LAPORAN TUGAS BESAR ANALISIS KOMPLEKSITAS ALGORITMA

Studi Efisiensi Algoritma *Iteratif* dan *Rekursif* untuk Menghitung Akumulasi Jam Belajar Mingguan dalam Periode Semester

Dosen Pengampu Mata Kuliah:
Selly Meliana M.Kom

Disusun Oleh:

KELOMPOK Okidoki

Rini Anisa	103052300017
Petrosina Angwarmase	103052300119

PROGRAM STUDI S1 SAINS DATA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM
BANDUNG
2024

DESKRIPSI STUDI KASUS PERMASALAHAN

Judul topik ini, *"Studi Efisiensi Algoritma Iteratif dan Rekursif untuk Menghitung Akumulasi Jam Belajar Mingguan dalam Periode Semester,"* berfokus pada analisis perbandingan dua pendekatan algoritma dalam menghitung total jam belajar yang dihabiskan oleh mahasiswa selama satu semester. Setiap minggu, mahasiswa mencatat jumlah jam yang mereka habiskan untuk belajar, dan data ini akan dijumlahkan secara keseluruhan dengan menggunakan metode iteratif dan rekursif.

Pada pendekatan iteratif, kita akan menggunakan loop untuk menjumlahkan jam belajar dari setiap minggu hingga akhir semester. Sedangkan dalam pendekatan rekursif, fungsi akan memanggil dirinya sendiri untuk menambahkan jam belajar setiap minggu, mulai dari minggu pertama hingga minggu terakhir. Tujuan dari studi ini adalah untuk membandingkan kedua metode dari sisi efisiensi, yaitu waktu eksekusi dan penggunaan memori, terutama ketika jumlah minggu semakin besar. Dengan demikian, hasil studi ini akan menunjukkan pendekatan mana yang lebih optimal untuk menghitung total waktu belajar mahasiswa pada skala yang lebih besar, seperti dalam periode semester penuh.

DEKSKRIPSI DUA ALGORITMA YANG DIPILIH UNTUK MENYELESAIKAN PERMASALAHAN

1. **Algoritma Iteratif**

```go
// Fungsi untuk menghitung total jam belajar dengan metode iteratif
func TotalJamBelajarIteratif(jamBelajar []int, N int) int {
    total := 0
    for i := 0; i < N; i++ {
        total += jamBelajar[i]
    }
    return total
}
```

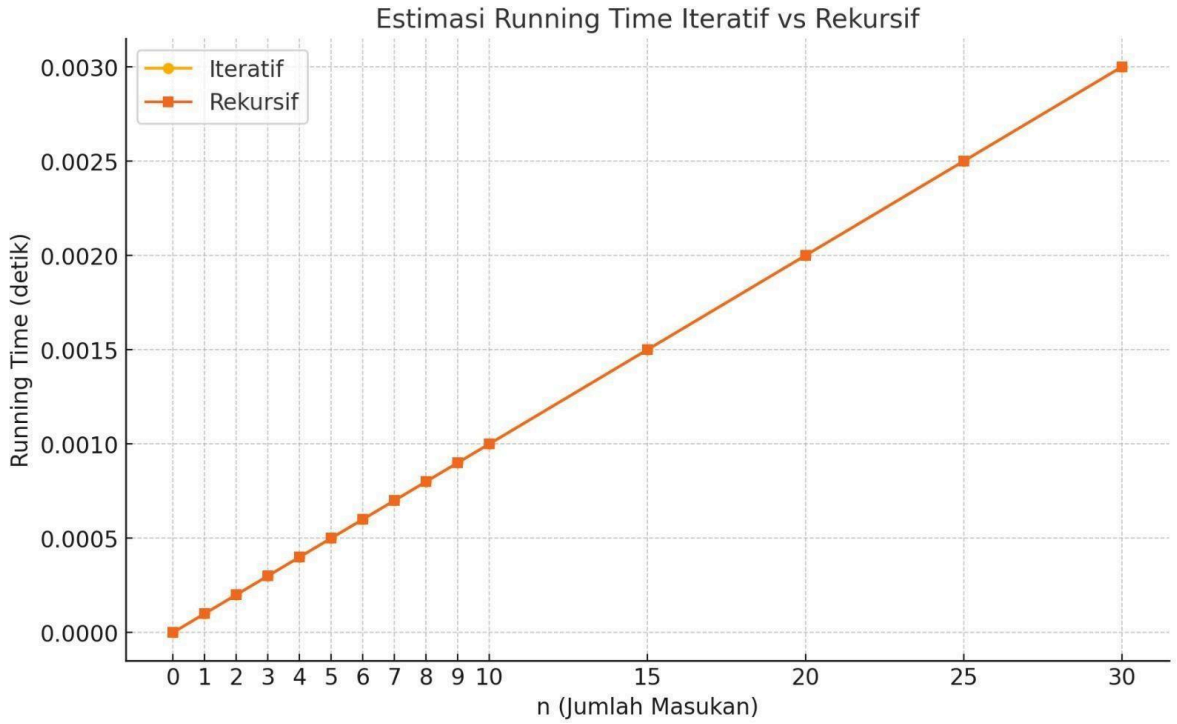
Algoritma ini menggunakan struktur perulangan untuk menjumlahkan jam belajar dari setiap minggu dalam sebuah daftar yang mewakili data mingguan. Proses iteratif melibatkan inisialisasi variabel total pada nilai nol, kemudian menjumlahkan setiap elemen secara berurutan.

2. **Algoritma Rekursif**

```go
// Fungsi rekursif untuk menghitung total jam belajar
func akumulasi(jamBelajar []int, N int) int {
    if N == 0 {
        return 0 // Basis rekursif: jika tidak ada elemen
    }
    return jamBelajar[N-1] + akumulasi(jamBelajar, N-1) // Rekursif
}
```

Algoritma ini menggunakan pendekatan rekursif untuk menghitung total jam belajar. Prosesnya adalah membagi masalah menjadi submasalah yang lebih kecil hingga mencapai kondisi dasar, yaitu ketika data hanya memiliki satu elemen.

GRAFIK PERBANDINGAN RUNNING TIME



ANALISIS PERBANDINGAN KEDUA ALGORITMA

1. **Kelas**

Kompleksitas

```go
// Fungsi untuk menghitung total jam belajar dengan metode iteratif
func TotalJamBelajarIteratif(jamBelajar []int, N int) int {
    total := 0
    for i := 0; i < N; i++ {
        total += jamBelajar[i]
    }
    return total
}
```

Iteratif:

$$\sum_{i=0}^{N-1} 1 = N - 1 - 0 + 1$$

$$= N \in O(N)$$

Rekursif:

```go
// Fungsi rekursif untuk menghitung total jam belajar
func akumulasi(jamBelajar []int, N int) int {
    if N == 0 {
        return 0 // Basis rekursif: jika tidak ada elemen
    }
    return jamBelajar[N-1] + akumulasi(jamBelajar, N-1) // Rekursif
}
```

T(N) = {0 jika N = 0

{1 + T(N-1) jika N > 0

↓

T(1) = 1

T(0) = 0

T(N) = 1 + T(N-1)

T(N-1) = 1 + (1 + T(N-2))

T(N-2) = 1(1 + (1 + T(N-3)))

T(N) = i + T(N-i)

i = n - i

= N-1 + T(N-(N-1))

= N-1 + T(N-N+1)

= N-1 + T(1)

= N-1 + 1

= N ∈ O(N)

2. **Running Time**

Masukan	T(n)		Estimasi Running Time	
n	Iteratif -> n	Rekursif ->n	Iteratif	Rekursif
0	0	0	0	0
1	1	1	0,0001	0,0001
2	2	2	0,0002	0,0002
3	3	3	0,0003	0,0003
4	4	4	0,0004	0,0004
5	5	5	0,0005	0,0005
5	5	5	0,0005	0,0005
6	6	6	0,0006	0,0006
7	7	7	0,0007	0,0007
8	8	8	0,0008	0,0008
9	9	9	0,0009	0,0009
10	10	10	0,001	0,001
15	15	15	0,0015	0,0015
20	20	20	0,002	0,002
25	25	25	0,0025	0,0025
30	30	30	0,003	0,003

Asumsikan Cop = 10^-4

$$T(n) \approx c_{op} C(n)$$

REFERENSI

Power point materi, buku Levitin, dan beberapa sumber lain di website.

LINK GITHUB

https://github.com/revrinn/Tubes-AKA