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

```python
class JalurTercepatIndonesia:
    def __init__(self):
        self.daftarKota = {}
        self.jumlahKota = 0
        self.peta = {}

    def tambahkanKota(self, kota):
        if kota not in self.daftarKota:
            self.daftarKota[kota] = {}
            self.jumlahKota += 1

    def tambahkanJalan(self, kota1, cities):
        for kota in cities:
            if kota1 in self.daftarKota and kota in self.daftarKota:
                self.daftarKota[kota1][kota] = cities[kota]
                self.daftarKota[kota][kota1] = cities[kota]

    def tampilkanKota(self):
        print("Daftar Kota:")
        for kota in self.daftarKota:
            print("-", kota)
        print(f"Jumlah Kota: {self.jumlahKota}")

    def tampilkanPeta(self):
        print("\nPeta:")
        for i, kota in enumerate(self.peta):
            print(f"{i+1}. {kota}")
            for kotaTetangga, rute in self.peta[kota].items():
                print(f"   --> {kotaTetangga}:")
                for jalan in rute.values():
                    print(f"      --> {jalan} km")

```

```python
    def buatkanRute(self, start):
        peta = {}
        distances, routes = self.dijkstra(start)
        for kota, jarak in distances.items():
            peta[kota] = {
                "rute": self.route(distances, routes, start, kota),
                "jarak": jarak
            }
        self.peta[start] = peta

    def dijkstra(self, start):
        unvisited = list(self.daftarKota.keys())
        distances = {city: float('inf') for city in self.daftarKota}
        routes = {}
        distances[start] = 0

        while unvisited:
            current = min(unvisited, key=lambda city: distances[city])
            unvisited.remove(current)

            for tetangga, jarak in self.daftarKota[current].items():
                total_jarak = distances[current] + jarak
                if total_jarak < distances[tetangga]:
                    distances[tetangga] = total_jarak
                    routes[tetangga] = current

        del distances[start]
        return distances, routes
```

```python
    def route(self, distances, routes, start, target):
        path = {}
        kota = target
        while kota != start:
            if routes[kota] == start:
                path[kota] = distances[kota]
            else:
                path[kota] = round(distances[kota] - distances[routes[kota]], 1)
            kota = routes[kota]
        return path

    def cariRuteTercepat(self, dari, ke):
        dari = dari.title()
        ke = ke.title()
        if dari in self.daftarKota and ke in self.daftarKota:
            print(f"\nDari {dari} ke {ke}, berjarak {self.peta[dari][ke]['jarak']} km")
            print("Dengan rute:")
            for jalan, jarak in reversed(self.peta[dari][ke]['rute'].items()):
                print(f"\t ---> {jalan} ({jarak} km)")
        else:
            print(f"{dari} atau {ke} tidak berada di daftar kota")


# === Daftar Kota & Edge (30 edge, 10 kota) ===
kota = [
    "Jakarta", "Bandung", "Surabaya", "Semarang", "Yogyakarta",
    "Medan", "Palembang", "Makassar", "Bali", "Pontianak"
]
```

```
peta = JalurTercepatIndonesia()

for k in kota:
    peta.tambahkanKota(k)

# Tambahkan 30 jalur antar kota dengan jarak (km)
peta.tambahkanJalan("Jakarta", {"Bandung": 150, "Semarang": 450, "Palembang": 600})
peta.tambahkanJalan("Bandung", {"Yogyakarta": 390, "Surabaya": 680, "Pontianak": 870})
peta.tambahkanJalan("Semarang", {"Yogyakarta": 120, "Surabaya": 320, "Makassar": 1050})
peta.tambahkanJalan("Yogyakarta", {"Bali": 550, "Palembang": 700, "Makassar": 1180})
peta.tambahkanJalan("Surabaya", {"Bali": 430, "Makassar": 950, "Medan": 1800})
peta.tambahkanJalan("Medan", {"Palembang": 920, "Pontianak": 1100, "Bali": 1720})
peta.tambahkanJalan("Palembang", {"Pontianak": 980, "Bali": 960})
peta.tambahkanJalan("Makassar", {"Bali": 780, "Jakarta": 1350})
peta.tambahkanJalan("Jakarta", {"Pontianak": 850, "Bali": 950})

# Buat rute semua kota
for k in kota:
    peta.buatkanRute(k)

# Tampilkan daftar kota dan peta
peta.tampilkanKota()
peta.tampilkanPeta()
```

```
# Interaktif cari rute tercepat
print("\n=== CARI RUTE TERCEPAT ===")
while True:
    dari = input("Cari Rute Tercepat dari (exit untuk keluar): ")
    if dari.lower() == "exit":
        break
    ke = input("Ke kota: ")
    if ke.lower() == "exit":
        break
    peta.cariRuteTercepat(dari, ke)
    print()
```

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)

2. __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.

3. tambahkanKota(self, kota)

Menambahkan nama kota ke dalam daftar jika belum ada

4. tambahkanJalan(self, kota1, cities)

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

```
# Tambahkan 30 jalur antar kota dengan jarak (km)
peta.tambahkanJalan("Jakarta", {"Bandung": 150, "Semarang": 450, "Palembang": 600})
```

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

5. tampilkanKota(self)

Menampilkan seluruh kota yang sudah ditambahkan dan jumlahnya

6. tampilkanPeta(self)

Menampilkan isi dari peta:

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

Outpunya akan seperti ini

```
1. Jakarta
  --> Bandung:
     --> {'Bandung': 150} km
     --> 150 km
```

7. buatkanRute(self, start)

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

8. 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.

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

Membangun rute dari start ke target berdasarkan hasil Dijkstra.

10. 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:

```
=== CARI RUTE TERCEPAT ===
Cari Rute Tercepat dari (exit untuk keluar): jakarta
Ke kota: surabaya

Dari Jakarta ke Surabaya, berjarak 770 km
Dengan rute:
         ---> Semarang (450 km)
         ---> Surabaya (320 km)
```

11. Daftar kota

```
kota = [
    "Jakarta", "Bandung", "Surabaya", "Semarang", "Yogyakarta",
    "Medan", "Palembang", "Makassar", "Bali", "Pontianak"
]
```

Kumpulan nama kota di Indonesia sebagai simpul dalam graf.

12. Loop tambah kota

```
for k in kota:
    peta.tambahkanKota(k)
```

Menambahkan semua kota ke dalam objek peta

13. Penambahan 30 jalur (edge)

```
# Tambahkan 30 jalur antar kota dengan jarak (km)
peta.tambahkanJalan("Jakarta", {"Bandung": 150, "Semarang": 450, "Palembang": 600})
peta.tambahkanJalan("Bandung", {"Yogyakarta": 390, "Surabaya": 680, "Pontianak": 870})
peta.tambahkanJalan("Semarang", {"Yogyakarta": 120, "Surabaya": 320, "Makassar": 1050})
peta.tambahkanJalan("Yogyakarta", {"Bali": 550, "Palembang": 700, "Makassar": 1180})
peta.tambahkanJalan("Surabaya", {"Bali": 430, "Makassar": 950, "Medan": 1800})
peta.tambahkanJalan("Medan", {"Palembang": 920, "Pontianak": 1100, "Bali": 1720})
peta.tambahkanJalan("Palembang", {"Pontianak": 980, "Bali": 960})
peta.tambahkanJalan("Makassar", {"Bali": 780, "Jakarta": 1350})
peta.tambahkanJalan("Jakarta", {"Pontianak": 850, "Bali": 950})
```

Menambahkan koneksi antara kota, lengkap dengan jaraknya

14. buatkanRute(k)

```
# Buat rute semua kota
for k in kota:
    peta.buatkanRute(k)
```

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

15. Input interaktif

```
# Interaktif cari rute tercepat
print("\n=== CARI RUTE TERCEPAT ===")
while True:
    dari = input("Cari Rute Tercepat dari (exit untuk keluar): ")
    if dari.lower() == "exit":
        break
    ke = input("Ke kota: ")
    if ke.lower() == "exit":
        break
    peta.cariRuteTercepat(dari, ke)
    print()
```

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

Tampilan Output:

```
Daftar Kota:
- Jakarta
- Bandung
- Surabaya
- Semarang
- Yogyakarta
- Medan
- Palembang
- Makassar
- Bali
- Pontianak
Jumlah Kota: 10

Peta:
1. Jakarta
  --> Bandung:
     --> {'Bandung': 150} km
     --> 150 km
  --> Surabaya:
     --> {'Surabaya': 320, 'Semarang': 450} km
     --> 770 km
  --> Semarang:
     --> {'Semarang': 450} km
     --> 450 km
  --> Yogyakarta:
     --> {'Yogyakarta': 390, 'Bandung': 150} km
     --> 540 km
```

```
  --> Medan:
     --> {'Medan': 920, 'Palembang': 600} km
     --> 1520 km
  --> Palembang:
     --> {'Palembang': 600} km
     --> 600 km
  --> Makassar:
     --> {'Makassar': 1350} km
     --> 1350 km
  --> Bali:
     --> {'Bali': 950} km
     --> 950 km
  --> Pontianak:
     --> {'Pontianak': 850} km
     --> 850 km
2. Bandung
  --> Jakarta:
     --> {'Jakarta': 150} km
     --> 150 km
  --> Surabaya:
     --> {'Surabaya': 680} km
     --> 680 km
  --> Semarang:
     --> {'Semarang': 120, 'Yogyakarta': 390} km
     --> 510 km
  --> Yogyakarta:
     --> {'Yogyakarta': 390} km
     --> 390 km
  --> Medan:
     --> {'Medan': 920, 'Palembang': 600, 'Jakarta': 150} km
     --> 1670 km
  --> Palembang:
     --> {'Palembang': 600, 'Jakarta': 150} km
     --> 750 km
  --> Makassar:
     --> {'Makassar': 1350, 'Jakarta': 150} km
     --> 1500 km
```

```
    --> Bali:
        --> {'Bali': 550, 'Yogyakarta': 390} km
        --> 940 km
    --> Pontianak:
        --> {'Pontianak': 870} km
        --> 870 km
3. Surabaya
    --> Jakarta:
        --> {'Jakarta': 450, 'Semarang': 320} km
        --> 770 km
    --> Bandung:
        --> {'Bandung': 680} km
        --> 680 km
    --> Semarang:
        --> {'Semarang': 320} km
        --> 320 km
    --> Yogyakarta:
        --> {'Yogyakarta': 120, 'Semarang': 320} km
        --> 440 km
    --> Medan:
        --> {'Medan': 1800} km
        --> 1800 km
    --> Palembang:
        --> {'Palembang': 700, 'Yogyakarta': 120, 'Semarang': 320} km
        --> 1140 km
    --> Makassar:
        --> {'Makassar': 950} km
        --> 950 km
    --> Bali:
        --> {'Bali': 430} km
        --> 430 km
    --> Pontianak:
        --> {'Pontianak': 870, 'Bandung': 680} km
        --> 1550 km
```

```
4. Semarang
   --> Jakarta:
      --> {'Jakarta': 450} km
      --> 450 km
   --> Bandung:
      --> {'Bandung': 390, 'Yogyakarta': 120} km
      --> 510 km
   --> Surabaya:
      --> {'Surabaya': 320} km
      --> 320 km
   --> Yogyakarta:
      --> {'Yogyakarta': 120} km
      --> 120 km
   --> Medan:
      --> {'Medan': 920, 'Palembang': 700, 'Yogyakarta': 120} km
      --> 1740 km
   --> Palembang:
      --> {'Palembang': 700, 'Yogyakarta': 120} km
      --> 820 km
   --> Makassar:
      --> {'Makassar': 1050} km
      --> 1050 km
   --> Bali:
      --> {'Bali': 550, 'Yogyakarta': 120} km
      --> 670 km
   --> Pontianak:
      --> {'Pontianak': 850, 'Jakarta': 450} km
      --> 1300 km
```

```
5. Yogyakarta
   --> Jakarta:
      --> {'Jakarta': 150, 'Bandung': 390} km
      --> 540 km
   --> Bandung:
      --> {'Bandung': 390} km
      --> 390 km
   --> Surabaya:
      --> {'Surabaya': 320, 'Semarang': 120} km
      --> 440 km
   --> Semarang:
      --> {'Semarang': 120} km
      --> 120 km
   --> Medan:
      --> {'Medan': 920, 'Palembang': 700} km
      --> 1620 km
   --> Palembang:
      --> {'Palembang': 700} km
      --> 700 km
   --> Makassar:
      --> {'Makassar': 1050, 'Semarang': 120} km
      --> 1170 km
   --> Bali:
      --> {'Bali': 550} km
      --> 550 km
   --> Pontianak:
      --> {'Pontianak': 870, 'Bandung': 390} km
      --> 1260 km
```

```
6. Medan
   --> Jakarta:
      --> {'Jakarta': 600, 'Palembang': 920} km
      --> 1520 km
   --> Bandung:
      --> {'Bandung': 150, 'Jakarta': 600, 'Palembang': 920} km
      --> 1670 km
   --> Surabaya:
      --> {'Surabaya': 1800} km
      --> 1800 km
   --> Semarang:
      --> {'Semarang': 120, 'Yogyakarta': 700, 'Palembang': 920} km
      --> 1740 km
   --> Yogyakarta:
      --> {'Yogyakarta': 700, 'Palembang': 920} km
      --> 1620 km
   --> Palembang:
      --> {'Palembang': 920} km
      --> 920 km
   --> Makassar:
      --> {'Makassar': 780, 'Bali': 1720} km
      --> 2500 km
   --> Bali:
      --> {'Bali': 1720} km
      --> 1720 km
   --> Pontianak:
      --> {'Pontianak': 1100} km
      --> 1100 km
```

```
7. Palembang
  --> Jakarta:
     --> {'Jakarta': 600} km
     --> 600 km
  --> Bandung:
     --> {'Bandung': 150, 'Jakarta': 600} km
     --> 750 km
  --> Surabaya:
     --> {'Surabaya': 320, 'Semarang': 120, 'Yogyakarta': 700} km
     --> 1140 km
  --> Semarang:
     --> {'Semarang': 120, 'Yogyakarta': 700} km
     --> 820 km
  --> Yogyakarta:
     --> {'Yogyakarta': 700} km
     --> 700 km
  --> Medan:
     --> {'Medan': 920} km
     --> 920 km
  --> Makassar:
     --> {'Makassar': 780, 'Bali': 960} km
     --> 1740 km
  --> Bali:
     --> {'Bali': 960} km
     --> 960 km
  --> Pontianak:
     --> {'Pontianak': 980} km
     --> 980 km
```

```
8. Makassar
   --> Jakarta:
      --> {'Jakarta': 1350} km
      --> 1350 km
   --> Bandung:
      --> {'Bandung': 150, 'Jakarta': 1350} km
      --> 1500 km
   --> Surabaya:
      --> {'Surabaya': 950} km
      --> 950 km
   --> Semarang:
      --> {'Semarang': 1050} km
      --> 1050 km
   --> Yogyakarta:
      --> {'Yogyakarta': 120, 'Semarang': 1050} km
      --> 1170 km
   --> Medan:
      --> {'Medan': 1720, 'Bali': 780} km
      --> 2500 km
   --> Palembang:
      --> {'Palembang': 960, 'Bali': 780} km
      --> 1740 km
   --> Bali:
      --> {'Bali': 780} km
      --> 780 km
   --> Pontianak:
      --> {'Pontianak': 850, 'Jakarta': 1350} km
      --> 2200 km
```

```
9. Bali
  --> Jakarta:
     --> {'Jakarta': 950} km
     --> 950 km
  --> Bandung:
     --> {'Bandung': 390, 'Yogyakarta': 550} km
     --> 940 km
  --> Surabaya:
     --> {'Surabaya': 430} km
     --> 430 km
  --> Semarang:
     --> {'Semarang': 120, 'Yogyakarta': 550} km
     --> 670 km
  --> Yogyakarta:
     --> {'Yogyakarta': 550} km
     --> 550 km
  --> Medan:
     --> {'Medan': 1720} km
     --> 1720 km
  --> Palembang:
     --> {'Palembang': 960} km
     --> 960 km
  --> Makassar:
     --> {'Makassar': 780} km
     --> 780 km
  --> Pontianak:
     --> {'Pontianak': 850, 'Jakarta': 950} km
     --> 1800 km
```

```
10. Pontianak
  --> Jakarta:
     --> {'Jakarta': 850} km
     --> 850 km
  --> Bandung:
     --> {'Bandung': 870} km
     --> 870 km
  --> Surabaya:
     --> {'Surabaya': 680, 'Bandung': 870} km
     --> 1550 km
  --> Semarang:
     --> {'Semarang': 450, 'Jakarta': 850} km
     --> 1300 km
  --> Yogyakarta:
     --> {'Yogyakarta': 390, 'Bandung': 870} km
     --> 1260 km
  --> Medan:
     --> {'Medan': 1100} km
     --> 1100 km
  --> Palembang:
     --> {'Palembang': 980} km
     --> 980 km
  --> Makassar:
     --> {'Makassar': 1350, 'Jakarta': 850} km
     --> 2200 km
  --> Bali:
     --> {'Bali': 950, 'Jakarta': 850} km
     --> 1800 km
```

```
=== CARI RUTE TERCEPAT ===
Cari Rute Tercepat dari (exit untuk keluar): jakarta
Ke kota: surabaya

Dari Jakarta ke Surabaya, berjarak 770 km
Dengan rute:
         ---> Semarang (450 km)
         ---> Surabaya (320 km)
```