

# **TUGAS MODUL PRAKTIKUM 3**



**Disusun oleh :**

**Anne Audistya Fernanda**

**140810180059**

**Kelas A**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

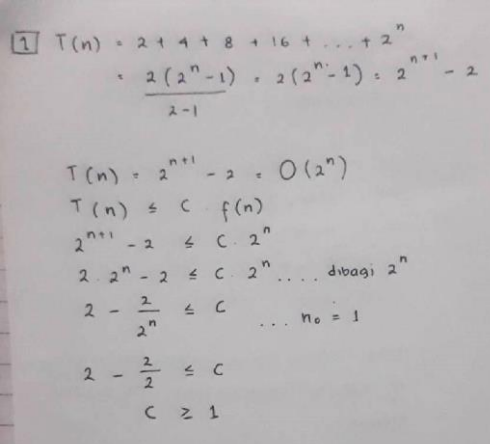
**2020**

## Latihan Analisa

Minggu ini kegiatan praktikum difokuskan pada latihan menganalisa, sebagian besar tidak perlu menggunakan komputer dan mengkode program, gunakan pensil dan kertas untuk menjawab persoalan berikut!

1. Untuk  $T(n) = 2 + 4 + 8 + 16 + \dots + n^2$ , tentukan nilai  $C$ ,  $f(n)$ ,  $n_0$ , dan notasi Big-O sedemikian sehingga  $T(n) = O(f(n))$  jika  $T(n) \leq C$  untuk semua  $n \geq n_0$

Jawab :

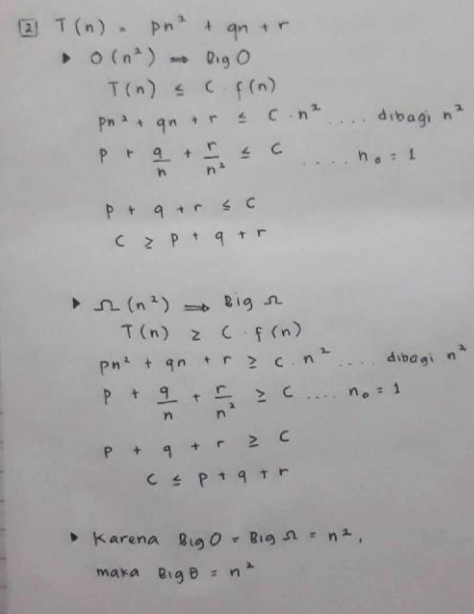


1  $T(n) = 2 + 4 + 8 + 16 + \dots + 2^n$   
 $= \frac{2(2^n - 1)}{2 - 1} = 2(2^n - 1) = 2^{n+1} - 2$   
 $T(n) = 2^{n+1} - 2 = O(2^n)$   
 $T(n) \leq C \cdot f(n)$   
 $2^{n+1} - 2 \leq C \cdot 2^n$   
 $2 \cdot 2^n - 2 \leq C \cdot 2^n \dots \text{dibagi } 2^n$   
 $2 - \frac{2}{2^n} \leq C \dots n_0 = 1$   
 $2 - \frac{2}{2} \leq C$   
 $C \geq 1$

2. Buktikan bahwa untuk konstanta-konstanta positif  $p$ ,  $q$ , dan  $r$ :

$$T(n) = pn^2 + qn + r \text{ adalah } O(n^2), \Omega(n^2), \Theta(n^2)$$

Jawab :



2  $T(n) = pn^2 + qn + r$   
►  $O(n^2) \Rightarrow \text{Big } O$   
 $T(n) \leq C \cdot f(n)$   
 $pn^2 + qn + r \leq C \cdot n^2 \dots \text{dibagi } n^2$   
 $p + \frac{q}{n} + \frac{r}{n^2} \leq C \dots n_0 = 1$   
 $p + q + r \leq C$   
 $C \geq p + q + r$   
►  $\Omega(n^2) \Rightarrow \text{Big } \Omega$   
 $T(n) \geq C \cdot f(n)$   
 $pn^2 + qn + r \geq C \cdot n^2 \dots \text{dibagi } n^2$   
 $p + \frac{q}{n} + \frac{r}{n^2} \geq C \dots n_0 = 1$   
 $p + q + r \geq C$   
 $C \leq p + q + r$   
► Karena  $\text{Big } O = \text{Big } \Omega = n^2$ ,  
maka  $\text{Big } \Theta = n^2$

3. Tentukan waktu kompleksitas asimptotik (Big-O, Big-Ω, dan Big-Θ) dari kode program berikut:

```

for k ← 1 to n do
  for i ← 1 to n do
    for j ← 1 to n do
       $w_{ij} \leftarrow w_{ij} \text{ or } w_{ik} \text{ and } w_{kj}$ 
    endfor
  endfor
endfor

```

Jawab :

3)   
 for k ← 1 to n do  
   for i ← 1 to n do  
     for j ← 1 to n do  
        $w_{ij} \leftarrow w_{ij} \text{ or } w_{ik} \text{ and } w_{kj} \Rightarrow n \cdot n \cdot n$   
     endfor  
   endfor  
 endfor  
 $T(n) = n^3$

▶ Big O  
 $n^3 \leq C \cdot n^3 \dots$  dibagi  $n^3$   
 $1 \leq C$   
 $C \geq 1$

▶ Big Ω  
 $n^3 \geq C \cdot n^3 \dots$  dibagi  $n^3$   
 $1 \geq C$   
 $C \leq 1$

▶ Big Θ  
 Karena Big O = Big Ω =  $n^3$ ,  
 maka Big Θ =  $\Theta(n^3)$

4. Tulislah algoritma untuk menjumlahkan dua buah matriks yang masing-masing berukuran  $n \times n$ . Berapa kompleksitas waktunya  $T(n)$ ? dan berapa kompleksitas waktu asimptotiknya yang dinyatakan dalam Big-O, Big-Ω, dan Big-Θ?

Jawab :

4) Algoritma penjumlahan matriks  $n \times n$   
 for i ← 1 to n do  
   for j ← 1 to n do  
      $m_{ij} \leftarrow a_{ij} + b_{ij} \Rightarrow n \cdot n$   
   endfor  
 end for  
 $T(n) = n^2$

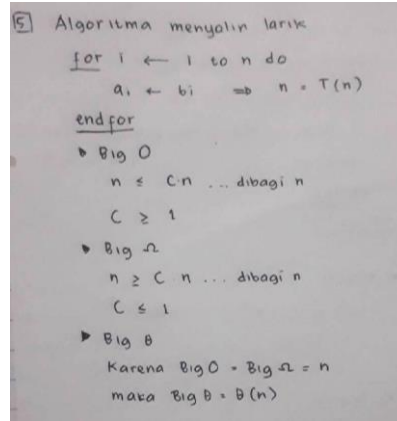
▶ Big O  
 $n^2 \leq C \cdot n^2 \dots$  dibagi  $n^2$   
 $C \geq 1$

▶ Big Ω  
 $n^2 \geq C \cdot n^2 \dots$  dibagi  $n^2$   
 $C \leq 1$

▶ Big Θ  
 Karena Big O = Big Ω =  $n^2$ ,  
 maka big Θ =  $\Theta(n^2)$

5. Tulislah algoritma untuk menyalin (copy) isi sebuah larik ke larik lain. Ukuran elemen larik adalah  $n$  elemen. Berapa kompleksitas waktunya  $T(n)$ ? dan berapa kompleksitas waktu asimptotiknya yang dinyatakan dalam Big-O, Big- $\Omega$ , dan Big- $\Theta$ ?

Jawab :



6. Diberikan algoritma Bubble Sort sebagai berikut:

```

procedure BubbleSort(input/output a1, a2, ..., an; integer)
{ Mengurut tabel integer TabInt[1..n] dengan metode pengurutan bubble-
  sort
  Masukan: a1, a2, ..., an
  Keluaran: a1, a2, ..., an (terurut menaik)
}
Deklarasi
  k : integer { indeks untuk traversal tabel }
  pass : integer { tahapan pengurutan }
  temp : integer { peubah bantu untuk pertukaran elemen tabel }
Algoritma
  for pass ← 1 to n - 1 do
    for k ← n downto pass + 1 do
      if ak < ak-1 then
        { pertukarkan ak dengan ak-1 }
        temp ← ak
        ak ← ak-1
        ak-1 ← temp
      endif
    endfor
  endfor

```

- Hitung berapa jumlah operasi perbandingan elemen-elemen tabel!
- Berapa kali maksimum pertukaran elemen-elemen tabel dilakukan?
- Hitung kompleksitas waktu asimptotik (Big-O, Big- $\Omega$ , dan Big- $\Theta$ ) dari algoritma Bubble Sort tersebut!

Jawab :

6 a) jumlah operasi perbandingan  
 $1 + 2 + 3 + 4 + \dots + (n-1)$   
 $= \frac{n(n-1)}{2}$  kali

b) berapa kali maksimum pertukaran elemen-elemen tabel dilakukan?  
 $= \frac{n(n-1)}{2}$  kali

c) Hitung Kompleksitas

► Best Case (semua terurut)  
 $\frac{(n-1)n}{2}$  kali,  $T_{\min}(n) = \frac{n(n-1)}{2}$   
 $= \underline{n^2 - n}$

► Worst Case (semua terbalik)  
 Perbandingan  $\rightarrow \frac{n(n-1)}{2}$   
 Memasukkan nilai  $\rightarrow \frac{3n(n-1)}{2}$   
 $T_{\max}(n) = \frac{4n(n-1)}{2} = 2n^2 - 2n$

Big O	Big Ω
$2n^2 - 2n \leq C \cdot n^2 \dots$ dibagi $n^2$	$\frac{n^2 - n}{2} \geq C \cdot n^2 \dots$ dibagi $n^2$
$2 - \frac{2}{n} \leq C \dots n = 1$	$\frac{1}{2} - \frac{1}{2n} \geq C \dots n = 1$
$2 - 2 \leq C$	$\frac{1}{2} - \frac{1}{2} \geq C$
$C \geq 0$	$C \leq 0$

7. Untuk menyelesaikan problem X dengan ukuran N tersedia 3 macam algoritma:
- Algoritma A mempunyai kompleksitas waktu  $O(\log N)$
  - Algoritma B mempunyai kompleksitas waktu  $O(N \log N)$
  - Algoritma C mempunyai kompleksitas waktu  $O(N^2)$

Untuk problem X dengan ukuran  $N=8$ , algoritma manakah yang paling cepat? Secara asimptotik, algoritma manakah yang paling cepat?

Jawab :

7 a) Algoritma A  $\rightarrow O(\log N)$   
 b) Algoritma B  $\rightarrow O(N \log N)$   
 c) Algoritma C  $\rightarrow O(N^2)$

Jika  $N = 8$ , mana Algoritma yang paling efektif?

a)  $O(\log 8) = O(3 \log 2)$   
 b)  $O(8 \log 8) = O(24 \log 2)$   
 c)  $O(8^2) = O(64)$

yang paling efektif adalah algoritma A, karena semakin kecil  $O()$  semakin efektif.

8. Algoritma mengevaluasi polinom yang lebih baik dapat dibuat dengan metode Horner berikut:

$$p(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \dots + x(a_{n-1} + a_n x)))) \dots)$$

function p2(input x : real) → real  
( Mengembalikan nilai p(x) dengan metode Horner)

**Deklarasi**

k : integer

b<sub>1</sub>, b<sub>2</sub>, ..., b<sub>n</sub> : real

**Algoritma**

b<sub>n</sub> ← a<sub>n</sub>

for k ← n - 1 downto 0 do

    b<sub>k</sub> ← a<sub>k</sub> + b<sub>k+1</sub> \* x

endfor

return b<sub>0</sub>

Hitunglah berapa operasi perkalian dan penjumlahan yang dilakukan oleh algoritma diatas, Jumlahkan kedua hitungan tersebut, lalu tentukan kompleksitas waktu asimptotik (Big-O)nya. Manakah yang terbaik, algoritma p atau p2?

Jawab :

18) Operasi memasukkan nilai

- b<sub>n</sub> ← a<sub>n</sub>                      1 kali
- b<sub>k</sub> ← a<sub>k</sub> + b<sub>k+1</sub> \* x            n kali

T(n) = n + 1

O(n) = untuk p<sup>2</sup>

Algoritma P

Pengjumlahan    n kali

Perkalian            n kali

T(n) = 2n

P<sup>2</sup> lebih baik dari P