Nama : Wita Adelia

Nim : 20220046

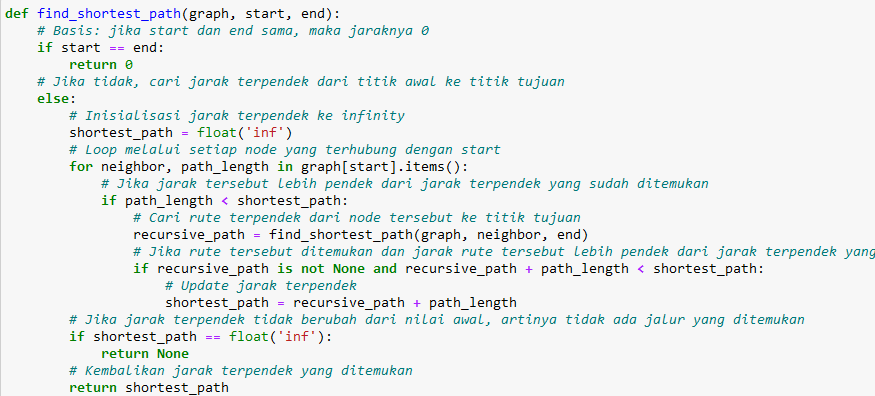
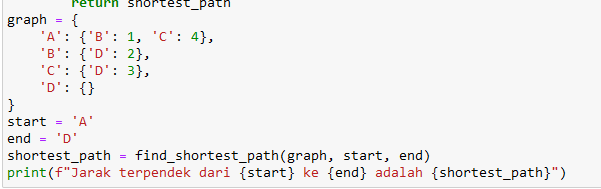
Matkul : Prak.DAA

Divide and Conquer

Tugas 1.

**Fungsi untuk mencari rute terpendek dari titik awal ke titik tujuan**

Contoh input data menggunakan uji coba Jupyter



Dengan Hasil Output



Pada kode di atas, kita menggunakan pendekatan Divide and Conquer untuk mencari

rute terpendek dari titik awal ke titik tujuan dalam sebuah grafik (graph). Kita memulai

dari titik awal dan melakukan iterasi melalui setiap node yang terhubung dengan titik

awal.

Untuk setiap node yang terhubung, kita menghitung jarak dari titik awal ke node tersebut

dan kemudian mencari rute terpendek dari node tersebut ke titik tujuan dengan

menggunakan pendekatan rekursif. Jika jarak rute terpendek yang ditemukan lebih

pendek dari jarak terpendek yang sudah ditemukan sebelumnya, kita mengupdate jarak

terpendek tersebut.

**Latihan :**

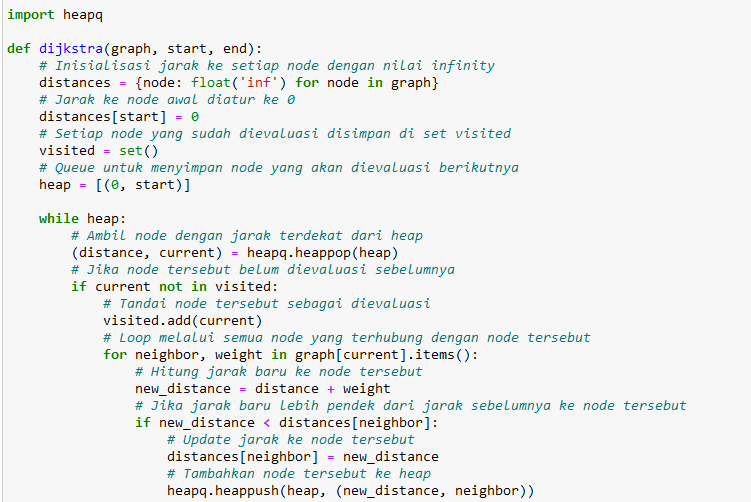
contoh kode python untuk mengetahui jarak terpendek dari node A ke Node E, jika

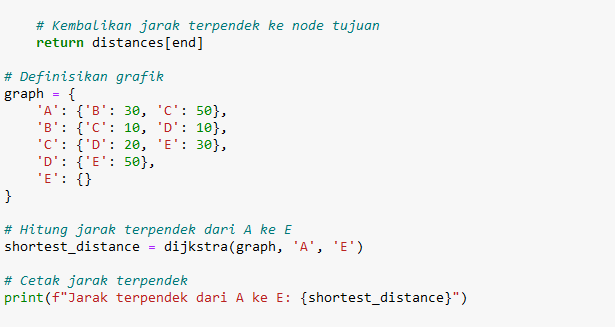
diketahui : jarak node A ke B =30 jarak node A ke C = 50 jarak node B ke C =10 jarak

node C ke D = 40 jarak node dari B ke D = 10 jarak dari node C ke D = 20 jarak dari node

C ke E = 30 jarak dari node D ke E = 50

Uji coba kode python ini mengunakan Jupyter dengan input nilai yang ada di atas.





Dengan Hasil Output Nilai.

Dengan Jarak Terpendek dari A ke E adalah 70



Ini termasuk kode Python untuk menjalankan algoritma Dijkstra, sebuah algoritma untuk mencari jarak terpendek pada grafik berbobot positif tunggal. Namun, baris pertama pada kode di atas yaitu import heapq adalah kode untuk mengimpor modul heapq, yang digunakan di dalam fungsi Dijkstra untuk melakukan operasi heap pada data. Heapq adalah modul dalam Python yang menyediakan implementasi struktur data heap untuk data yang dapat diubah secara dinamis. Dalam konteks Dijkstra, struktur heap digunakan untuk memprioritaskan simpul-simpul yang belum dikunjungi yang memiliki jarak terpendek ke simpul awal (start node) saat ini.

1. Pertama-tama, kita melakukan import library heapq, yang digunakan untuk membuat priority queue.
2. Kemudian, kita mendefinisikan fungsi dijkstra yang akan mengambil tiga parameter: graph (graf yang akan dicari jarak terpendeknya), start (node awal), dan end (node tujuan).
3. Pertama-tama, kita inisialisasi jarak ke setiap node dengan nilai infinity, kecuali untuk node awal yang jaraknya adalah 0.
4. Kemudian, kita menyimpan node-node yang belum dikunjungi dalam priority queue yang diurutkan berdasarkan jaraknya dari node awal.
5. Selama ada node yang belum dikunjungi, kita ambil node yang jaraknya terpendek dan loop melalui setiap node yang terhubung dengannya.
6. Untuk setiap tetangga yang belum dikunjungi, kita hitung jarak baru dari node awal melalui node saat ini ke tetangganya. Jika jarak tersebut lebih pendek dari jarak yang
7. sudah ada ke tetangganya, maka kita update jarak terpendek ke tetangganya dan tambahkan tetangga ke daftar node yang belum dikunjungi.
8. Setelah semua node dikunjungi, kita kembalikan j