LAPORAN TUGAS BESAR
MANAGEMENT ACTIVITY LIBRARY SYSTEM BERBASIS WEBSITE

Disusun untuk memenuhi Tugas Mata Kuliah
Analisis Kompleksitas Algoritma

BERTRAND LIANTO	(103072400019)
BIMA LUTHFI NURHAKIM	(103072400030)
AHMAD NUR FAJRI	(103072430007)

Program Studi S1 Informatika
Fakultas Informatika
Universitas Telkom

2025

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan Hidayah-nya sehingga kami dapat menyelesaikan tugas laporan tentang "Management Activity Library System Berbasis Website" dengan deskripsi singkat yaitu "Dapat melakukan manajemen perpustakaan sekolah/kampus sehingga dapat memantau kinerja aktivitas perpustakaan dengan menggunakan program berbasis Website dengan Kompleksitas Algoritma menggunakan metode analysis seperti rekursif ataupun iterative. implementasi ini diharapkan dapat memudahkan user bisa juga disebut friendly user dalam implementasi programnya di lapangan langsung."

Adapun tujuan dari penulisan laporan ini adalah untuk memenuhi tugas pada mata kuliah Analisis Kompleksitas Algoritma Semester III dengan dosen pengampu Afifah Mutiara Pertiwi, S.Kom., M.Cs. Tidak lupa kami sampaikan terima kasih kepada dosen pengampu mata kuliah Analisis Kompleksitas Algoritma yang memberikan arahan dan bimbingan dalam pembuatan laporan ini dan orang tua yang selalu mendukung kelancaran tugas kami.

Akhirnya, kami sampaikan terima kasih atas perhatiannya terhadap laporan ini, kami menyadari bahwa tugas yang ditulis ini masih jauh dari kata sempurna. Oleh karena itu, kritik dan saran yang membangun kami butuhkan demi kesempurnaan laporan kami ini.

Surabaya, 18 Desember 2025

Penulis

DAFTAR ISI

BAB I PENDAHULUAN

1.1 Latar Belakang

Saat ini, Cara mengelola perpustakaan di kampus atau di dunia pendidikan bahkan umum sudah jauh berbeda karena teknologi digital saat ini. Sistem manual yang dulu sering membuat antrian yang sangat panjang atau pendataan yang berantakan sudah mulai ditinggalkan. Sebagai gantinya, diperlukannya sebuah sistem yang mudah diakses, seperti Management Activity Library System berbasis Website. Platform ini dibuat agar pihak perpustakaan bisa memantau semua aktivitas dengan lebih mudah, mulai dari pendataan buku hingga memantau kinerja perpustakaan. Disini tujuan utamanya adalah membuat pekerjaan admin jadi lebih ringan dan memberikan pengalaman yang nyaman bagi para siswa maupun umum untuk membaca buku untuk menambah wawasan mereka.

Membahas tentang platform yang dibuat, terdapat beberapa bagian penting yang menentukan seberapa cepat sistem ini bekerja atau bisa disebut algoritma. Salah satu fitur yang paling sering digunakan dalam platform perpustakaan adalah fitur searching. Agar pencarian buku tidak memakan waktu lama, biasanya saat jumlah datanya sudah mencapai hingga puluh ribuan, diperlukannya metode pemrograman yang paling efisien. Mungkin disini kegunaan dari Analisis Kompleksitas Algoritma menjadi sangat penting. Disini tidak hanya membuat fitur searching bisa berjalan, tetapi juga harus melihat bahwa cara kerja di baliknya benar - benar optimal dan tidak membebani server.

Selanjutnya untuk pembahasan lebih luas ada dua cara utama dalam membuat fitur searching, yaitu metode iterasi dan metode rekursi. Kedua ini punya kelebihan dan kekurangannya masing - masing saat diterapkan pada sistem platform perpustakaan berbasis web. Disini akan dilakukannya perbandingan secara langsung bagaimana metode iterasi dan rekursi ini bekerja, mulai dari cara penulisan code hingga bagaimana performanya saat menangani data yang besar. Perbandingan ini sangat berguna untuk menentukan mana yang lebih cocok digunakan agar platform website tetap lancar dan tidak terjadi masalah seperti misal crash dan lain sebagainya.

Untuk pembahasan selanjutnya adalah tentang kompleksitas waktu dari masing - masing metode. Untuk melihat sejauh mana platform ini berkembang jadi tidak hanya memiliki fitur yang lengkap tetapi juga friendly user atau mudah digunakan oleh siapa saja. Dengan begini

bisa dilihat bahwa platform dari Management Activity Library System akan tetap stabil dan responsif meskipun digunakan oleh banyak user secara bersamaan. Dengan analisis ini, dapat menjadi bukti bahwa sistem yang dibangun bisa menjadi solusi yang cerdas dan efesien untuk manajemen perpustakaan modern.

1.2 Maksud Dan Tujuan

Tujuan utama adalah untuk mengetahui dan mengimplementasikan bagaimana cara pemilihan logika pemrograman antara metode iterasi dan rekursi, sangat berpengaruh terhadap performa sistem manajemen perpustakaan berbasis website. Dengan analisis ini, dapat mengetahui bahwa fitur searching data tidak hanya cuma berfungsi, melainkan juga bisa berjalan dengan efesiensi maksimal sehingga tidak membebani sumber daya server. Selain itu development sistem ini bertujuan untuk membuat alur kerja digital yang lebih terstruktur, memudahkan pengembang dalam mengoptimalkan fitur, dan juga kenyamanan user dalam berinteraksi dengan data. Adapun tujuan yang ingin dicapai adalah sebagai berikut.

1. Untuk menguji perbedaaan kecepatan dan waktu dalam memproses searching menggunakan iterasi dan rekursi.
2. Untuk mengetahui algoritma terbaik untuk suatu fitur dalam perangkat lunak.
3. Untuk mengelola kumpulan buku online agar lebih mudah dicari oleh pengguna dan diakses.
4. Untuk memberitahukan informasi mengenai perpustakaan kepada pengguna secara online.

1.3 Rumusan Masalah

Kebutuhan sistem manajemen perpustakaan yang cepat dan responsif menimbulkan mungkin pertanyaan tentang efektivitas pemrograman yang digunakan di balik fitur searching data buku. Masalah utamanya adalah bagaimana mengimplementasikan algoritma searching dalam dua metode ada iteratif dan rekursif ke dalam sebuah sistem manajemen perpustakaan berbasis website agar dapat menangani data dalam jumlah besar secara optimal. Dan ada beberapa algoritma paling efisien seperti mana yang cocok untuk berdasarkan kasus yang sedang di implementasikan.

1. Mencari buku sesuai nama untuk dipinjam (sequential searching)
2. Mengurutkan tahun release buku (sorting)
3. Menampilkan buku sesuai genre (display all setelah searching)

BAB II METODE PENCARIAN
DENGAN ITERASI DAN REKURSI
LIBRARY BERBASIS WEB

2.1 Iterasi

Untuk melakukan metode pencarian ataupun peng-generate'an data terdapat beberapa cara salah satunya adalah ada cara dengan menggunakan metode iterasi dimana ini digunakan dengan cara melakukan looping program hingga kondisi tertentu tercapai. Beberapa loop digunakan seperti for pada kasus ini digunakan untuk menginputkan data yang diulang hingga 5 kali sebanyak 1000 data sehingga mencapai 5000 data atau nilai ke n, Yang berarti O(n^2) karena terdapat syntax seperti query += "(?, ?, ?())" ini melakukan penyimpanan dan penyalinan data seperti iterasi pertama menyalin 1 bagian, iterasi 2 menyalin 2 data hingga seterusnya seperti 1 + 2 + 3 + 4 + 5 + 6 + 7 + … = n(n+1)/2 yang secara notasi ini adalah Big O yang jika disederhanakan adalah O(n^2).

2.1.1 Implementasi Code

```go
// MEMBUAT 5000 BUKU ITERATIF
func generate5kBooks(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Access-Control-Allow-Origin", "*")
    w.Header().Set("Content-Type", "application/json")
    start := time.Now()
    tx, err := db.Begin()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    titles := []string{"Misteri", "Teknologi", "Klasik", "Data", "Petualangan", "Dunia"}
    authors := []string{"J.K. Rowling", "George Orwell", "Tolkien", "Asimov", "Harari"}
    batchSize := 1000
    for b := 0; b < 5; b++ {
        query := "INSERT INTO books (title, author, image_url, created_at) VALUES "
        var values []any
        for i := 0; i < batchSize; i++ {
            query += "(?, ?, ?, NOW()),"
            randomTitle := titles[rng.Intn(len(titles))]
            randomAuthor := authors[rng.Intn(len(authors))]
            randomImage := coverImagePool[rng.Intn(len(coverImagePool))]
            bookTitle := fmt.Sprintf("%s %d", randomTitle, (b*batchSize)+i)
            values = append(values, bookTitle, randomAuthor, randomImage)
        }
        query = query[0 : len(query)-1]
        _, err = tx.Exec(query, values...)
        if err != nil {
            tx.Rollback()
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
    }
    tx.Commit()
    duration := time.Since(start).Seconds()
    json.NewEncoder(w).Encode(map[string]any{
        "status":   "success",
        "duration": duration,
    })
}
```

2.1.2 Fungsi Dan Alur Program

- **Fungsi generate5kBooks**

  Fungsi generate5kBooks merupakan HTTP handler yang digunakan untuk membuat dan menyimpan 5000 data buku secara iteratif dengan teknik batch insert guna meningkatkan efisiensi eksekusi database. Jadi beberapa generate query dari database di inputkan ke dalam iteratif untuk dihitung dan memilih judul acak yang dimiliki perpustakaan sesuai tabel judul yang dimiliki dan diteruskan output untuk bisa interaktif kepada user.

- **Alur Program**

  Program mengatur, Access-Control-Allow-Origin menjadi * agar endpoint dapat diakses dari berbagai origin. Content-Type menjadi application/json. Waktu awal dicatat menggunakan time.Now() untuk menghitung durasi eksekusi. Program memulai transaksi database menggunakan db.Begin(). Jika terjadi kesalahan, sistem langsung mengembalikan error HTTP 500. Beberapa array disiapkan untuk menghasilkan data buku secara acak, meliputi.

  Judul buku, Nama penulis. Selain itu, batchSize ditentukan sebesar 1000, sehingga total 5000 data dibagi menjadi 5 batch. Perulangan luar (for b := 0; b < 5; b++) mengatur jumlah batch. Perulangan dalam (for i := 0; i < batchSize; i++) membangun data buku dalam satu batch. Query SQL INSERT INTO books dikonstruksi secara dinamis dengan placeholder (?, ?, ?, NOW()). Data judul, penulis, dan gambar dimasukkan ke dalam slice values. Setelah query selesai disusun, query dieksekusi menggunakan tx.Exec(query, values). Jika terjadi kesalahan, transaksi dibatalkan (Rollback) dan error dikirim ke user. Setelah seluruh batch berhasil diproses, transaksi database dikonfirmasi menggunakan tx.Commit(). Lama waktu proses dihitung dalam satuan detik menggunakan time.Since(start).Seconds(). Program mengembalikan respons JSON yang berisi status proses (success), Lama waktu eksekusi.

- **Tujuan Fungsi**

  Fungsi ini bertujuan untuk menghasilkan data buku dalam jumlah besar secara efisien menggunakan metode iteratif dan batch processing, serta sebagai pembanding performa terhadap metode rekursif. Selain itu meggunakan processing iteratif karena langsung terhubung terhadap server yang bisa dibilang kecil oleh karena itu fungsi peng-generate'an dilakukan looping atau secara bertahap untuk menghidari server limit. Ini sangat ampuh untuk jenis website dengan skala yang kecil sehingga sangat interaktif secara realtime.

2.2 Rekursi

Pada implementasi teknik rekursi ini adalah gimana caranya melakukan perulangan dengan memanggil function dirinya sendiri sehingga terjadinya looping. Ini juga salah satu proses alternatif selain menggunakan iteratif. Kenapa harus tetap ada rekursi kalau sudah ada iteratif, kemungkinan besarnya adalah untuk memanajemen fungsi program biasanya agar jauh lebih ringkas dan rapi meskipun jika dibandingkan kecepatan dengan function iteratif ia kalah tetapi juga tetap memiliki kelebihannya juga dalam manajemen dan mencari file di dalam sub folder karena ia mencatat letak terakhir dimana ia berada dengan call stack dari para iterasi. Call stack sangat berguna untuk mencatat titik terakhir tetapi jika data sudah sangat besar ia akan membuat overhead call stack. Pada function ini juga tetap menggunakan kompleksitas O(n) karena ia mendefinisikan data secara berulang seperti pada syntax ini titles := []string{...}, sehingga prosesnya menumpuk pada RAM membuat waktu lebih lambat.

2.2.1 Implementasi Code

```go
// HANDLER GENERATE BUKU DIGITAL SECARA REKURSIF
func generateDigitalRecursive(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    start := time.Now()
    tx, _ := db.Begin()
    insertRecursive(1, 5000, tx)
    tx.Commit()
    duration := time.Since(start).Seconds()
    json.NewEncoder(w).Encode(map[string]any{
        "status":   "success",
        "duration": duration,
        "method":   "recursive",
    })
}
```

2.2.2 Fungsi Dan Alur Program

- **Fungsi generateDigitalRecursive**

  Fungsi generateDigitalRecursive berperan sebagai HTTP handler yang digunakan untuk meng-generate data buku digital ke dalam Database menggunakan teknik rekursif. Sesuai dengan kebutuhan ini hanyalah sebatas alat untuk uji banding antara 2 teknik perulangan untuk melihat hasil yang lebih baik dan yang sesuai untuk menghasilkan jumlah buku dengan jumlah yang sangat besar.

- **Alur Program**

  Program mengatur header Content-Type menjadi application/json, yang menandakan respons yang dikirim ke user berbentuk data JSON. Waktu awal eksekusi disimpan menggunakan time.Now() untuk menghitung durasi proses nantinya. Fungsi memulai transaksi database menggunakan db.Begin(). Transaksi ini digunakan agar seluruh proses insert data dapat dikelola secara atomik. Fungsi insertRecursive(1, 5000, tx) dipanggil untuk melakukan proses penyimpanan data buku ke database secara rekursif, dimulai dari data pertama hingga data ke-5000 dengan menggunakan transaksi yang sama.

  Setelah proses rekursif selesai, transaksi database dikonfirmasi menggunakan tx.Commit() agar seluruh perubahan tersimpan secara permanen. Durasi eksekusi dihitung dengan selisih waktu antara waktu awal dan waktu selesai proses menggunakan time.Since(start).Seconds(). Program mengirimkan respons dalam format JSON yang berisi: Status proses (success), Lama waktu eksekusi, Metode yang digunakan rekursif.

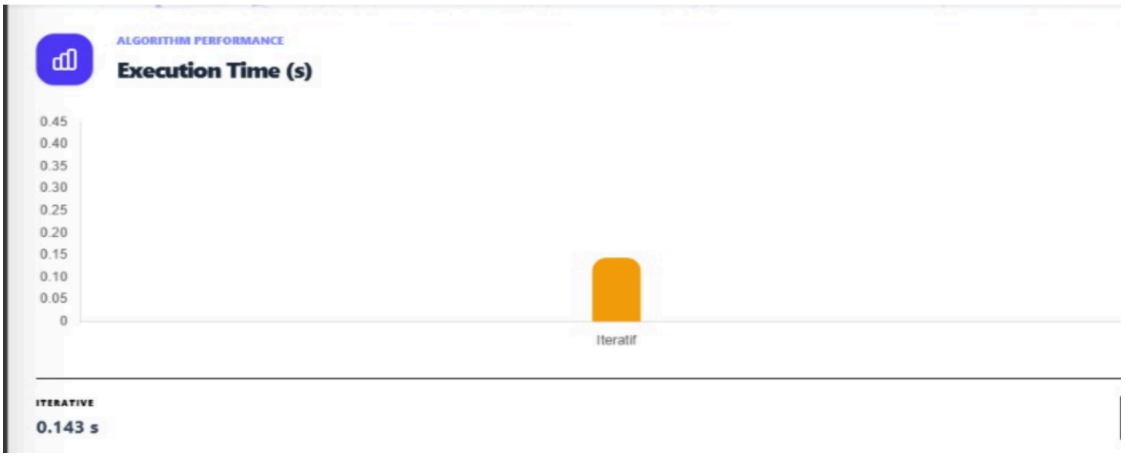
- **Tujuan Fungsi**

  Fungsi ini bertujuan untuk menguji dan menerapkan metode rekursif dalam pembuatan data buku digital serta membandingkan performanya dengan metode lain, seperti iteratif. Selain itu fungsi ini bertujuan untuk menghantarkan data dari user frontend untuk disimpan dalam database.

BAB III PERBANDINGAN
ITERASI DAN REKURSI

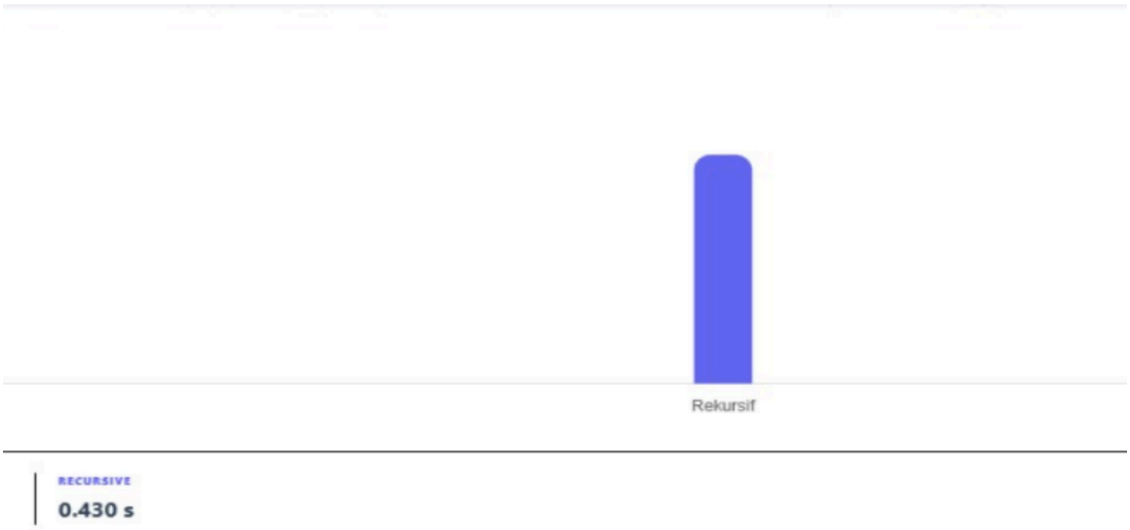
3.1 Kompleksitas Waktu

3.1.1 Iterasi (memuat perhitungan dan pencarian t(n) dan kompleksitas waktu)



Disini kami telah mengimplementasikan Algoritma Iteratif dalam melakukan fitur Generate buku sebanyak 5000 data, yang hasilnya menunjukan angka 0.143 detik.

3.1.2 Rekursi (memuat perhitungan dan pencarian t(n) dan kompleksitas waktu)



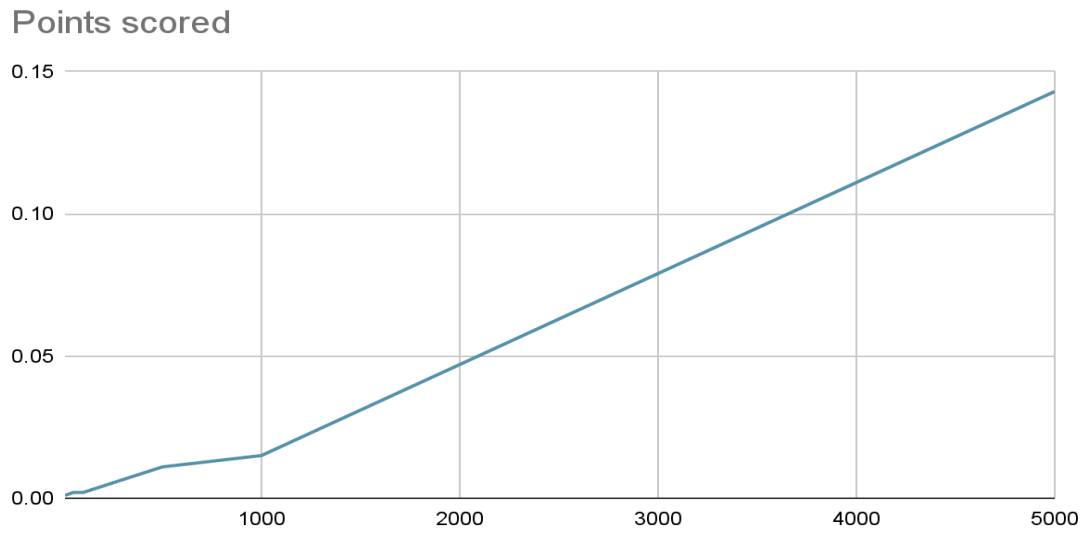
Disini kami juga telah mengimplementasikan Algoritma Rekursi untuk melakukan fitur yang sama yaitu Generate buku sebanyak 5000 data, hasilnya menunjukan angka 0.43 detik.

3.1.3 Perbandingan Algoritma Iteratif dan Rekursi

Kita dapat melihat perbandingan kedua Algoritma disini dalam melakukan Generate buku, sebanyak 5000 data. Terlihat jelas bahwa execution time T(n) yang diperoleh dari Algoritma Rekursi jauh lebih tinggi atau lebih lama daripada execution time T(n) yang diperoleh dari Algoritma Iteratif. Perbandingan Algoritma keduanya memiliki selisih running time sekitar 0.287 detik, dimana terlihat jelas bahwa Algoritma Iteratif lebih efisien daripada Algoritma Rekursi, walaupun dalam pengimplentasian Algoritma Rekursi jauh lebih sederhana daripada Algoritma Iteratif.

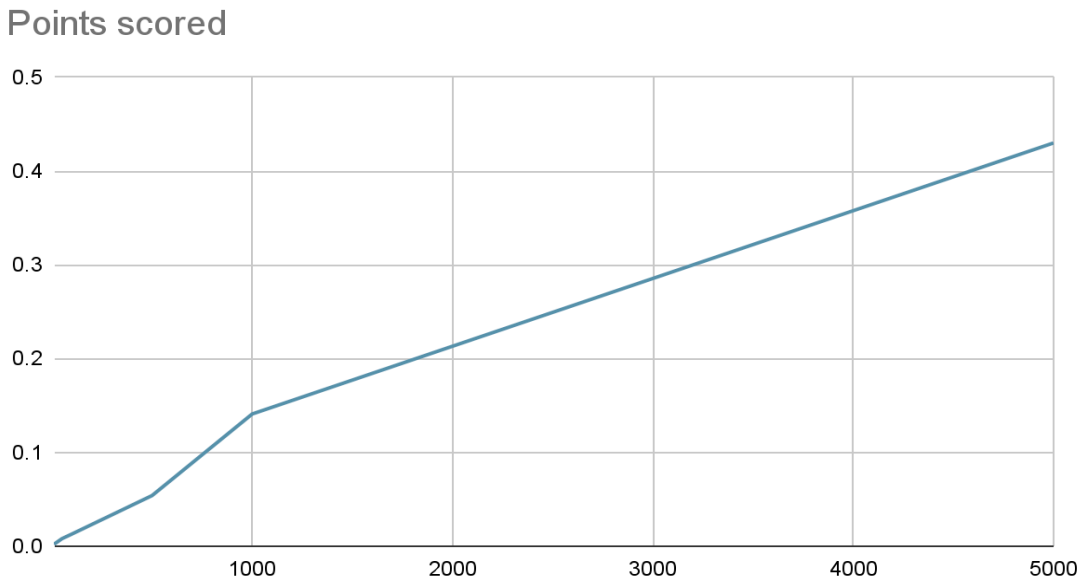
3.2 Grafik

3.2.1 Iterasi



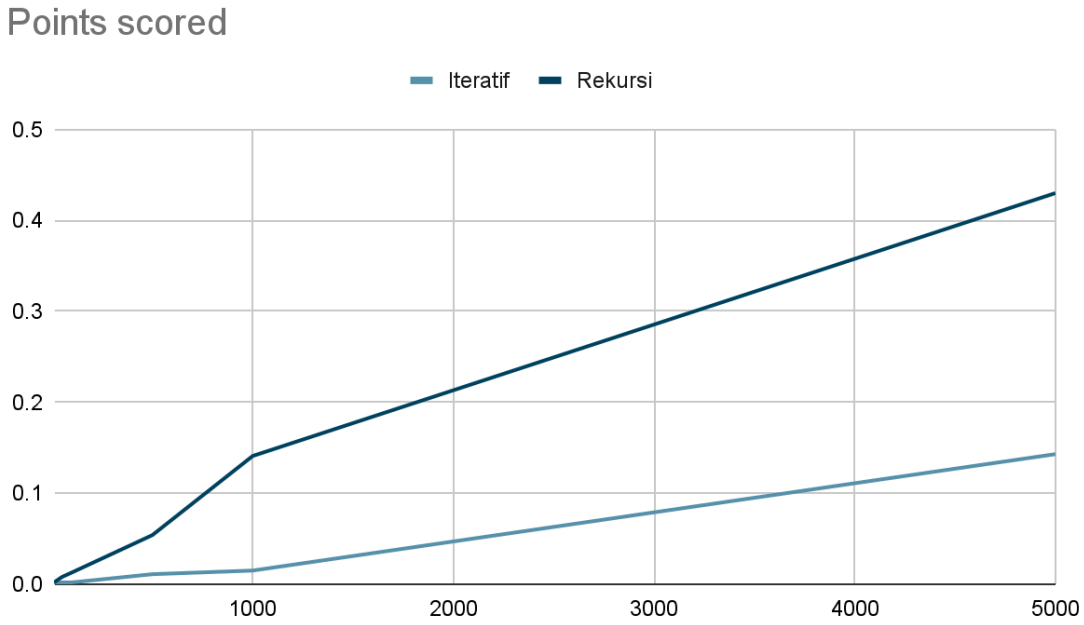
Grafik ini menunjukan perbandingan data dan running time untuk algoritma Iterasi, dimana pada saat 10 data, running time yang diperlukan adalah 0.001 detik, kemudian saat 50 dan 100 data running time yang diperlukan kurang lebih sama yaitu sekitar 0.002 detik, kemudian pada saat 500 data running time yang diperlukan adalah 0.011 detik, kemudian pada saat 1000 data running time yang diperlukan adalah 0.015 detik, dan terakhir ketika 5000 data maka running time yang diperlukan adalah 0.143 detik.

3.2.2 Rekursi



Grafik ini menunjukan perbandingan data dan running time untuk algoritma Rekursi, dimana pada saat 10 data, running time yang diperlukan adalah 0.002 detik, kemudian pada saat 50 data running time yang diperlukan adalah 0.008 detik, kemudian pada saat 100 data maka running time yang diperlukan adalah 0.013 detik, kemudian pada saat 500 data running time yang diperlukan adalah 0.054 detik, kemudian pada saat 1000 data running time yang diperlukan adalah 0.141 detik, dan terakhir ketika 5000 data maka running time yang diperlukan adalah 0.43 detik.

3.2.3 Perbandingan Grafik Iteratif dan Rekursi



Pada Grafik perbandingan antara Algoritma Rekursi dan Iteratif, terlihat jelas bahwa Algoritma Rekursi memiliki waktu running time yang lebih signifikan daripada Algoritma Iteratif, yang meunjukkan pada berapapun data yang kita gunakan, Algoritma Iteratif selalu lebih efisien daripada Algoritma Rekursi.

BAB IV PENUTUP

4.1 Kesimpulan

Berdasarkan analisa diatas diperoleh bahwa algoritma menggunakan Iteratif menunjukan perbedaan efisiensi yang cukup signifikan sebesar 0.287 detik sehingga dapat disimpulkan bahwa untuk kasus generate buku algoritma iteratif adalah pilihan yang terbaik untuk berapapun jumlah data yang akan di generate.

4.2 Saran

Sebaiknya untuk mengimplementasikan fitur seperti generate buku kita dapat menggunakan algoritma iteratif karena perbedaan running time sebesar 0.287 detik pada 5000 data akan lebih berpengaruh pada data yang lebih besar, sehingga kita perlu memilih algoritma yang paling efisien, paling cocok sesuai kebutuhan dan tercepat agar perangkat lunak dapat berjalan dengan paling baik.