**PRAKTIKUM IMPLEMENTASI ALGORITMA DIJKSTRA DALAM PEMODELAN JARINGAN TRANSPORTASI ANTAR KOTA DI INDONESIA**

**Mata Kuliah**

Struktur Data

**Oleh**

1. Meyssa Aqila Adikara (24091397076)
2. Diana Safitri (24091397084)
3. Imelya Urivaturosiah (24091397091)
4. Octavia Habeahan (24091397094)

**Kelas**

2024C

****

**PROGRAM STUDI MANAJEMEN INFORMATIKA**

**FAKULTAS VOKASI**

**UNIVERSITAS NEGERI SURABAYA**

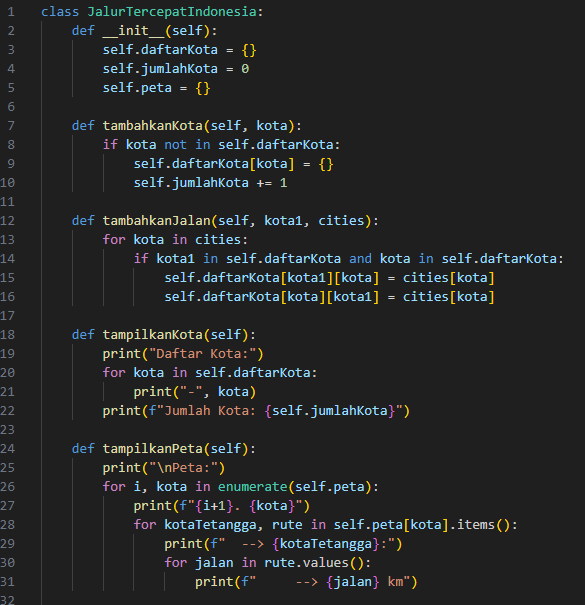
**2025**

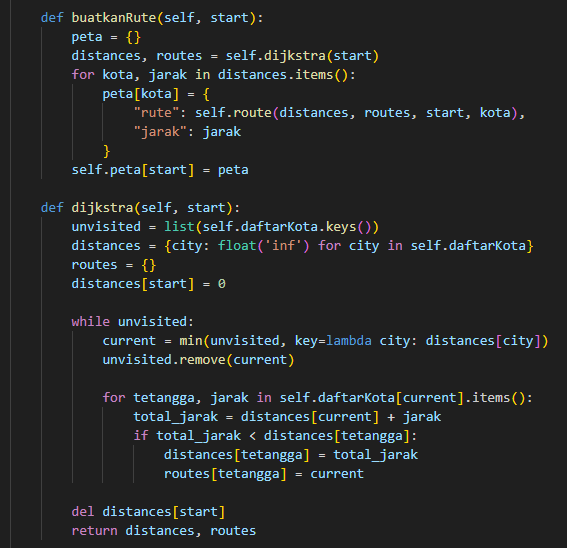
**Sistem Rekomendasi Rute Perjalanan Jarak Terpendek Menggunakan Graf dan Algoritma Greedy**

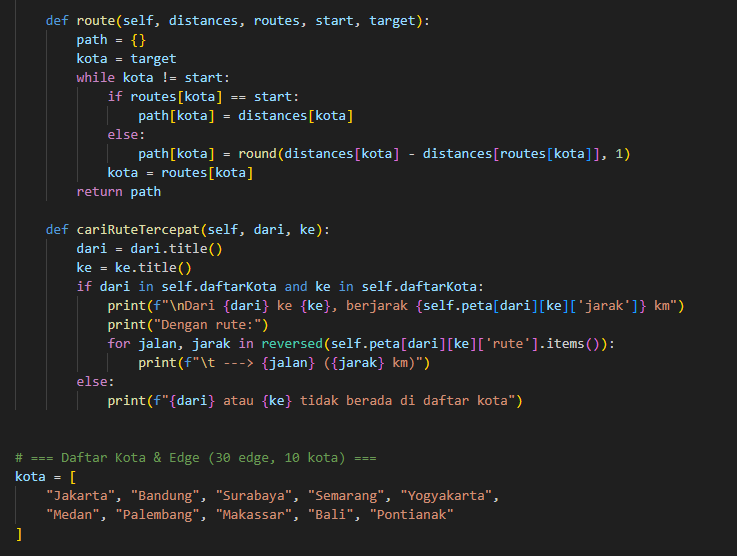
**Tujuan Program**

Program ini dibuat untuk menentukan rute terpendek (berdasarkan jarak) dari satu kota ke kota lain di Indonesia menggunakan algoritma Dijkstra. Penerapanya berbasisbstruktur data graf berbobot tak berarah.

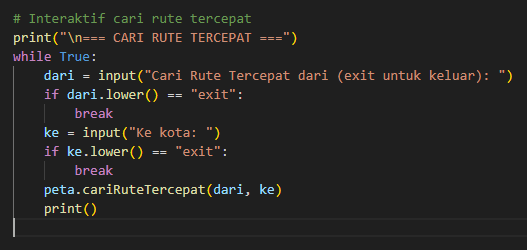
Kode lengkap

****

****

****

****

****

Penjalasan Struktur Program

1. class JalurTercepatIndonesia

Mendefinisikan sebuah class (struktur data) untuk membuat pbjek peta yang berisi kota dan jalur antar kota dengan konsep graph berbobot (berbasis jarak)

1. \_\_init\_\_(self)

Inisialisasi objek:

* daftarKota: dictionary menyimpan semua kota dan kota tetangganya
* jumlahKota: menghitung hasil perhitungan rute terpendek dari setiap kota.
* peta: menyimpan hasil perhitungan rute terpendek dari setiap kota.

1. tambahkanKota(self, kota)

Menambahkan nama kota ke dalam daftar jika belum ada

1. tambahkanJalan(self, kota1, cities)

Menambahkan jalur antar kota lengkap dengan jaraknya (km), bersifat dua arah (undirected graph).



Artinya: Jakarta ↔ bandung dan Jakarta ↔ Semarang dengan jarak masing-masing.

1. tampilkanKota(self)

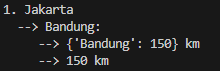
Menampilkan seluruh kota yang sudah ditambahkan dan jumlahnya

1. tampilkanPeta(self)

Menampilkan isi dari peta:

* menunjukan dari kota A ke kota B’
* disertai jarak rutenya.

Outpunya akan seperti ini



1. buatkanRute(self, start)

Membuat dan menyimpan semua rute tercepat (berdasarkan jarak) dari start ke semua kota lainya, menggunakan algoritma dijkstra.

1. Dijkstra(self, start)

Implementasi algoritma Dijkstra:

* Menghitung jarak minimum dari start ke semua kota lainnya,
* Menggunkan pendekatan greedy unruk mencari kota terdekat yang belum dikunjungi.

Outputnya:

* distances → total jarak dari start ke kota tertentu.
* routes → jalur sebelumnya yang dilalui.

1. Route(self, distances, routes, start, target)

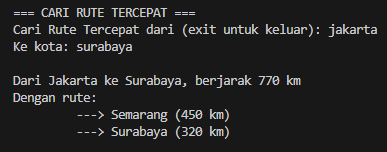
Membangun rute dari start ke target berdasarkan hasil Dijkstra.

1. cariRuteTercepat(self, dari, ke)

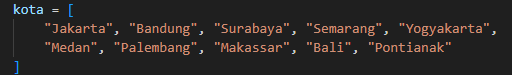
Fungsi interaktif untuk mencari dan menampilkan:

* Jarak total dari **dari** ke **ke**
* Kota-kota yang dilalui (rute)
* Panjang setiap segmen rute

Output:



1. Daftar kota



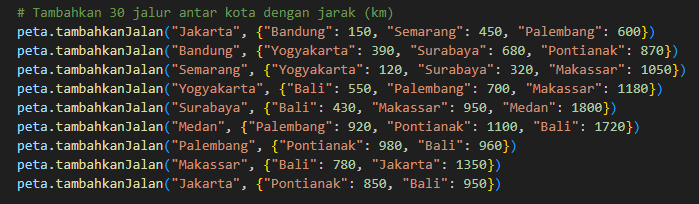
Kumpulan nama kota di Indonesia sebagai simpul dalam graf.

1. Loop tambah kota



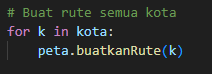
Menambahkan semua kota ke dalam objek peta

1. Penambahan 30 jalur (edge)



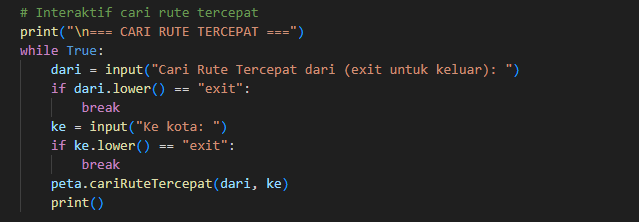
Menambahkan koneksi antara kota, lengkap dengan jaraknya

1. buatkanRute(k)



Menghitung rute dari setiap kota ke kota lain, agar ketika kita cari rute, hasilnya sudah tersedia.

1. Input interaktif



Memungkinkan pengguna mencari rute tercepat dari satu kota ke kota lain sampai mereka ketik “exit”.

Tampilan Output:

