Aplikasi Sistem Informasi dan Analisis Obat dengan Benchmarking ANALISIS KOMPLEKSITAS ALGORITMA



ANGGOTA KELOMPOK:

NUR AHMADI ADITYA NANDA (103022300149)

ADE FATHIA NURAINI (103022300134)

DOSEN PENGAMPU:

LIDYA NINGSIH, S.T, M.T

TELKOM UNIVERSITY
PRODI S1 REKAYASA PERANGKAT LUNAK
2024

DAFTAR ISI

Pendahuluan	3
Deskripsi Studi Kasus Pendahuluan	4
1. Database Obat	4
2. Fitur Utama	4
A. Pencarian Obat	4
B. Analisis Struktur Kimia	5
C. Analis Pembelian Obat	5
D. Benchmark Performance Algorithm	6

PENDAHULUAN

Aplikasi ini merupakan alat untuk pencarian dan analisis data obat, dilengkapi fitur benchmarking performa metode iteratif dan rekursif. Dengan antarmuka sederhana dan visualisasi data, aplikasi ini mempermudah manajemen dan evaluasi data obat untuk kebutuhan akademik atau operasional. aplikasi ini dilengkapi dengan mekanisme perbandingan performa pencarian menggunakan metode **iteratif** dan **rekursif**. Benchmark ini divisualisasikan dalam bentuk diagram garis, memungkinkan pengguna memahami efisiensi kedua metode berdasarkan ukuran data yang diuji.

DESKRIPSI STUDI KASUS PERMASALAHAN

Aplikasi ini dirancang untuk menjawab tantangan tersebut melalui implementasi metode iteratif dan rekursif untuk pencarian obat, serta menampilkan performa masing-masing metode dalam bentuk visualisasi. Dengan demikian, aplikasi ini membantu pengguna memilih pendekatan terbaik untuk pengelolaan data yang optimal.

1. Database Obat

- Tujuan: Menyimpan data obat berupa nama, harga, dan struktur kimia.
- Format: Menggunakan list of dictionaries di Python.
- **Skalabilitas**: Database ditambah dengan data dummy (2000 entri) untuk pengujian performa algoritma.

```
# Database obat

obat_database = [

{"nama": "Paracetamol", "harga": 5000, "struktur": "C8H9NO2"},

{"nama": "Ibuprofen", "harga": 15000, "struktur": "C13H1802"},

{"nama": "Amoxicillin", "harga": 20000, "struktur": "C16H19N3O5S"},

]
```

2. Fitur Utama

A. Pencarian Obat

Metode yang Digunakan:

1. Iteratif

- Melakukan pencarian dengan looping.
- o Efisien untuk ukuran data kecil hingga sedang.

```
# Fungsi iteratif untuk pencarian obat

def cari_obat_iteratif(keyword):

return [obat for obat in obat_database if keyword.lower() in obat["nama"].lower()]
```

2. Rekursif

- Melakukan pencarian dengan pendekatan berbasis fungsi yang memanggil dirinya sendiri.
- Kurang optimal untuk data besar karena membutuhkan memori lebih.

```
# Fungsi rekursif untuk pencarian obat
def cari_obat_rekursif(keyword, index=0, hasil=None):
    if hasil is None:
        hasil = []
    if index >= len(obat_database):
        return hasil
    if keyword.lower() in obat_database[index]["nama"].lower():
        hasil.append(obat_database[index])
    return cari_obat_rekursif(keyword, index + 1, hasil)
```

B. Analisis Struktur Kimia

- **Tujuan**: Menampilkan struktur kimia obat yang ditemukan dalam bentuk hierarki atom.
- **Metode**: Rekursif untuk mengekstrak dan menganalisis setiap atom dalam string struktur kimia.

C. Analisis Pembelian Obat

- **Tujuan**: Mengidentifikasi pola pembelian obat dari berbagai bulan.
- **Visualisasi**: Diagram batang dan garis menggunakan **Matplotlib** untuk menunjukkan total belanja per bulan.

```
Atom: C, Level: 0
Atom: 8, Level: 1
Atom: H, Level: 2
Atom: 9, Level: 3
```

D. Benchmark Performa Algoritma

Tujuan: Membandingkan performa metode pencarian iteratif dan rekursif pada ukuran data yang berbeda.

Proses:

- Ukuran dataset yang diuji: 100, 500, 1000, 1500, dan 2000 entri.
- Waktu eksekusi diukur menggunakan modul time.perf_counter.
- Hasil: Ditampilkan dalam diagram garis yang menunjukkan waktu eksekusi (dalam detik) terhadap ukuran data.

```
def visualisasi_benchmark():
    keyword = entry_keyword.get()
    # Ukuran data untuk benchmark
    sizes = [100, 500, 1000, 1500, 2000]
    waktu_iteratif, waktu_rekursif = benchmark_cari_obat(keyword, sizes)
    # Visualisasi dengan diagram garis
    plt.figure(figsize=(10, 6))
    plt.plot(sizes, waktu_rekursif, marker='o', label='Rekursif', color='orange')
    plt.title("Perbandingan Waktu Eksekusi")
    plt.xlabel("Ukuran Data")
    plt.ylabel("Waktu (detik)")
    plt.legend()
    plt.grid(True)
    for i, size in enumerate(sizes):
       plt.text(size, waktu_iteratif[i], f"{waktu_iteratif[i]:.4f}", ha='right', color='blue')
       plt.text(size, waktu_rekursif[i], f"{waktu_rekursif[i]:.4f}", ha='right', color='orange')
    plt.tight_layout()
```

3. Antarmuka Pengguna (GUI)

Dibangun menggunakan modul **Tkinter** untuk memudahkan interaksi pengguna.

Komponen GUI:

1. Input Kata Kunci:

o Entry untuk memasukkan nama obat yang dicari.

2. Tombol Fungsi:

- Cari Obat (Iteratif): Memulai pencarian menggunakan metode iteratif.
- Cari Obat (Rekursif): Memulai pencarian menggunakan metode rekursif.
- Benchmark dan Visualisasi: Menampilkan hasil benchmarking algoritma dalam bentuk diagram garis.
- Analisis Struktur Kimia: Menganalisis struktur kimia dari obat yang ditemukan.
- Analisis Pembelian Obat: Menampilkan diagram pola pembelian obat.

Tampilan GUI:

Antarmuka sederhana dengan tata letak yang terorganisir menggunakan frame dan grid layout.

4. Studi Kasus dan Implementasi

Studi Kasus:

- Pencarian obat berdasarkan nama, seperti Paracetamol.
- Analisis struktur kimia obat tertentu.
- Mengukur performa algoritma pencarian pada dataset besar.

Hasil Implementasi:

- Memberikan hasil pencarian secara cepat dengan visualisasi waktu eksekusi.
- Membantu pengguna memilih algoritma terbaik untuk dataset besar.

5. Kesimpulan

Aplikasi ini adalah alat yang efektif untuk pencarian dan analisis data obat dengan evaluasi performa algoritma yang disertai visualisasi. Program ini cocok digunakan untuk mendukung pengelolaan data obat secara praktis dan efisien.

REFERENSI

Dokumentasi Tkinter

- · Referensi untuk membangun antarmuka GUI.
- https://docs.python.org/3/library/tkinter.html

Dokumentasi Matplotlib

- Referensi untuk membuat visualisasi data.
- https://matplotlib.org/stable/contents.html

Penggunaan List dan Dictionary di Python

- Penjelasan tentang struktur data yang digunakan untuk database.
- https://docs.python.org/3/tutorial/datastructures.html

Rekursi dalam Python

- Panduan untuk memahami cara kerja fungsi rekursif.
- https://realpython.com/python-recursion/

Time Complexity of Algorithms

- Untuk memahami analisis efisiensi algoritma iteratif dan rekursif.
- https://www.bigocheatsheet.com/

Python Performance Benchmarking

- Mengukur performa kode menggunakan modul time.
- https://docs.python.org/3/library/time.html