dapat membantu pengobatan dan merawat orang sakit serta memberikan pelayanan darurat selama 24 jam kepada masyarakat Kota Medan yang menderita sakit atau kecelakaan dan membutuhkan pertolongan segera. Bagi warga kota medan penting untuk mengetahui dimana letak rumah sakit tersebut dengan rute terpendek dari lokasi terdekat pasien . Informasi mengenai jarak antar rumah sakit sangat dibutuhkan disini (Syarifudin & Ramanda, 2021).

Penentuan rute terpendek tersebut dapat diatasi dengan menggunakan teori graf. Graf merupakan rumpun ilmu matematika yang memuat informasi tertentu dengan menggambarkan berbagai macam struktur yang bertujuan untuk membuat visualisasi suatu objek agar lebih mudah dimengerti. Objek graf dapat berupa titik dan sisi yang menghubungkan titik tersebut, dimana bentuk matematis pada graf  $G$  dapat dinyatakan sebagai  $G = (V, E)$ , dimana  $V$  merupakan himpunan titik-titik dan  $E$  merupakan himpunan sisi yang menghubungkan titik-titik. Terdapat berbagai macam algoritma-algoritma yang bisa digunakan untuk menyelesaikan masalah dalam mencari rute terpendek.(Surbakti et al., 2022).

Salah satu algoritma dalam mencari jarak terpendek yaitu algoritma *Floyd warshall*. Algoritma *Floyd warshall* merupakan matriks hubung graf berarah berlabel dan keluarannya adalah rute terpendek dari semua titik ke semua titik.

Untuk mencapai rute terpendek, algoritma ini memulai iterasi dari titik awal dan kemudian memperpanjang rute dengan mengevaluasi setiap titik sehingga bobotnya seminimal mungkin (Hasibuan, 2019).

Pentingnya mengetahui rute terpendek dalam menentukan jarak rumah sakit yang dituju membuat penulis melakukan penerapan algoritma *floyd warshall* dalam melakukan pencarian jarak terpendek pada rumah sakit di kota Medan. Berdasarkan uraian tersebut, penulis memilih judul **“Implementasi Algoritma Floyd Warshall dalam Mencari Jarak Terpendek Rumah Sakit di Kota Medan”**.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang, maka rumusan masalah yang akan diangkat dalam penelitian ini adalah :

1. Bagaimana cara mengumpulkan data lokasi dan jarak antar rumah sakit di Kota Medan yang diperlukan sebagai input untuk algoritma Floyd-Warshall?
2. Apa struktur data yang paling efisien untuk merepresentasikan jaringan rumah sakit dan jarak antar rumah sakit dalam algoritma Floyd-Warshall?
3. Bagaimana proses implementasi algoritma Floyd-Warshall untuk menghitung jarak terpendek antar semua pasangan rumah sakit di Kota Medan?
4. Bagaimana cara mengintegrasikan hasil dari algoritma Floyd-Warshall ke dalam sistem informasi geografis atau aplikasi navigasi untuk memudahkan pengguna dalam mencari rute terpendek ke rumah sakit terdekat?
5. Apa kendala yang mungkin dihadapi dalam proses implementasi algoritma Floyd-Warshall di konteks dunia nyata, khususnya di Kota Medan, dan bagaimana cara mengatasi kendala tersebut?
6. Bagaimana cara mengukur efektivitas dan efisiensi algoritma Floyd-Warshall dalam menentukan jarak terpendek antar rumah sakit dibandingkan dengan metode lainnya?
7. Apa dampak dari penggunaan algoritma Floyd-Warshall terhadap waktu respon darurat dan manajemen rute transportasi medis di Kota Medan?

## 1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut :

1. Penelitian ini merupakan implementasi dalam menemukan jarak terpendek dalam memilih rumah sakit.

2. Membahas bagaimana mencari jarak terdekat dalam memilih rumah sakit dari posisi *user*.
3. Algoritma yang digunakan pada penelitian ini adalah algoritma *Floyd warshall*.
4. Implementasi penelitian ini menggunakan bahasa pemrograman Python.
5. Aplikasi yang akan dibangun berbentuk desktop.

#### **1.4 Tujuan Penelitian**

Tujuan pada penelitian ini adalah untuk mendapatkan jarak terpendek rumah sakit di kota Medan yang dapat membantu pasien dalam memilih rumah sakit mana yang terdekat terutama ketika dalam keadaan darurat.

#### **1.5 Manfaat Penelitian**

Manfaat penelitian ini agar pengguna lebih mudah dalam menentukan jarak Rumah Sakit yang terdekat di sekitarnya, dan juga sebagai referensi pengembangan penelitian pada peneliti selanjutnya.

#### **1.6 Sistematika Penulisan**

Sistematika penulisan yang dilakukan pada penelitian ini yaitu :

##### **1. PENDAHULUAN**

Bab ini membahas permasalahan jarak terdekat Rumah Sakit yang ada di Kota Medan.

##### **2. TINJAUAN PUSTAKA**

Bab ini membahas teori-teori yang berhubungan dengan Graf, Jarak Terpendek, dan Algoritma *Floyd warshall* serta penelitian yang relevan.

##### **3. ANALISIS DAN PERANCANGAN**

Bab ini membahas uraian masalah tentang rancangan penelitian berdasarkan analisis masalah yang digambarkan dalam bentuk

diagram umum, *flowchart*, *use case* diagram pada perancangan sistem.

#### **4. HASIL DAN PEMBAHASAN**

Bab ini membahas hasil implementasi pembuatan sistem menjadi sebuah program serta pembahasan dari pengujian sistem berdasarkan implementasi sistem yang dijalankan apakah sudah sesuai dengan perancangan atau tidak, serta menemukan kesalahan atau kekurangan pada sistem.

#### **5. KESIMPULAN DAN SARAN**

Bab ini membahas kesimpulan dari pembahasan sampai hasil dari pengujian sistem dan juga saran sebagai masukan untuk peneliti selanjutnya yang diharapkan sebagai pemecah masalah.

## BAB 2

### TINJAUAN PUSTAKA

#### 2.1 Rumah Sakit

Rumah sakit adalah fasilitas kesehatan yang memberikan pelayanan kesehatan kepada Masyarakat yang memerlukan diagnosis penyakit, pengobatan, perawatan, dan rehabilitasi. Rumah sakit biasanya memiliki berbagai fasilitas seperti ruang operasi, unit perawatan intensif, ruang masuk, laboratorium, dan peralatan pencitraan medis seperti *rontgen* dan CT scan. Rumah sakit memiliki berbagai jenis staf medis dan paramedis, termasuk dokter, perawat, ahli terapi fisik, dan staf medis, yang bekerja sama untuk memberikan perawatan terbaik kepada pasien (Pratama et al., 2023).

Rumah sakit merupakan tempat pelayanan kesehatan yang memenuhi salah satu kebutuhan masyarakat yaitu pengobatan penyakit tertentu pada pasien. Kehadiran rumah sakit sangat penting karena membantu pengobatan dan perawatan pasien serta menyediakan ruang gawat darurat 24 jam bagi masyarakat yang mengalami kecelakaan ataupun memerlukan penanganan segera. (Puspita et al., 2023).

#### 2.2 Jalur Terpendek (*shortest path*)

Jalur terpendek atau *shortest path* adalah mencari jalan terpendek diantara 2 titik atau simpul dengan menggunakan bobot minimal dari suatu titik asal menuju titik tujuan (Baharudin et al., 2021).

Pencarian lintasan terpendek pada sebuah graf dapat dilakukan dengan perhitungan manual atau melalui komputer, ini akan mengurangi fungsi linier khusus dari lintasan seperti jarak, waktu, dan biaya yang dihadapi selama perjalanan (Melladia, 2020). Dengan mencari lintasan terpendek, seseorang dapat menghemat waktu, tenaga, dan biaya dalam mencari titik tujuan. (Arthalia Wulandari & Sukmasetyan, 2022).

Dengan menggunakan graf, lintasan terpendek dapat ditemukan. Jenis graf yang digunakan adalah graf berbobot, di mana setiap sisi dari graf memiliki nilai

atau bobot. Nilai-nilai ini dapat mencakup hal-hal seperti waktu, biaya, dan sebagainya (Arga et al., 2021).

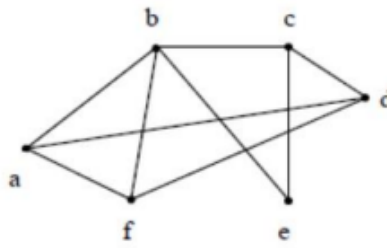
Terdapat beberapa jenis lintasan terpendek yakni :

1. Lintasan terpendek antara dua buah simpul.
2. Lintasan terpendek antara semua pasangan simpul.
3. Lintasan terpendek dari simpul tertentu ke semua simpul yang lain.
4. Lintasan terpendek antara dua buah simpul yang melalui beberapa simpul tertentu.

### 2.3 Graf

Graf merupakan kumpulan objek terstruktur yang pasangan objeknya mempunyai hubungan atau keterkaitan tertentu. Sebuah graf  $G$  didefinisikan sebagai sepasang himpunan  $(V, E)$  dimana  $V$  = himpunan vertex  $\{v_1, v_2, \dots, v_n\}$  dan  $E$  = himpunan sisi (busur) yang menghubungkan simpul  $\{e_1, e_2, \dots, e_n\}$  atau dapat ditulis dengan notasi  $G = (V, E)$  setiap sisi berhubungan dengan satu atau dua simpul. Dua buah simpul dikatakan berhubungan jika ada sisi yang menghubungkan keduanya. Berdasarkan orientasi yang ada pada sisinya, graf dapat dikelompokkan menjadi dua yaitu Graf berarah (direct graf) yaitu graf yang setiap sisinya diberikan arah sehingga untuk dua simpul  $v_i$  dan  $v_j$ , maka  $(v_i, v_j) \neq (v_j, v_i)$  dan graf tak berarah (undirect graf) yaitu graf yang sisinya tidak mengandung arah sehingga untuk dua simpul  $v_i$  dan  $v_j$  maka  $(v_i, v_j) = (v_j, v_i)$ . Selain itu juga dikenal graf berbobot yaitu graf yang sisinya memiliki bobot atau nilai (Wita, 2022).





**Gambar 2.1** Graf

Sebuah graf  $G = (V, E)$  tidak boleh hanya memiliki satu buah busur/edge tanpa satupun titik/vertex yang termuat didalam graf  $G$  tersebut. Suatu busur graf  $G$  dapat dinyatakan dalam bentuk pasangan yang tidak terurut dari sembarang dua buah simpul yang berbeda pada himpunan  $V$  tersebut (Maulani, 2023).

Graf digunakan untuk merepresentasikan objek-objek diskrit dan hubungan objek-objek tersebut. Dalam teori graf, terdapat beberapa algoritma yang dapat digunakan untuk memecahkan permasalahan rute terpendek, antara lain algoritma djikstra, algoritma *Bellman-ford*, algoritma *Greedy*, algoritma *Floyd-warshall* dan lain-lain (Buako et al., 2021).

## 2.4 Algoritma Floyd Warshall

Algoritma *Floyd warshall* merupakan suatu metode yang melakukan pemecahan dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait yang artinya Solusi tersebut dibentuk dari Solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi lebih dari satu. Algoritma *Floyd warshall* merupakan algoritma yang mengambil jarak minimal dari suatu titik ke titik lainnya (Yohana et al., 2023).

Algoritma *Floyd-Warshall* merupakan algoritma untuk melakukan pencarian lintasan terpendek pada suatu graf berbobot (*weighted graph*). Algoritma *Floyd-warshall* dapat menjamin keberhasilan dalam menentukan Solusi minimum dikarenakan algoritma ini dapat membandingkan seluruh kemungkinan lintasan pada graf di setiap sisi dari seluruh simpul yang dilalui (Riksa Herlambang et al., 2021).

Dalam menentukan lintasan terpendek dengan algoritma *Floyd-Warshall* memulai iterasi dari titik awal dan kemudian memperpanjang lintasan dengan

mengevaluasi titik demi titik hingga mencapai titik tujuan dengan jumlah bobot yang paling minimum. Misalkan  $W = W_0$  adalah matriks keterhubungan graf berarah berbobot mula-mula.  $W^*$  adalah matriks keterhubungan minimal dengan  $W_{i,j}$  = lintasan terpendek dari titik  $v_i$  ke  $v_j$ . Algoritma *Floyd-Warshall* untuk mencari lintasan terpendek adalah sebagai berikut :

1.  $W = W_0$
2. Untuk  $k = 1$  hingga  $n$ , lakukan :  
 Untuk  $i = 1$  hingga  $n$ , lakukan :  
 Untuk  $j = 1$  hingga  $n$ , lakukan :  
 Jika  $W_{[i,j]} > W_{i,k} + W_{[k,j]}$  maka tukar  $W_{[i,j]}$  dengan  $W_{[i,k]} + W_{[k,j]}$ .
3.  $W^* = W$ .

Dalam iterasinya untuk mencari lintasan terpendek, algoritma *Floyd-Warshall* membentuk  $n$  matriks, sesuai dengan iterasi- $k$ . Ini akan menyebabkan prosesnya lambat, terutama untuk nilai  $n$  yang besar. Meskipun waktu prosesnya bukanlah yang tercepat, algoritma *Floyd-Warshall* sering digunakan untuk menghitung lintasan terpendek karena kesederhanaannya. Disamping itu, implementasi algoritma *Floyd-Warshall* sangat mudah dibuat.

Matriks keterhubungan  $W$  yang digunakan untuk menyatakan graf berarah berbobot sama dengan matriks yang digunakan untuk menyatakan graf berbobot, yaitu elemen-elemennya menyatakan bobot garis. Secara umum matriks keterhubungan untuk menyatakan graf berarah berbobot tidaklah simetris karena bobot garis dari titik  $v_i$  ke  $v_j = (W_{i,j})$  tidak sama dengan bobot garis dari titik  $v_j$  ke  $v_i = (W_{j,i})$  dan  $W_{i,i} = \infty$  untuk semua  $i$ .

Algoritma *Floyd-Warshall* diatas hanya menghitung jarak terpendek dari semua titik ke semua titik tetapi tidak menjelaskan bagaimana *path* terpendek, maka harus ditambahkan matriks bujur sangkar  $Z$  (ukuran  $n \times n$ ) yang disusun sebagai berikut :

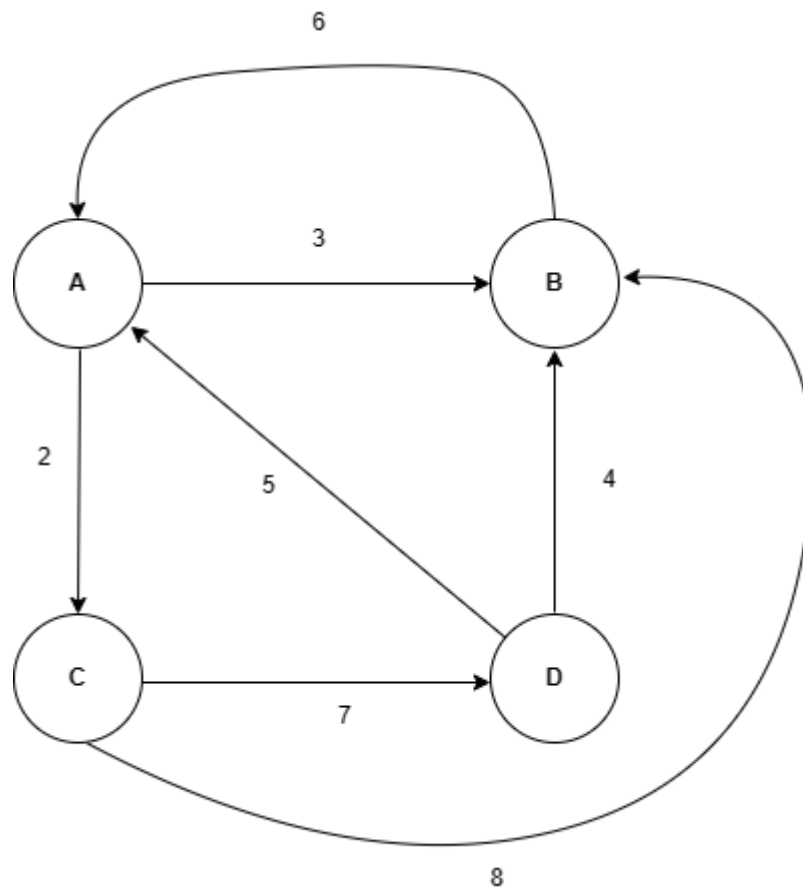
$$\text{Inisialisasi } Z_{i,j}^0 = \begin{cases} j, & \text{jika } W_{i,j}^{(0)} \neq \infty \\ 0, & \text{jika } W_{i,j}^{(0)} = \infty \end{cases}$$

Dalam iterasi ke- $k$ , apabila titik  $v_k$  disisipkan antara titik  $i$  dan titik  $j$  (berarti menukar  $W_{i,j}$  dengan  $W_{i,k} + W_{k,j}$ ), maka ganti  $Z_{i,j}$  dengan  $Z_{i,k}$ . Agar lebih efisien, penggantian matriks  $Z$  dilakukan bersama-sama dengan iterasi pencarian jarak terpendeknya. adalah sebagai berikut :

1.  $W = W_0 ; Z = Z_0$
2. Untuk  $k = 1$  hingga  $n$ , lakukan :
  - Untuk  $i = 1$  hingga  $n$ , lakukan :
  - Untuk  $j = 1$  hingga  $n$ , lakukan :
  - Jika  $W_{[i,j]} > W_{[i,k]} + W_{[k,j]}$  maka
    - a. Tukar  $W_{[i,j]}$  dengan  $W_{[i,k]} + W_{[k,j]}$ .
    - b. Ganti  $Z_{i,j}$  dengan  $Z_{i,k}$
3.  $W^* = W$ .

Berikut contoh penyelesaian pencarian jalur terpendek untuk graf berarah menggunakan algoritma *Floyd-Warshall* :

Tentukan lintasan terpendek dari titik A, B, C, dan D menggunakan algoritma *Floyd-Warshall* untuk graf berarah berbobot pada gambar 2.2 berikut ini!



**Gambar 2.2** Graf Berarah Dan Berbobot

**Penyelesaian :**

Langkah 1 : Menginisialisasi jarak menggunakan simpul graf

**Tabel 2.1** Langkah 1 Inisialisasi Simpul graf

	A	B	C	D
A	0	3	2	
B	6	0	$\infty$	$\infty$
C	$\infty$	8	0	7
D	5	4	$\infty$	0

Pada tabel 2.1 dapat dilihat matriks bobot nilai graf pada gambar 2.2. Dimana pada baris A kolom A nilai bobot adalah 0, kemudian pada kolom A baris B nilai bobot adalah 3, pada kolom A baris C nilai bobot adalah 2, dan pada kolom A baris D nilai

bobot adalah infiniti ( $\infty$ ). Kemudian pada kolom B baris A nilai bobot adalah 6, pada kolom B baris B nilai bobot adalah 0, pada kolom B baris C nilai bobot adalah infiniti ( $\infty$ ), pada kolom B baris D nilai bobot adalah infiniti ( $\infty$ ).

Kemudian pada kolom C baris A nilai bobot adalah infiniti ( $\infty$ ), pada kolom C baris B nilai bobot adalah 8, pada kolom C baris C nilai bobot adalah 0, pada kolom C baris D nilai bobot adalah 7. Kemudian pada kolom D baris A nilai bobot adalah 5, pada kolom D baris B nilai bobot adalah 4, pada kolom D baris C nilai bobot adalah infiniti ( $\infty$ ), pada kolom D baris D nilai adalah 0. Nilai bobot pada matriks tersebut sesuai dengan graf pada gambar 2.2

**Tabel 2.2** Langkah 2 Inisialisasi Simpul Graf melalui A

	A	B	C	D
A	0	3	2	$\infty$
B	6	0	8	15
C	$\infty$	8	0	7
D	5	4	7	0

Pada tabel 2.2 dapat dilihat matriks Langkah ke 2 bobot nilai graf yang melalui simpul A. Dimana pada baris A kolom A nilai bobot adalah 0, kemudian pada kolom A baris B nilai bobot adalah 3, pada kolom A baris C nilai bobot adalah 2, dan pada kolom A baris D nilai bobot adalah infiniti ( $\infty$ ). Kemudian pada kolom B baris A nilai bobot adalah 6, pada kolom B baris B nilai bobot adalah 0, pada kolom B baris C nilai bobot adalah 8, pada kolom B baris D nilai bobot adalah 15.

Kemudian pada kolom C baris A nilai bobot adalah infiniti ( $\infty$ ), pada kolom C baris B nilai bobot adalah 8, pada kolom C baris C nilai bobot adalah 0, pada kolom C baris D nilai bobot adalah 7. Kemudian pada kolom D baris A nilai bobot adalah 5, pada kolom D baris B nilai bobot adalah 4, pada kolom D baris C nilai bobot adalah 7, pada kolom D baris D nilai adalah 0. Nilai bobot tersebut berubah dari matriks tabel 2.1 dikarenakan mencari jalur melalui simpul A, dan akan berubah jika nilai bobot lebih kecil dari sebelumnya.

**Tabel 2.3** Langkah 3 Inisialisasi simpul Graf Melalui B

	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
<b>A</b>	0	3	2	18
<b>B</b>	6	0	8	15
<b>C</b>	14	8	0	7
<b>D</b>	5	4	7	0

Pada tabel 2.3 dapat dilihat matriks Langkah ke 3 bobot nilai graf yang melalui simpul B. Dimana pada baris A kolom A nilai bobot adalah 0, kemudian pada kolom A baris B nilai bobot adalah 3, pada kolom A baris C nilai bobot adalah 2, dan pada kolom A baris D nilai bobot adalah 18. Kemudian pada kolom B baris A nilai bobot adalah 6, pada kolom B baris B nilai bobot adalah 0, pada kolom B baris C nilai bobot adalah 8, pada kolom B baris D nilai bobot adalah 15.

Kemudian pada kolom C baris A nilai bobot adalah 14, pada kolom C baris B nilai bobot adalah 8, pada kolom C baris C nilai bobot adalah 0, pada kolom C baris D nilai bobot adalah 7. Kemudian pada kolom D baris A nilai bobot adalah 5, pada kolom D baris B nilai bobot adalah 4, pada kolom D baris C nilai bobot adalah 7, pada kolom D baris D nilai adalah 0. Nilai bobot tersebut berubah dari matriks tabel 2.2 dikarenakan mencari jalur melalui simpul B, dan akan berubah jika nilai bobot lebih kecil dari sebelumnya.

**Tabel 2.4** Langkah 4 Inisialisasi simpul Graf Melalui C

	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
<b>A</b>	0	3	2	9
<b>B</b>	6	0	8	15
<b>C</b>	14	8	0	7
<b>D</b>	5	4	7	0

Pada tabel 2.4 dapat dilihat matriks Langkah ke 4 bobot nilai graf yang melalui simpul C. Dimana pada baris A kolom A nilai bobot adalah 0, kemudian pada kolom A baris B nilai bobot adalah 3, pada kolom A baris C nilai bobot adalah 2, dan pada kolom A baris D nilai bobot adalah 9. Kemudian pada kolom B baris A nilai bobot adalah 6, pada kolom B baris B nilai bobot adalah 0, pada kolom B baris C nilai bobot adalah 8, pada kolom B baris D nilai bobot adalah 15.

Kemudian pada kolom C baris A nilai bobot adalah 14, pada kolom C baris B nilai bobot adalah 8, pada kolom C baris C nilai bobot adalah 0, pada kolom C baris D nilai bobot adalah 7. Kemudian pada kolom D baris A nilai bobot adalah 5, pada kolom D baris B nilai bobot adalah 4, pada kolom D baris C nilai bobot adalah 7, pada kolom D baris D nilai adalah 0. Nilai bobot tersebut berubah dari matriks tabel 2.3 dikarenakan mencari jalur melalui simpul C, dan akan berubah jika nilai bobot lebih kecil dari sebelumnya.

**Tabel 2.5** Langkah 5 Inisialisasi simpul Graf Melalui D

	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
<b>A</b>	0	3	2	9
<b>B</b>	6	0	8	15
<b>C</b>	12	8	0	7
<b>D</b>	5	4	7	0

Pada tabel 2.5 dapat dilihat matriks Langkah ke 5 bobot nilai graf yang melalui simpul D. Dimana pada baris A kolom A nilai bobot adalah 0, kemudian pada kolom A baris B nilai bobot adalah 3, pada kolom A baris C nilai bobot adalah 2, dan pada kolom A baris D nilai bobot adalah 9. Kemudian pada kolom B baris A nilai bobot adalah 6, pada kolom B baris B nilai bobot adalah 0, pada kolom B baris C nilai bobot adalah 8, pada kolom B baris D nilai bobot adalah 15.

Kemudian pada kolom C baris A nilai bobot adalah 12, pada kolom C baris B nilai bobot adalah 8, pada kolom C baris C nilai bobot adalah 0, pada kolom C baris D nilai bobot adalah 7. Kemudian pada kolom D baris A nilai bobot adalah 5, pada kolom D baris B nilai bobot adalah 4, pada kolom D baris C nilai bobot adalah 7, pada kolom D baris D nilai adalah 0. Nilai bobot tersebut berubah dari matriks tabel 2.4 dikarenakan mencari jalur melalui simpul D, dan akan berubah jika nilai bobot lebih kecil dari sebelumnya.

Hasil bobot nilai Graf tersebut dapat dilihat pada table 2.6 :

**Tabel 2.6** Hasil bobot nilai Graf yang terpendek

	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
<b>A</b>	0	3	2	9
<b>B</b>	6	0	8	15
<b>C</b>	12	8	0	7
<b>D</b>	5	4	7	0

Tabel 2.6 menunjukkan hasil nilai bobot graf yang terdekat dari satu simpul ke simpul lainnya.

## 2.5 Pemrograman Python

Python merupakan bahasa pemrograman yang sangat populer yang dibangun pada tahun 1991 oleh Guido van Rossum. Bahasa pemrograman ini dapat digunakan sebagai pembuatan website, pengembangan perangkat lunak, *scripting system*, dan juga jaringan. Python merupakan Bahasa pemrograman serbaguna yang banyak digunakan untuk memecahkan masalah yang kompleks diberbagai bidang (Jalolov, 2023).

Python merupakan bahasa pemrograman yang bersifat *Open Source*, sehingga dapat dikembangkan secara gratis tanpa lisensi dan sesuai kebutuhan. Python bahasa pemrograman yang sangat fleksibel dan mudah dipelajari. Selain itu, struktur data pada Python sangat efektif, sehingga pemrograman berorientasi objek



yang lebih sederhana tetapi efektif, dapat bekerja pada berbagai platform, dan dapat digabungkan dengan bahasa pemrograman lain untuk membuat aplikasi.

Menurut Angelina M. T. I. Sambu et al., (2023) Python memiliki fitur-fitur yakni :

### **1. *Simple***

Python adalah bahasa pemrograman yang sederhana dan minimalis. Karena itu, mudah dipahami dan digunakan, pengembang aplikasi dapat berkonsentrasi pada penyelesaian masalah.

### **2. *Mudah dipahami***

Python memiliki sintaks yang sangat mudah dipahami dibandingkan dengan bahasa pemrograman lainnya.

### **3. *Gratis dan open source***

Python dapat digunakan dengan mudah untuk membuat program baru, membaca sumber kode, memodifikasi, dan menggunakan komponennya, dan mendistribusikan salinan perangkat lunak dengan mudah.

### **4. *Bahasa pemrograman Tingkat tinggi***

Python merupakan Bahasa pemrograman tingkat tinggi yang mensederhanakan pemrograman komputer yang kompleks.

### **5. *Portable***

Python telah dimodifikasi untuk dapat bekerja pada berbagai platform karena sifatnya *Open-Source*. Semua program Python dapat bekerja pada salah satu platform tanpa mengalami modifikasi apa pun.

## **6. Interpreter**

Python mengubah kode sumber menjadi *bytecodes* sebagai bentuk perantara, dan kemudian menerjemahkannya ke dalam bahasa komputer. Ini membuat prosesnya jauh lebih mudah karena tidak diperlukan untuk mengkompilasi program.

## **7. Object-oriented**

Python mendukung pemrograman berorientasi prosedur serta pemrograman berorientasi objek. Dalam bahasa *procedure-oriented*, program dibangun di sekitar prosedur atau fungsi yang tidak lain adalah potongan program yang dapat digunakan kembali. Dalam bahasa *object-oriented*, program ini dibangun di sekitar objek yang menggabungkan data dan fungsionalitas.

## **8. Extensible**

Jika sebuah program membutuhkan beberapa kode penting atau sebagian algoritma, Python dapat membaca bagian dari program C atau C++.

## **9. Extensive libraries**

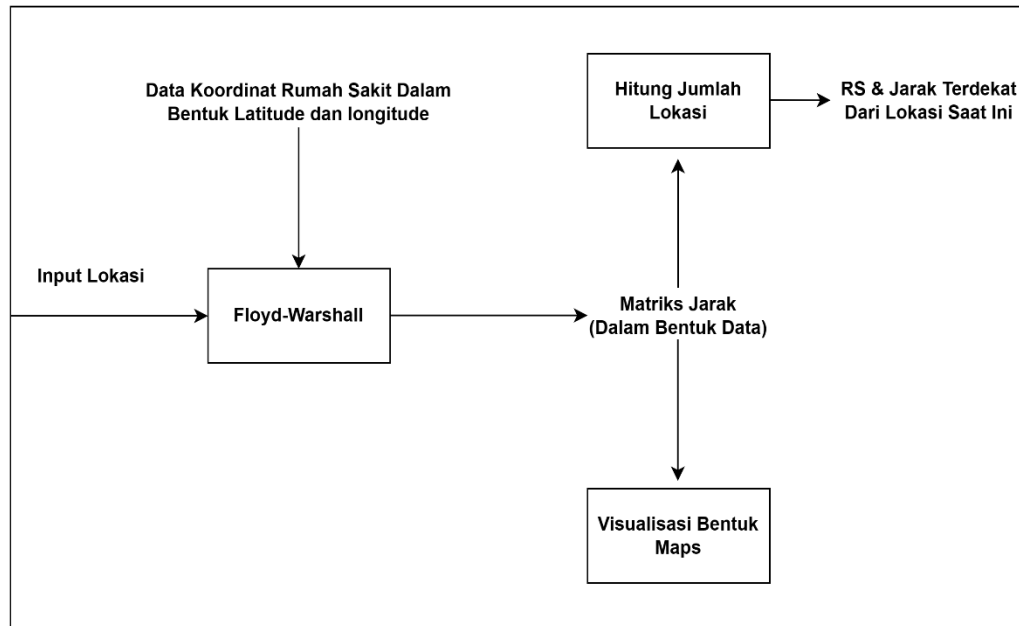
Dengan dasar yang sangat besar, *library* Python dapat melakukan banyak hal yang bergantung pada sistem, seperti ekspresi reguler, pembuatan dokumentasi, pengujian unit, *threading*, *database*, *web browser*, *CGI*, *FTP*, *email*, *XML*, *XML\_RPC*, *HTML*, *file WAV*, *kriptografi*, *GUI (Graphical User Interface)*, dan lainnya (Angelina M. T. I. Sambu et al., 2023).

## BAB 3

### ANALISIS DAN PERANCANGAN

#### 3.1 Rancangan Penelitian

Adapun rancangan penelitian ini dapat dilihat pada diagram umum berikut ini :

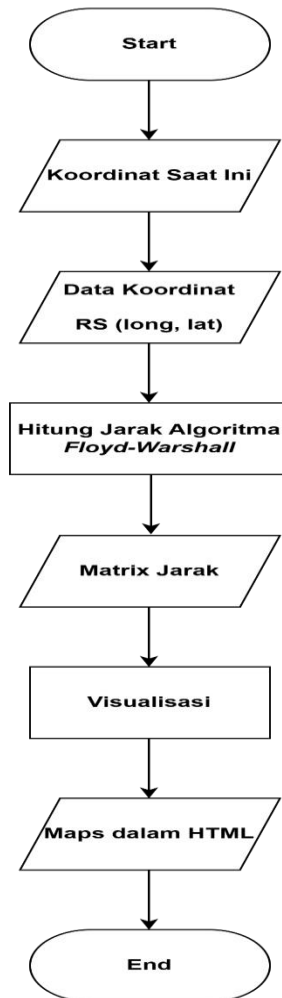


**Gambar 3.1** Diagram Umum Pencarian Jarak Terdekat

Pada gambar 3.1 dapat dilihat diagram umum pencarian jarak terdekat dengan menggunakan algoritma *Floyd warshall*. Dimana mekanisme diagram umum tersebut yaitu *input* lokasi saat ini, kemudian nantinya akan mendapatkan data koordinat *latitude* dan *longitude* rumah sakit dengan menggunakan algoritma *Floyd warshall*. Kemudian hitung jumlah lokasi rumah sakit dalam bentuk data matriks. Kemudian visualisasikan lokasi jarak rumah sakit kedalam maps dalam bentuk maps.

### 3.2 Flowchart Sistem

Adapun *flowchart* sistem yang digunakan dapat dilihat pada gambar 3.2 berikut ini.

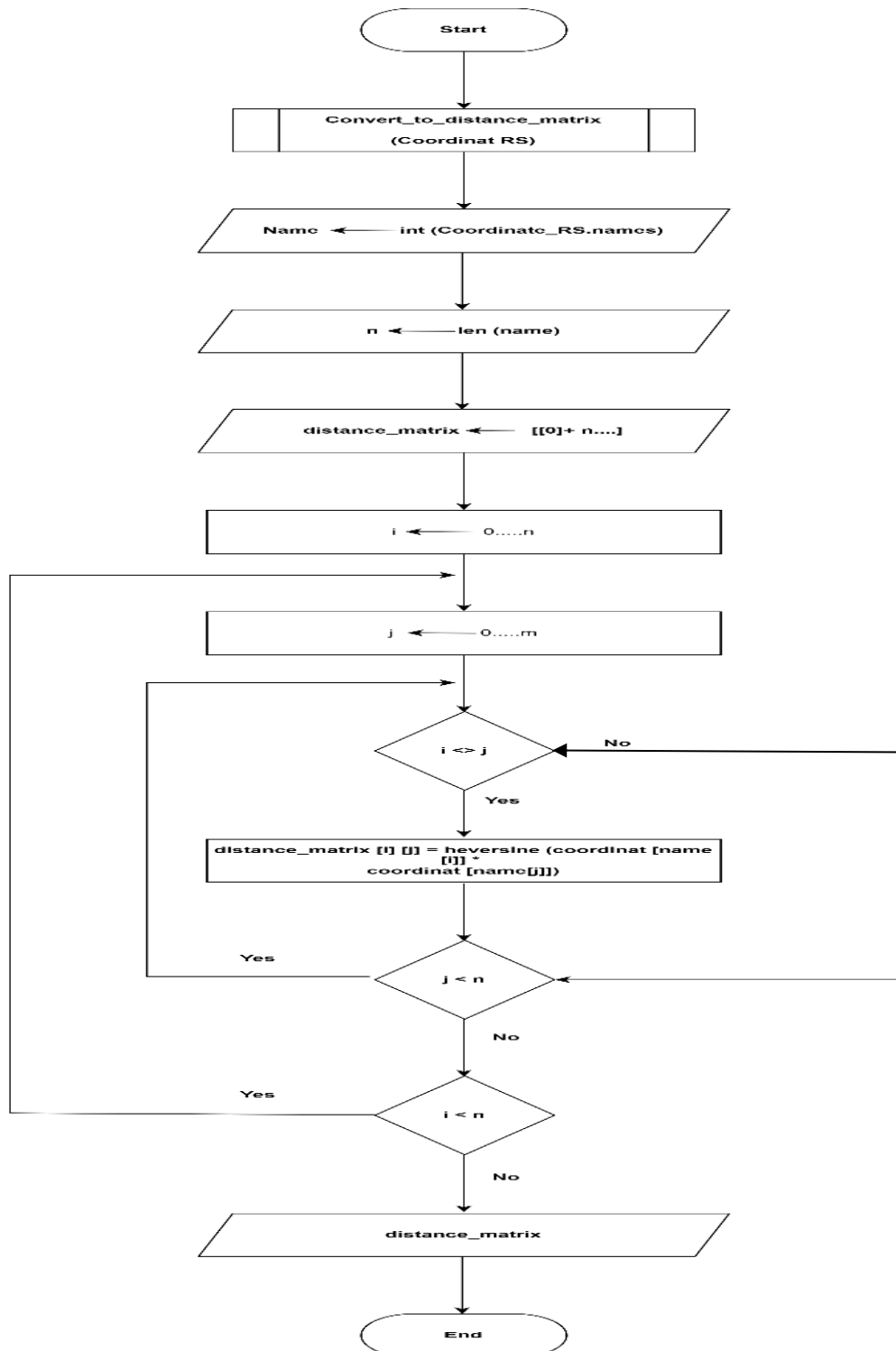


**Gambar 3.2** *Flowchart* system

Pada gambar 3.2 dapat dilihat *flowchart* sistem, dimana *flowchart* dimulai dari *start*, kemudian *input* titik koordinat yang nantinya akan mengeluarkan data koordinat rumah sakit berdasarkan *latitude* dan *longitude*, kemudian tahapan selanjutnya adalah menghitung jarak dengan menggunakan algoritma *Floyd warshall* yang nantinya akan menghasilkan matrix jarak. Kemudian tahapan selanjutnya yaitu proses visualisasi yang nantiya akan mengeluarkan *maps* atau peta dalam format *file* HTML, kemudian program selesai.

### 3.2.1 Flowchart Konversi Koordinat Menjadi Matrix

Adapun *flowchart* Konversi koordinat mejadi matrix dapat dilihat pada gambar 3.3 berikut ini.

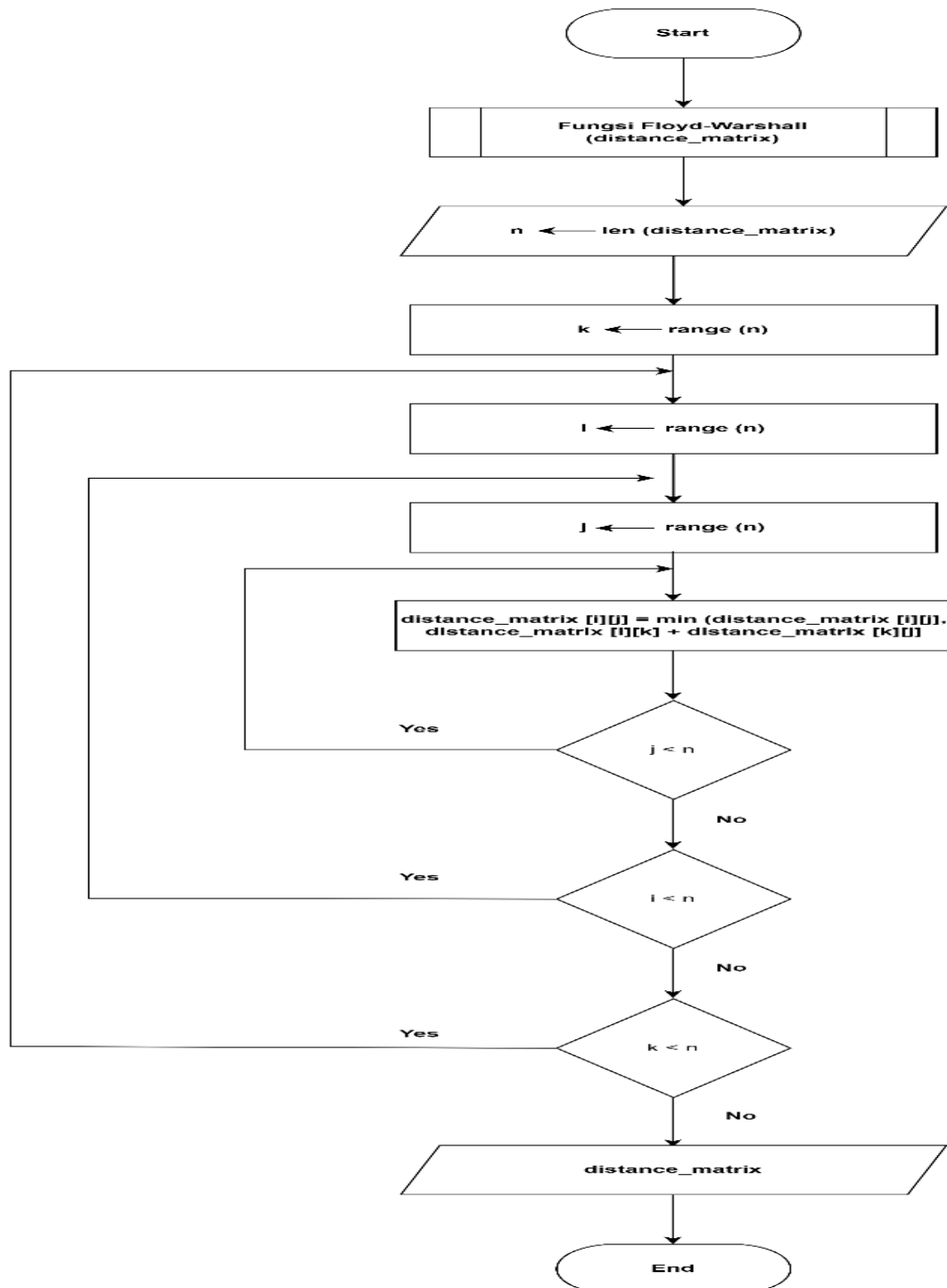


**Gambar 3.3** Flowchart Konversi Koordinat Menjadi Matrix

Gambar 3.3 merupakan *flowchart fungsi* konversi koordinat menjadi matrix, dimana fungsi ini parameter *inputannya* adalah koordinat rumah sakit. Selanjutnya *variable name* diisi dengan list nama-nama rumah sakit. *Variable n* diisi dengan Panjang *variable name*. *variable distance matrix* diisi dengan array nilai masing-masing urutan matrix dikali  $n$ . Proses perulangan untuk *distance matrix* setiap  $i$  dan  $j$  menggunakan fungsi *heversine*. Proses ini akan berlaku jika nilai  $i \neq j$ .

### 3.2.2 Flowchart Algoritma Floyd Warshall

Adapun *flowchart* Algoritma *Floyd Warshall* dapat dilihat pada gambar 3.4 berikut ini.



**Gambar 3.4** *Flowchart* Algoritma *Floyd Warshall*

Gambar 3.4 merupakan *flowchart* fungsi algoritma *Floyd warshall* dimana parameter *inputannya* adalah *distance matrix* hasil dari konversi koordinat menjadi *matrix*. *Variable n* diisi Panjang *distance matrix*, kemudian hitung nilai *distance matrix i* dan *j* menggunakan fungsi *minimum* antara *distance matrix i,j* dengan *distance matrix i,k + distance matrix k,j*.



## BAB 4

### HASIL DAN PEMBAHASAN

#### 4.1 Implementasi Sistem

Implementasi sistem merupakan tahapan lanjutan dari proses analisis dan perancangan sistem, dimana pada tahap ini dibangun sebuah sistem sebagai implementasi dari hasil tahapan sebelumnya. Pada penelitian ini, sistem dibangun dengan menggunakan Bahasa pemrograman python.

Setelah selesai merancang sistem yang telah dibuat, Langkah selanjutnya yaitu mencoba mengimplementasikan sistem.

##### 4.1.1 Ruang Lingkup Penerapan

Sistem ini dibuat dan diuji dengan menggunakan spesifikasi *software* dan *hardware* sebagai berikut:

Spesifikasi *Software*:

1. Sistem Operasi Windows 10 *Home Single Language* 64 bit.
2. Visual Studio Code.
3. Python 3.12-64 bit.

Spesifikasi *Hardware*:

1. Intel Core i3-6006U 2.000 GHz.
2. Memory 4096 Ram.
3. Harddisk 500 GB.

#### 4.2 Tampilan Program

Tampilan program merupakan halaman yang pertama kali sistem akan dijalankan. Halaman ini berisikan syntax-syntax python yang berfungsi untuk mengubah titik lokasi asal hingga titik lokasi tujuan.

```

# =====
# Program Utama
# =====

# Koordinat Saat ini
# Lokasi : Fakultas Ilmu Komputer Universitas Sumatera Utara
origin_coord = (3.5627538774238974, 98.6597629101321)

# Koordinat Rumah Sakit
coordinates = {
    "RS Methodist" : (3.5858612863874493, 98.69308855515669),
    "RS Deli Medan" : (3.587898696699665, 98.68778089089506),
    "RS Murni Teguh Memorial" : (3.594254043002926, 98.68209763338041),
    "RS Columbia Asia Medan" : (3.5891143169534456, 98.67660446925613),
    "RSU Martha Friska Multatuli" : (3.579177431290832, 98.68192597200152),
    "RSUD Dr. Pirngadi" : (3.6029915107998804, 98.68913574991466),
    "Siloam Hospitals" : (3.588942992236914, 98.67334290321371),
    "RSU Madani" : (3.5783207981486376, 98.70321198313947),
    "RSU Materna" : (3.587572393480807, 98.67093964390934),
    "RS Putri Hijau" : (3.6028201886910605, 98.67299958045594),
    "RSI Malahayati" : (3.589628290843874, 98.67248459631928),
    "RSIA Stella Maris" : (3.57660752949485, 98.6796943742324),
    "RSU Sarah Medan" : (3.5911702108293273, 98.66269989772292),
    "RSU Muhammadiyah" : (3.586887093331837, 98.70990677691596),
    "RS Santa Elisabeth" : (3.5784921248378376, 98.67694779217025),
    "RS Univ. Royal Prima" : (3.601352769630362, 98.65301549391648),
    "RSU Bakti" : (3.571162717050642, 98.70122891667422),
    "RS Advent Medan" : (3.594093390651877, 98.65001106218978),
    "RS Bunda Thamrin" : (3.5875830649577187, 98.65052604632643),
    "RSU Bina Kasih" : (3.581757997388521, 98.612760542972),
    "RS Bhayangkara Tk II" : (3.5776461626642773, 98.6572208401029),
    "RSU Imelda Pekerja Indonesia" : (3.625959049796053, 98.67455863937015),
    "RSU Mitra Medika-Amplas" : (3.540373845712716, 98.71015938702203),
    "RS USU" : (3.570455033267342, 98.65707001667424),
    "RSU Sri Ratu" : (3.5939266925203333, 98.66479477872403),
    "RSU Bahagia" : (3.5658959332498648, 98.70117335529535),
    "RS Mitra Medika Premiere" : (3.5893677093481027, 98.66649775676082),
    "RS Hermina Medan" : (3.5983491565495895, 98.6274440008109),
    "RSU Haji Medan" : (3.6137451865209864, 98.71452867159807),
    "RS Martha Friska" : (3.636642495302991, 98.66971164632642),
    "RSU Fajar" : (3.5512868103402124, 98.67095079184195),
}

```

**Gambar 4.1** Tampilan program

Pada gambar 4.1 dapat dilihat tampilan program, dimana pada tampilan program tersebut terdapat syntax-syntax python yang berfungsi sebagai memasukkan titik lokasi pertama user dengan cara memasukkan titik koordinat *latitude* dan *longitudenya*. Kemudian juga pada tampilan program tersebut sudah terdapat beberapa titik koordinat lokasi rumah sakit seperti rumah sakit methosit, rumah sakit Deli Medan, rumah sakit Murni Teguh dan lainnya.

### 4.2.1 Tampilan Perhitungan Jarak Terpendek Rumah Sakit

Tampilan perhitungan jarak rumah sakit terpendek merupakan tampilan hasil perhitungan jarak rumah sakit dengan menggunakan algoritma *Floyd-warshall*. Adapun tampilan perhitungan jarak tersebut dapat dilihat pada gambar 4.2 berikut ini :

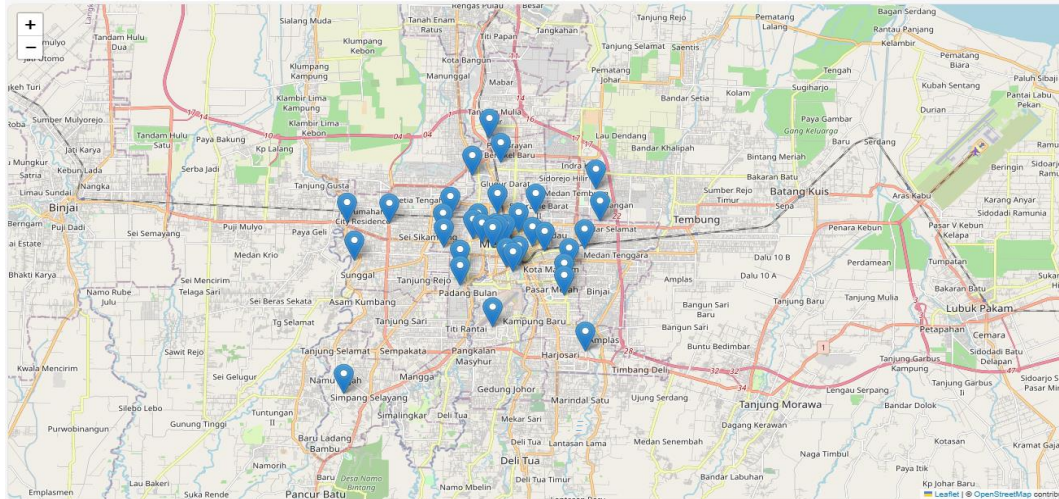
```
Jarak terpendek antar setiap pasangan node (dalam meter):
[0, 631.0944267384311, 1535.7990921868234, 1864.7737410897532, 1444.6385115482074, 1954.6532483087196, 2217.9449839627355, 1401.8620252332885, 2465.375394274952, 2
919.973561469356, 2324.6151319163378, 1807.8534632737544, 3423.715844000646, 1869.9207812164357, 1969.7883822823358, 4769.118402248625, 1067.4633263771393, 4867.44
5743142186, 4727.34324192537, 8926.268732917937, 4083.988862485777, 4910.021236372435, 5401.138321672187, 4348.893999858118, 3265.516807089014, 2394.502255167318,
2976.613529906297, 7416.1589832492, 3908.273607519276, 6214.038673563265, 4562.484170759642, 2992.6808365309143, 5110.293585215026, 9402.239554466376, 11828.930657
929553]
[631.0944267384311, 0, 947.2027878271001, 1247.6684357285346, 1167.3170251556467, 1684.9662680213232, 1606.4855850570468, 2016.6624921458363, 1869.3432364677803, 2
333.183457768259, 1708.3948071169355, 1543.2755204213242, 2807.0789904100793, 2458.0405212460255, 1593.548736156409, 4138.026057383455, 2385.48096116508, 4247.7885
7850153, 4134.579855492254, 8353.513024604701, 3577.9620601701663, 4479.27403567846, 5839.028817377067, 3921.514930178386, 2637.5132641026976, 2862.662775002592, 2
367.577703165456, 6796.051382146973, 4131.699989872797, 5779.098309295253, 4479.078013246361, 3420.186772836758, 4557.213715092208, 8786.694457970963, 11510.106403
916993]
[1535.7990921868234, 947.2027878271001, 0, 835.6150058026377, 1676.5509712827798, 1246.5902373351153, 1136.9744153951076, 2937.6085660357935, 1444.0654337527824, 1
388.0553705292917, 1184.345247484691, 1980.245550867954, 2179.830846454833, 3193.0321646539196, 1843.4735728666572, 3322.537601284987, 3331.742378491685, 3560.891
135949802, 3581.3690853060734, 7819.268322910198, 3321.4644610513405, 3623.3486468613855, 6752.2783711319415, 3836.3522274117977, 1920.5058187010554, 3797.99726221
5149, 1814.4793903074192, 6082.308504111228, 4201.227646785361, 4909.710603725895, 4935.292143886686, 3840.860745786978, 3612.15622317503, 8088.757088562764, 11519
45961221535]
[1864.7737410897532, 1247.6684357285346, 835.6150058026377, 0, 1252.8536156598589, 2077.2680527076054, 362.4592858616338, 3187.424950586077, 651.6257094084397, 157
5.6561316638451, 460.76843428582593, 1432.3441437197152, 1559.925373380864, 3704.0776094931184, 1181.7482476623843, 2950.400660953406, 3384.162265758866, 3002.7266
44462851, 2899.1075283643377, 7132.256966218733, 2500.71411602832, 4103.233196680616, 6575.761132661743, 3000.7718844912595, 1415.6317617850443, 3754.9873170157966,
1121.9646406583781, 5551.449455347212, 5021.346502022406, 5339.962069419618, 4252.766775088498, 4555.571866868778, 3788.0466385566706, 7599.611608039553, 10684.301
183071588]
[1444.6385115482074, 1167.3170251556467, 1676.5509712827798, 1252.8536156598589, 0, 2766.245850657732, 1444.4531780676866, 2364.200525964932, 1535.550400208565, 2
809.4006276388195, 1564.6972545078834, 378.14498393276966, 2516.1111906526559, 3221.4045788149147, 557.6994329688388, 4046.4623823931965, 2320.19462803194, 3910.93
47038050273, 3607.8572835693185, 7681.194439089236, 2747.011400496374, 5265.737998271252, 5332.455755063659, 2924.018156764963, 2510.8149927378277, 2594.8753760854
32, 2053.1693650936515, 6411.050645352253, 8278.753708786594, 4532.0111078696755, 3331.9082850528625, 4437.302157612773, 5039.255674894808, 8341.66207429843, 10394
316557542323]
[2217.9449839627355, 1606.4855850570468, 1136.9744153951076, 362.4592858616338, 1444.4531780676866, 2347.7505389920357, 0, 3518.941433142579, 307.179340076387, 154
3.5441477467848, 121.99228568442215, 1542.1549382283292, 1206.8104386439, 4064.182735630648, 1229.0187666365966, 2644.432798750475, 3672.355912460634, 2651.865466
218495, 2536.4561256483005, 6770.569046998169, 2186.1159438814907, 4118.208374031743, 6772.104783865717, 2736.342328437403, 1098.6428857736173, 4013.328840110483,
761.119221925068, 5199.963190693866, 5338.194493350367, 5319.22902518496, 4195.583719879478, 4910.962065480063, 3669.3655504649632, 7184.306510752288, 10416.84038
1829417]
[1401.8620252332885, 2016.6624921458363, 2937.6085660357935, 3187.424950586077, 2364.200525964932, 3156.853632715489, 3518.941433142579, 0, 3726.321620360883, 4320
```

**Gambar 4.2** Tampilan Hasil Perhitungan Jarak Terpendek Rumah Sakit

Pada gambar 4.2 dapat dilihat tampilan hasil perhitungan jarak terpendek rumah sakit, dimana pada gambar tersebut dapat dilihat hasil perhitungan dari titik mulai perjalanan hingga titik rumah sakit yang akan dituju. Perhitungan tersebut menggunakan algoritma *Floyd-warshall* sebagai metode perhitungan dan juga *latitude* dan *longitude* sebagai bobot perhitungan.

#### 4.2.2 Tampilan Peta Rumah Sakit

Tampilan peta rumah sakit merupakan titik-titik rumah sakit yang ada dikota medan. Titik rumah sakit ini digunakan untuk mengetahui letak rumah sakit yang ada dikota medan.

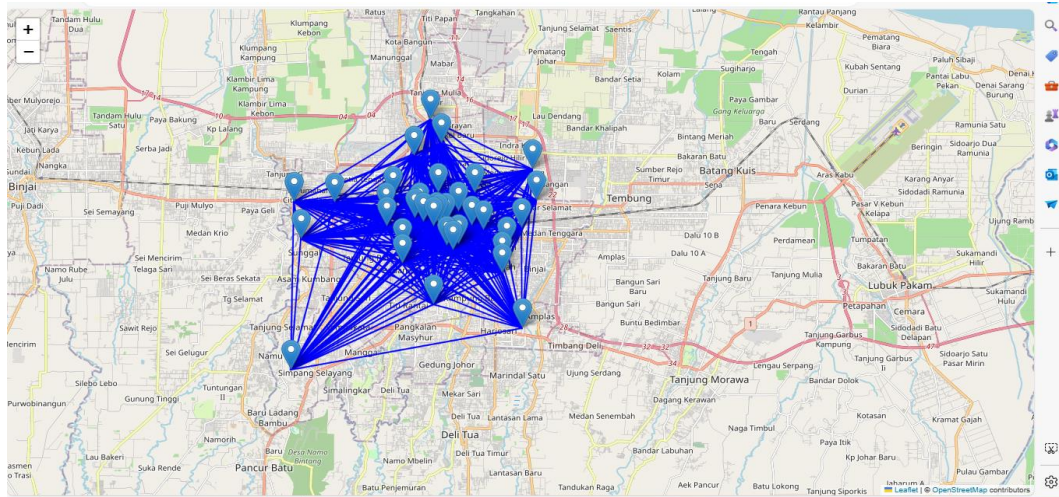


**Gambar 4.3** Tampilan Peta Rumah Sakit

Pada gambar 4.3 dapat dilihat tampilan peta rumah sakit. Tampilan peta ini berfungsi untuk menampilkan titik-titik rumah sakit yang ada dikota medan. Pada penelitian ini penulis menggunakan 20 titik rumah sakit.

#### 4.2.3 Tampilan Hasil Perhitungan Titik Peta Rumah Sakit

Tampilan hasil perhitungan titik peta rumah sakit merupakan tampilan titik peta yang dituju berdasarkan hasil perhitungan dengan menggunakan algoritma *Floyd-warshall*. Adapun tampilan hasil perhitungan titik peta rumah sakit tersebut dapat dilihat pada gambar 4.4 berikut ini.



**Gambar 4.4** Tampilan Hasil Perhitungan Titik Peta Rumah Sakit

Pada gambar 4.4 dapat dilihat tampilan hasil perhitungan titik peta rumah sakit. Pada tampilan tersebut dapat dilihat titikp peta rumah sakit dengan garis-garis biru sebagai arah dari perhitungan jarak rumah sakit tersebut. Garis-garis biru tersebut merupakan hasil perhitungan jarak terpendek dari titik koordinat awal sampai titik rumah sakit yang ada dikota medan.

## **BAB 5**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Setelah melakukan Analisis, Implementasi Sistem, dan Pengujian maka dapat disimpulkan bahwa:

1. Implementasi algoritma *Floyd warshall* dalam pencarian rumah sakit terdekat di Kota Medan berhasil dilakukan. Algoritma ini terbukti efektif dalam menghitung jarak terpendek antara titik pengguna dengan rumah sakit yang akan dituju di Kota Medan. Dan juga mampu menangani graf dengan jumlah simpul yang relatif banyak dan memberikan hasil yang akurat dalam waktu yang efisien.
2. Dengan menggunakan Algoritma *Floyd Warshall*, rute optimal antara titik pengguna dengan rumah sakit yang akan dituju dapat diidentifikasi dengan jelas. Hal ini memungkinkan pengelolaan rute yang lebih baik dan respons darurat yang lebih cepat.
3. Algoritma ini dapat diadaptasi dan digunakan untuk berbagai kota atau jaringan lainnya dengan sedikit penyesuaian pada data input, menunjukkan fleksibilitas dan skalabilitas algoritma.

#### **5.2 Saran**

1. Untuk meningkatkan hasil Analisa data, algoritma *Floyd warshall* dapat diintegrasikan dengan sistem informasi geografis untuk meningkatkan pemetaan interaktif dan analisis yang lebih mendalam.

2. Untuk meningkatkan hasil keakuratan pencarian jarak, pengumpulan data yang lebih rinci mengenai kondisi jalan, lalu lintas, dan waktu tempuh pada berbagai waktu dapat dilakukan.

### Daftar Pustaka

- Angelina M. T. I. Sambu Ua, Diandra Lestriani H, Elizabeth Sonia Kristanty Marpaung, Jesslyn Ong, Michelle Savinka, Putri Nurhaliza, & Rahmi Yulia Ningsih. (2023). Penggunaan Bahasa Pemrograman Python Dalam Analisis Faktor Penyebab Kanker Paru-Paru. *Jurnal Publikasi Teknik Informatika*, 2(2), 88–99. <https://doi.org/10.55606/jupti.v2i2.1742>
- Arga, E. S., Firmansyah, G. G., Imam, K., & Fauzi, M. (2021). Penerapan Algoritma Dijkstra pada Pencarian Jalur Terpendek. *Jurnal Bayesian*, 1(2), 134–142.  
<https://bayesian.lppmbinabangsa.id/index.php/home/article/download/15/19>
- Arthalia Wulandari, I., & Sukmasetyan, P. (2022). Implementasi Algoritma Dijkstra untuk Menentukan Rute Terpendek Menuju Pelayanan Kesehatan. *Jurnal Ilmiah Sistem Informasi (JISI)*, 1(1), 30–37.  
<https://doi.org/10.24127/jisi.v1i1.1953>
- Baharudin, I., Purwanto, A. J., Budiman, T. R., & Fauzi, M. (2021). Implementasi Algoritma Dijkstra Untuk Menentukan Jalur Terpendek Dalam Distribusi Barang. *Jurnal Pilar Nusa Mandiri*, 13(2), 181–186.  
<https://doi.org/https://doi.org/10.46306/lb.v2i2.74>
- Buako, Z., Yahya, L., & Achmad, N. (2021). Aplikasi Algoritma Floyd-Warshall Dengan Pendekatan Madm Dalam Menentukan Rute Terpendek Pengangkutan Sampah. *Euler: Jurnal Ilmiah Matematika, Sains Dan Teknologi*, 9(2), 62–70. <https://doi.org/10.34312/euler.v9i2.10979>
- Hasibuan, A. R. (2019). Penerapan Algoritma Floyd Warshall Untuk Menentukan Jalur Terpendek Dalam Pengiriman Barang. *Jurnal Riset Komputer (JURIKOM)*, 3(6), 20–24.
- Jalolov, T. S. (2023). Solving Complex Problems in Python. *AMERICAN Journal of Language, Literacy and Learning in STEM Education Wwww. Grnjournal.Us AMERICAN Journal of Language, Literacy and Learning in STEM Education*, 01(09), 2993–2769.
- Maulani, A. (2023). *Teori graf* (C. Basir & A. S. Husein (eds.); Issue 1). UNPAM PRESS. <https://anggigeo.wordpress.com/tag/teori-graf/>
- Melladia. (2020). Algoritma Genetika Menentukan Jalur Jalan dengan Lintasan



- Terpendek (Shortest Path). *Prosiding Seminar Nasional Sistem Informasi Dan Teknologi (SISFOTEK)*, 4(1), 112–117.  
<https://seminar.iaii.or.id/index.php/SISFOTEK/article/view/162>
- Pratama, A., Fauzi, H., Nur Indira, Z., & Purnama Adi, P. (2023). Analisis Faktor Penyebab Pending Klaim Rawat Inap Akibat Koding Rekam Medis Di Rumah Sakit Umum Daerah (RSUD) Dr. Soedirman Kebumen. *Jurnal Ilmiah Perekam Dan Informasi Kesehatan Imelda (JIPIKI)*, 8(1), 124–134.  
<https://doi.org/10.52943/jipiki.v8i1.1225>
- Puspita, Z. A., Fauziah, F., & Sholihati, I. D. (2023). Metode Haversine Formula Pada Pencarian Rumah Sakit Di Wilayah Jakarta Selatan Berbasis Android. *JUPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, 8(4), 1142–1153. <https://doi.org/10.29100/jupi.v8i4.3962>
- Putra, A. B. W., Rachman, A. A., Santoso, A., & Mulyanto, M. (2020). Perbandingan Hasil Rute Terdekat Antar Rumah Sakit di Samarinda Menggunakan Algoritma A\*(star) dan Floyd-Warshall. *Jurnal Sisfokom (Sistem Informasi Dan Komputer)*, 9(1), 59–68.  
<https://doi.org/10.32736/sisfokom.v9i1.685>
- Riksa Herlambang, I., Nurkamal Fauzan, M., & Nuraini Siti Fathonah, R. (2021). Penentuan Rute Terpendek Pendistribusian Barang Menggunakan Algoritma Floyd-Warshall. *Agustus*, 20(3), 430–439.  
<https://doi.org/https://doi.org/10.33633/tc.v20i3.4686>
- Surbakti, N. M., Septiana, D., Matematika, P. S., & Medan, U. N. (2022). SCRAMBLING INDEX DARI GRAF JARING LABA-LABA. *Deli Sains Informatika*, 2(1), 1–4.  
<https://jurnal.unds.ac.id/index.php/dsi/article/view/152>
- Syarifudin, I., & Ramanda, K. (2021). Penerapan Metode Dijkstra Pada Sistem Informasi Pencarian Jarak Terpendek Menuju Rumah Sakit di Wilayah Jakarta Barat. *Jurnal Sains Komputer & Informatika (J-SAKTI)*, 5(2), 540–550.
- Wita, D. S. (2022). Implementasi Algoritma Dijkstra Untuk Penentuan Rute Terpendek Puskesmas di Samarinda. *Inspiration: Jurnal Teknologi Informasi Dan Komunikasi*, 12(1), 88. <https://doi.org/10.35585/inspir.v12i1.2656>
- Yohana, Y., Syaripuddin, & A, Q. Q. (2023). Penentuan Rute Terpendek

Distributor Minimarket Menggunakan Algoritma Floyd Warshall. *BASIS Jurnal Ilmiah Matematika*, 2(2), 33–41.  
<http://jurnal.fmipa.unmul.ac.id/index.php/Basis/article/view/1078>

## LISTING PROGRAM

```

import math
import folium
from folium import plugins
from geopy.distance import geodesic

#
=====

# Fungsi-fungsi untuk membuat visualisasi di Maps
#
=====

# Fungsi untuk mengonversi matriks jarak menjadi edge list
def distance_matrix_to_edge_list(distance_matrix):
    edge_list = []
    n = len(distance_matrix)
    for i in range(n):
        for j in range(i + 1, n):
            if distance_matrix[i][j] != 0:
                edge_list.append((i, j, distance_matrix[i][j]))
    return edge_list

# Fungsi untuk membuat peta dengan edge list
def create_map(coordinates, edges):
    # Membuat peta
    m = folium.Map(location=[list(coordinates.values())[0][0],
                             list(coordinates.values())[0][1]], zoom_start=12)

    # Menambahkan marker untuk setiap titik

```

```

for name, value in coordinates.items():
    folium.Marker([value[0], value[1]], popup=name).add_to(m)

locations = list(coordinates.values())
# Menambahkan garis untuk setiap edge
for edge in edges:
    src, dst, dist = edge
    src_coord = locations[src]
    dst_coord = locations[dst]
    folium.PolyLine(locations=[(src_coord[0], src_coord[1]), (dst_coord[0],
        dst_coord[1])], color='blue', weight=2, opacity=1).add_to(m)

return m

def create_map_only(coordinates):
    # Membuat peta
    m = folium.Map(location=[list(coordinates.values())[0][0],
        list(coordinates.values())[0][1]], zoom_start=12)

    # Menambahkan marker untuk setiap titik
    for name, value in coordinates.items():
        folium.Marker([value[0], value[1]], popup=name).add_to(m)

    return m

#
=====

#
=====

```

```

# Fungsi-fungsi untuk perhitungan Floyd-Warshall
#
=====

# Fungsi untuk menghitung jarak antara dua titik dalam koordinat longitude dan
latitude
def haversine(coord1, coord2):
    lat1, lon1 = coord1
    lat2, lon2 = coord2
    R = 6371000 # Radius bumi dalam meter

    dlat = math.radians(lat2 - lat1)
    dlon = math.radians(lon2 - lon1)
    a = math.sin(dlat / 2) * math.sin(dlat / 2) + math.cos(math.radians(lat1)) *
        math.cos(math.radians(lat2)) * math.sin(dlon / 2) * math.sin(dlon / 2)
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
    distance = R * c

    return distance

# Fungsi untuk mengonversi nama titik dan koordinat ke dalam matriks jarak
menggunakan Haversine formula
def convert_to_distance_matrix(coordinates):
    names = list(coordinates.keys())
    n = len(names)
    distance_matrix = [[0] * n for _ in range(n)]

    for i in range(n):
        for j in range(n):
            if i != j:
                distance_matrix[i][j] = haversine(coordinates[names[i]], coordinates[names[j]])

```

```
return distance_matrix
```

```
# Fungsi Floyd-Warshall yang menggunakan jarak antar titik
```

```
def floyd_warshall(distance_matrix):
```

```
    n = len(distance_matrix)
```

```
    for k in range(n):
```

```
        for i in range(n):
```

```
            for j in range(n):
```

```
                distance_matrix[i][j] = min(distance_matrix[i][j], distance_matrix[i][k] +
                                                distance_matrix[k][j])
```

```
    return distance_matrix
```

```
# Fungsi untuk menjalankan algoritma Floyd-Warshall dengan nama titik dan
    koordinat
```

```
def floyd_warshall_with_coordinates(coordinates):
```

```
    distance_matrix = convert_to_distance_matrix(coordinates)
```

```
    return floyd_warshall(distance_matrix)
```

```
def get_distance(origin, destination):
```

```
    return geodesic(origin, destination).meters
```

```
def find_nearest_distances(origin_coord, coordinates):
```

```
    min_distance = float('inf')
```

```
    nearest_location = None
```

```
    for name, coordinate in coordinates.items():
```

```
        distance = haversine(origin_coord, coordinate)
```

```
        if distance < min_distance:
```

```
            min_distance = distance
```

```
nearest_location = name
```

```
return nearest_location, min_distance
```

```
#
```

```
=====
```

```
=====
```

```
# Program Utama
```

```
#
```

```
=====
```

```
=====
```

```
# Koordinat Saat ini
```

```
# Lokasi : Fasilkom-Ti Universitas Negeri Medan
```

```
origin_coord = (3.6084201172825, 98.71700920820898)
```

```
# Koordinat Rumah Sakit
```

```
coordinates = {
```

```
"RS Methodist" : (3.5858612863874493, 98.69308855515669),
```

```
"RS Deli Medan" : (3.587898696699665, 98.68778089089506),
```

```
"RS Murni Teguh Memorial" : (3.594254043002926, 98.68209763338041),
```

```
"RS Columbia Asia Medan" : (3.5891143169534456, 98.67660446925613),
```

```
"RSU Martha Friska Multatuli" : (3.579177431290832, 98.68192597200152),
```

```
"RSUD Dr. Pirngadi" : (3.6029915107998804, 98.68913574991466),
```

```
"Siloam Hospitals" : (3.588942992236914, 98.67334290321371),
```

```
"RSU Madani" : (3.5783207981486376, 98.70321198313947),
```

```
"RSU Materna" : (3.587572393480807, 98.67093964390934),
```

```
"RS Putri Hijau" : (3.6028201886910605, 98.67299958045594),
```

```
"RSI Malahayati" : (3.589628290843874, 98.67248459631928),
```

```
"RSIA Stella Maris" : (3.57660752949485, 98.6796943742324),
```

```

"RSU Sarah Medan" : (3.5911702108293273, 98.66269989772292),
"RSU Muhammadiyah" : (3.586887093331837, 98.70990677691596),
"RS Santa Elisabeth" : (3.5784921248378376, 98.67694779217025),
"RS Univ. Royal Prima": (3.601352769630362, 98.65301549391648),
"RSU Bakti" : (3.571162717050642, 98.70122891667422),
"RS Advent Medan" : (3.594093390651877, 98.65001106218978),
"RS Bunda Thamrin" : (3.5875830649577187, 98.65052604632643),
"RSU Bina Kasih" : (3.581757997388521, 98.612760542972),
"RS Bhayangkara Tk II" : (3.5776461626642773, 98.6572208401029),
"RSU Imelda Pekerja Indonesia" : (3.625959049796053, 98.67455863937015),
"RSU Mitra Medika-Amplas" : (3.540373845712716, 98.71015938702203),
"RS USU" : (3.570455033267342, 98.65707001667424),
"RSU Sri Ratu" : (3.5939266925203333, 98.66479477872403),
"RSU Bahagia" : (3.5658959332498648, 98.70117335529535),
"RS Mitra Medika Premiere" : (3.5893677093481027, 98.66649775676082),
"RS Hermina Medan" : (3.5983491565495895, 98.6274440008109),
"RSU Haji Medan": (3.6137451865209864, 98.71452867159807),
"RS Martha Friska" : (3.636642495302991, 98.66971164632642),
"RSU Fajar" : (3.5512868103402124, 98.67095079184195),
"RS Columbia Asia Aksara" : (3.599569636517977, 98.71629523253756),
"RSU Sufina Aziz" : (3.620054823848279, 98.66232045529536),
"RSU Sundari" : (3.5986825554528936, 98.60934533943201),
"RSU Pusat H. Adam Malik" : (3.5214070069317405, 98.60829467805311),
}

```

```

# Memanggil fungsi Floyd-Warshall dengan nama titik dan koordinat
result = floyd_warshall_with_coordinates(coordinates)

```

```

# Menampilkan hasil
print("Jarak terpendek antar setiap pasangan node (dalam meter):")
for row in result:
    print(row)

```



```
nearest_location, min_distance = find_nearest_distances(origin_coord,
    coordinates)
print("Lokasi Hospital terdekat: ", nearest_location)
print("Jarak : ", min_distance, " meter")
```

```
edges = distance_matrix_to_edge_list(result)
mapsonly = create_map_only(coordinates)
mymap = create_map(coordinates, edges)
```

```
mapsonly.save("map_rumah_sakit.html")
mymap.save("map_floyd_warshall_final.html")
print("Maps sudah di buat, mohon cek folder lokasi program...")
```

## CURRICULUM VITAE

### DATA DIRI

---



Nama : Surya Andika Ramadani Ginting  
Tanggal Lahir : 11 Januari 1999  
Jenis Kelamin : Laki-laki  
Alamat : Dusun 1 desa sembahe baru . Kec.  
Pancur batu  
Email : [surya.ginting07@gmail.com](mailto:surya.ginting07@gmail.com)

### RIWAYAT PENDIDIKAN

---

2006 – 2011 : SD swasta NUR ADIA, Medan  
2011 – 2014 : SMP swasta rakyat, Medan  
2014 – 2017 : SMA negeri 1 Sunggal, Medan  
2017 – 2024 : S-1 Ilmu Komputer Universitas Sumatera Utara

### PENGALAMAN MAGANG

---

2020 : Rumah Sakit Umum Pusat Haji Adam Malik Medan

### KEMAMPUAN

---

Pemrograman : HTML, Pascal, C, C++, C#, Java, PHP, Python,  
IDE : NPP, CodeBlocks, SharpDevelop, Netbeans, Visual  
Studio, Android Studio  
Basis data : MySQL  
Perangkat lunak : Ms. Office, Adobe Photoshop, Adobe Illustrator, Unity,