**LAPORAN TUGAS 2**

**PRAKTIKUM ANALISIS ALGORITMA**

****

Satrio Sadrakh Allesandro

140810160038

**PRODI S-1 TEKNIK INFORMATIKA**

**DEPARTEMEN ILMU KOMPUTER**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

**2018**

**Studi Kasus 1**

**Algoritma:**

maks 🡨 x1

i 🡨 2

while i ≤ n do

if xi > maks then

maks 🡨 xi

endif

i 🡨 i + 1

endwhile

**Kompleksitas Waktu**

1. Operasi pengisian nilai (assignment)

maks 🡨 x1 1 kali

i 🡨 2 1 kali

maks 🡨 xi (n-1) kali

i 🡨 i + 1 (n-1) kali

Jumlah seluruh operasi pengisian nilai (assignment) adalah

T1 = 1 + 1 + n - 1+ n - 1 = 2n

1. Operasi penjumlahan

i + 1 n kali

Jumlah seluruh operasi penjumlahan adalah

T2 = n

1. Operasi perbandingan

xi > maks n-1 kali

Jumlah seluruh operasi perbandingan adalah

T3 = n-1

Dengan demikian, kompleksitas waktu algoritma dihitung berdasarkan jumlah operasi aritmatika dan operasi pengisian nilai adalah:

T(n) = 2n + n + n - 1 = 4n -1

* Tmin(n=1) = 4\*1 - 1 = 3
* Tavg(n=(n+1)/2) = 4\*(n+1)/2 - 1 = 2n + 1
* Tmax(n) = 4n - 1

**Studi Kasus 2**

**Algoritma:**

i 🡨 1

found 🡨 false

while (i ≤ n) and (not found) do

if xi = y then

found 🡨 true

else

i 🡨 i + 1

endif

endwhile

{i < n or found}

If found then {y ditemukan}

idx 🡨 i

else

idx 🡨 0 {y tidak ditemukan}

endif

**Kompleksitas Waktu**

1. Operasi pengisian nilai (assignment)

i 🡨 1 1 kali

found 🡨 false 1 kali

found 🡨 true n kali

i 🡨 i + 1 n kali

idx 🡨 1 1 kali

idx 🡨 0 1 kali

Jumlah seluruh operasi pengisian nilai (assignment) adalah

T1 = 1 + 1 + n + n + 1 + 1 = 2n + 4

1. Operasi penjumlahan

i + 1 n kali

Jumlah seluruh operasi penjumlahan adalah

T2 = n

1. Operasi perbandingan

xi = y n kali

Jumlah seluruh operasi perbandingan adalah

T3 = n

Dengan demikian, kompleksitas waktu algoritma dihitung berdasarkan jumlah operasi aritmatika dan operasi pengisian nilai adalah:

T(n) = 2n + 4 + n + n = 4n + 4

* Tmin(n=1) = 4\*1 + 4 = 8
* Tavg(n=(n+1)/2) = 4\*(n+1)/2 + 4 = 2n + 6
* Tmax(n) = 4n + 4

**Studi Kasus 3**

**Algoritma:**

i 🡨 1

j 🡨 n

found 🡨 false

while (not found) and ( i ≤ j) do

mid 🡨 (i + j) div 2

if xmid = y then

found 🡨 true

else

if xmid < y then {mencari di bagian kanan}

i 🡨 mid + 1

else {mencari di bagian kiri}

j 🡨 mid – 1

endif

endif

endwhile

{found or i > j }

If found then

Idx 🡨 mid

else

Idx 🡨 0

endif

**Kompleksitas Waktu**

1. Operasi pengisian nilai (assignment)

i 🡨 1 1 kali

j 🡨 n 1 kali

found 🡨 false 1 kali

mid 🡨 (i + j) div 2 2log n kali

found 🡨 true 2log n kali

i 🡨 mid + 1 2log n kali

i 🡨 mid – 1 2log n kali

idx 🡨 I 1 kali

idx 🡨 0 1 kali

Jumlah seluruh operasi pengisian nilai (assignment) adalah

T1 = 1 + 1 + 1 + 2log n + 2log n + 2log n + 2log n + 1 + 1 = 4 2log n + 5

1. Operasi penjumlahan

i + j 2log n kali

mid + 1 2log n kali

mid – 1 2log n kali

Jumlah seluruh operasi penjumlahan adalah

T2 = 2log n + 2log n + 2log n = 3 2log n

1. Operasi pembagian

(i + j) div 2 2log n kali

Jumlah seluruh operasi penjumlahan adalah

T3 = 2log n

1. Operasi perbandingan

xmid = y 2log n kali

xmid < y 2log n kali

Jumlah seluruh operasi perbandingan adalah

T4 = 2 2log n

Dengan demikian, kompleksitas waktu algoritma dihitung berdasarkan jumlah operasi aritmatika dan operasi pengisian nilai adalah:

T(n) = 4 2log n + 5 + 3 2log n + 2log n + 2 2log n = 10 2log n + 5

* Tmin(n=1) = 10\*2log 1 + 5 = 5
* Tavg(n=(n+1)/2) = 10\*2log (n+1)/2 + 5
* Tmax(n) = 10 2log n + 5

**Studi Kasus 4**

**Algoritma:**

for i 🡨 2 to n do

insert 🡨 xi

j 🡨 i

while (j < i) and (x[j-i] > insert) do

x[j]🡨 x[j-1]

j🡨j-1

endwhile

x[j] = insert

endfor

**Kompleksitas Waktu**

1. Operasi pengisian nilai (assignment)

insert 🡨 xi n(n-1)/2 kali

j 🡨 i n(n-1)/2 kali

x[j]🡨 x[j-1] (n(n-1)/2)(n) kali

j🡨j-1 (n(n-1)/2)(n) kali

x[j] = insert n(n-1)/2 kali

Jumlah seluruh operasi pengisian nilai (assignment) adalah

T1 = (n – 1) \* 3 + (n-1)(n) + (n-1)(n) = 2n2 + n - 3

1. Operasi penjumlahan

j-1 (n-1)(n) kali

Jumlah seluruh operasi penjumlahan adalah

T2 = n2 - n

1. Operasi perbandingan

(j < i) (n-1)(n) kali

(x[j-i] > insert (n-1)(n) kali

Jumlah seluruh operasi perbandingan adalah

T3 = 2n2 – 2n

Dengan demikian, kompleksitas waktu algoritma dihitung berdasarkan jumlah operasi aritmatika dan operasi pengisian nilai adalah:

T(n) = 2n2 + n – 3 + n2 – n + 2n2 – 2n = 5n2 – 2n - 3

* Tmin(n=1) = 5\*1^2 – 2\*1 - 3 = 0
* Tavg(n=1/2n) = 5\*(1/2n)^2 – 2\*(1/2n) - 3 = 1.25 n2 – n - 3
* Tmax(n) = 5n2 – 2n - 3

**Studi Kasus 5**

**Algoritma:**

for i 🡨 n downto 2 do {pass sebanyak n-1 kali}

imaks 🡨 1

for j 🡨 2 to i do

if xj > ximaks then

imaks 🡨 j

endif

endfor

temp 🡨 xi

xi 🡨 ximaks

ximaks 🡨 temp

endfor

**Kompleksitas Waktu**

1. Operasi pengisian nilai (assignment)

imaks 🡨 1 n-1 kali

imaks 🡨 j (n-1)(n-1) kali

temp 🡨 xi n-1 kali

xi 🡨 ximaks n-1 kali

ximaks 🡨 temp n-1 kali

Jumlah seluruh operasi pengisian nilai (assignment) adalah

T1 = (n – 1) \* 4 + (n-1)(n-1) = n2 + 2n - 3

1. Operasi perbandingan

xj > ximaks (n-1)(n-1) kali

Jumlah seluruh operasi perbandingan adalah

T3 = n2 – 2n + 1

Dengan demikian, kompleksitas waktu algoritma dihitung berdasarkan jumlah operasi aritmatika dan operasi pengisian nilai adalah:

T(n) = n2 + 2n – 3 + n2 – 2n + 1= 2n2 - 2

* Tmin(n=1) = 2\*1^2 – 2 = 0
* Tavg(n=1/2n) = 2\*(1/2n)^2 – 2 = n2 – 2
* Tmax(n) = 2n2 - 2