STURUKTUR DATA

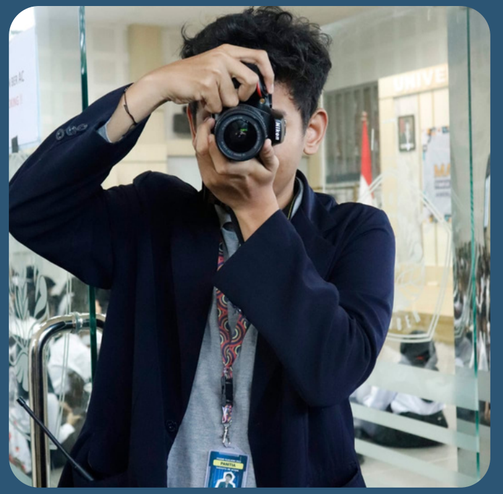
Aplikasi Penjadwalan TransJakarta

KELOMPOK 30

https://youtu.be/Z_ZNWpj6IcU

Farell Bimo Seno
222410103004

Muhammad Ainol. Y
222410103090



Andhik Rachmat Fauzi
222410103012

KELOMPOK. 30

Learn About

- Aplikasi ini dirancang dengan tujuan utama untuk memberikan informasi jadwal keberangkatan bus serta data terkini mengenai bus yang sudah berangkat. Dengan fitur-fitur yang disediakan, pengguna dapat dengan mudah mengakses informasi terkini mengenai waktu keberangkatan bus, dan juga mendapatkan data aktual mengenai bus yang sudah berangkat, memberikan kenyamanan dan kemudahan bagi para pengguna yang ingin merencanakan perjalanan menggunakan transportasi bus. Dengan demikian, aplikasi ini menjadi solusi praktis bagi mereka yang mengandalkan bus sebagai sarana transportasi utama.

Struktur Data

Hash table merupakan struktur data yang secara asosiatif menyimpan data. Dalam hal ini, data disimpan dalam format array, di mana setiap nilai data memiliki nilai indeks uniknya sendiri. Akses data akan menjadi sangat cepat jika Anda mengetahui indeks dari data yang diinginkan.

Dengan demikian, hash table menjadi struktur data di mana operasi penyisipan dan pencarian data terjadi sangat cepat terlepas dari ukuran data tersebut. Hash table menggunakan array sebagai media penyimpanan dan tekniknya untuk menghasilkan indeks suatu elemen yang dimasukkan atau ditempatkan.

Struktur data queue merupakan sebuah struktur data linier dengan prinsip operasi yang mewajibkan elemen data masuk pertama untuk keluar lebih dulu. Prinsip ini juga sering dikenal dengan sebutan FIFO atau First In First Out.

Struktur data queue sangat berbeda dengan struktur data stack yang melakukan penyimpanan data secara bertumpung dengan satu ujung terbuka untuk operasi data. Data pada struktur data queue disusun secara horizontal dan terbuka di kedua ujungnya, dimana ujung pertama untuk menghapus data dan ujung lainnya untuk menyisipkan data.

SOURCE CODE

Struktur data queue

```
class Queue:
    def __init__(self):
        self.items = []

    def is_empty(self):
        return self.items == []

    def enqueue(self, item):
        self.items.insert(0, item)

    def dequeue(self):
        if not self.is_empty():
            return self.items.pop()

    def size(self):
        return len(self.items)
```

Hash table

```
def update_status_in_csv(self, bus_number):
        with open(self.csv_filename, mode='r') as file:
            rows = list(csv.DictReader(file))
            for row in rows:
                if row["KodeBus"] == bus_number:
                    row["Status"] = "Departed"
        with open(self.csv_filename, mode='w', newline='') as file:
            fieldnames = ["KodeBus", "Status"]
            writer = csv.DictWriter(file, fieldnames=fieldnames)
            writer.writeheader()
            writer.writerows(rows)
```

```
    def update_schedule_to_csv(self):
        with open(self.csv_filename, mode='w', newline='') as file:
            Kodebus = ["KodeBus"]
            writer = csv.DictWriter(file, fieldnames=Kodebus)
            writer.writeheader()
            for bus in self.schedule.items:
                writer.writerow({"KodeBus": bus})
```

Thank You!

Kelompok 30

Public Transportation in Jakarta