

LAPORAN PRAKTIKUM 4

ANALISIS ALGORITMA



NAMA : Nadhifal Abdurrahman Rendusara
NPM : 140810180048
KELAS : B

PROGRAM STUDI S-1 TEKNIK INFORMATIKA
DEPARTEMEN ILMU KOMPUTER

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN

Worksheet 4

Studi Kasus 1: MERGE SORT

Setelah Anda mengetahui Algoritma Merge-Sort mengadopsi paradigma divide & conquer, lakukan Hal berikut:

1. Buat program Merge-Sort dengan bahasa C++
2. Kompleksitas waktu algoritma merge sort adalah $O(n \lg n)$. Cari tahu kecepatan komputer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

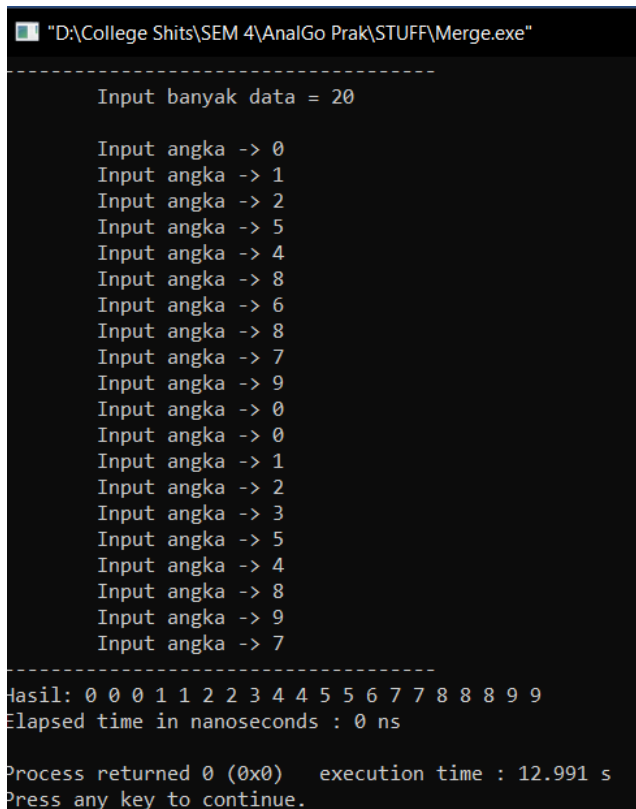
Jawaban Studi Kasus 1.1

```
1. /*
2. Nama    = Nadhifal A. Rendusara
3. NPM     = 140810180048
4. Kelas  = B
5. */
6. #include<iostream>
7. #include <chrono>
8. using namespace std;
9.
10. void satu(int* in, int p, int q,int r);
11. void merge(int* in, int p, int r);
12. void input(int* a, int& n);
13.
14. int main(){
15.     int in[100];
16.     int n;
17.     cout<<"\nPROGRAM MENGURUTKAN DENGAN MERGE SORT"<<endl;
18.     cout<<"-----"<<endl;
19.     input(in,n);
20.     auto start = chrono::steady_clock::now();
21.     merge(in,1,n);
22.     auto end = chrono::steady_clock::now();
23.     cout<<"-----"<<endl;
24.     cout << "Hasil: ";
25.     for(int i=0; i<n; i++){
26.         cout << in[i] << " ";
27.     }
28.
29.     cout<<endl;
30.     cout << "Elapsed time in nanoseconds : "
31.         << chrono::duration_cast<chrono::nanoseconds>(end - start).count()
32.         << " ns" << endl;
33.
34.     return 0;
35. }
36.
37. void satu(int* in, int p, int q,int r){
38.     int n1 = q-p+1;
39.     int n2 = r-q;
40.     int L[n1+1];
41.     int R[n2+1];
42.     for (int i=1; i<=n1; i++){
43.         L[i-1] = in[(p-1)+i-1];
44.     }
45.
46.     for (int j=1; j<=n2; j++){
47.         R[j-1] = in[(q-1)+j];
48.     }
49.
50.     int i=0;
```

```

51.     int j=0;
52.     L[n1]=2147483647;
53.     R[n2]=2147483647;
54.
55.     for (int k=(p-1); k<r; k++){
56.         if(L[i]<=R[j]){
57.             in[k]=L[i];
58.             i = i+1;
59.         }
60.         else{
61.             in[k]=R[j];
62.             j = j+1;
63.         }
64.     }
65. }
66.
67. void merge(int* in, int p, int r){
68.     int q;
69.     if(p<r){
70.         q = (p+r)/2;
71.         merge(in, p, q);
72.         merge(in, q+1, r);
73.
74.         satu(in, p, q, r);
75.     }
76. }
77.
78. void input(int* a, int& n){
79.     cout << "\tInput banyak data = "; cin >> n;
80.     cout<<endl;
81.     for (int i=0; i<n; i++){
82.         cout << "\tInput angka -> "; cin >> a[i];
83.     }
84. }

```



```

"D:\College Shits\SEM 4\AnalGo Prak\STUFF\Merge.exe"
-----
Input banyak data = 20

Input angka -> 0
Input angka -> 1
Input angka -> 2
Input angka -> 5
Input angka -> 4
Input angka -> 8
Input angka -> 6
Input angka -> 8
Input angka -> 7
Input angka -> 9
Input angka -> 0
Input angka -> 0
Input angka -> 1
Input angka -> 2
Input angka -> 3
Input angka -> 5
Input angka -> 4
Input angka -> 8
Input angka -> 9
Input angka -> 7
-----
Hasil: 0 0 0 1 1 2 2 3 4 4 5 5 6 7 7 8 8 8 9 9
Elapsed time in nanoseconds : 0 ns

Process returned 0 (0x0)   execution time : 12.991 s
Press any key to continue.

```

Jawaban Studi Kasus 1.2

Untuk di program di atas, hasilnya adalah 0 ns
Tapi jika sesuai dengan O -> T $(20 \log_{10} 20) = 26$

Studi Kasus 2: SELECTION SORT

Selection sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma selection sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma selection sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode recursion-tree** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma selection sort dengan menggunakan bahasa C++

Jawaban Studi Kasus 2

```
for i ← n downto 2 do {pass sebanyak n-1 kali}
  imaks ← 1
  for j ← 2 to i do
    if  $x_j > x_{imaks}$  then
      imaks ← j
    endif
  endfor
  {pertukarkan  $x_{imaks}$  dengan  $x_i$ }
  temp ←  $x_i$ 
   $x_i$  ←  $x_{imaks}$ 
   $x_{imaks}$  ← temp
endfor
```

Subproblem = 1
Masalah setiap subproblem = $n-1$
Waktu proses pembagian = n
Waktu proses penggabungan = n

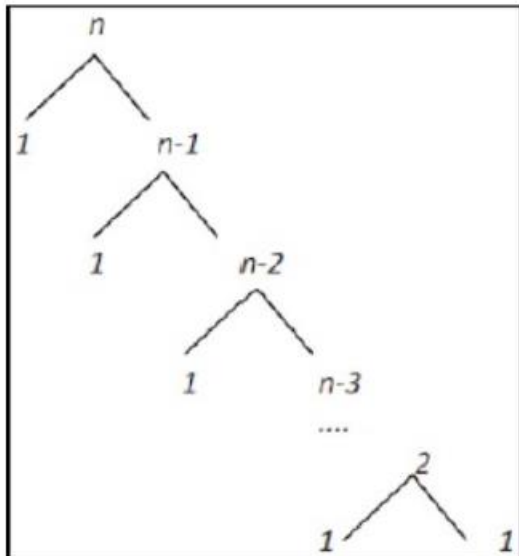
$$\begin{aligned} T(n) &= cn + cn-c + cn-2c + \dots + 2c + cn \\ &= c((n-1)(n-2)/2) + cn \\ &= c((n^2-3n+2)/2) + cn \\ &= c(n^2/2) - (3n/2) + 1 + cn \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn + cn-c + cn-2c + \dots + 2c + cn \\ &= c((n-1)(n-2)/2) + cn \\ &= c((n^2-3n+2)/2) + cn \\ &= c(n^2/2) - (3n/2) + 1 + cn \\ &= \Omega(n^2) \end{aligned}$$

$$T(n) = cn^2$$

$$= \Theta(n^2)$$

$$T(n) = \{\theta(1) T(n-1) + \theta(n)\}$$



```

1.  /*
2.  Nama      = Nadhifal A. Rendusara
3.  NPM       = 140810180048
4.  Kelas    = B
5.  */
6.  #include <iostream>
7.  #include<conio.h>
8.  using namespace std;
9.
10. int data[100],data2[100];
11. int n;
12.
13. void swtch(int a, int b);
14. void selectionSort();
15.
16. int main(){
17.     cout<<"\nPROGRAM MENGURUTKAN DENGAN SELECTION SORT"<<endl;
18.     cout<<"-----"<<endl;
19.     cout<<"\tMasukkan Jumlah Data : ";cin>>n;
20.     cout<<endl;
21.     for(int i=1;i<=n;i++){
22.         cout<<"\tMasukkan data ke-"<<i<<" : ";
23.         cin>>data[i];
24.         data2[i]=data[i];
25.     }
26.     selectionSort();
27.     cout << "-----" << endl;
28.     cout<<"\tData Setelah di Sort : "<<endl;
29.     cout<<"\t";
30.     for(int i=1; i<=n; i++){
31.         cout<<" "<<data[i];
32.     }
33.     cout << "\n=====\\n";
34.     getch();
35. }
36.
37. void swtch(int a, int b){

```

```

38.     int t;
39.     t = data[b];
40.     data[b] = data[a];
41.     data[a] = t;
42. }
43.
44. void selectionSort(){
45.     int pos,i,j;
46.     for(i=1;i<=n-1;i++) {
47.         pos = i;
48.         for(j = i+1;j<=n;j++) {
49.             if(data[j] < data[pos]) pos = j;
50.         }
51.         if(pos != i) swtch(pos,i);
52.     }
53. }

```

```

"D:\College Shits\SEM 4\AnalGo Prak\STUFF\Selection.exe"

PROGRAM MENGURUTKAN DENGAN SELECTION SORT
-----
Masukkan Jumlah Data : 5

Masukkan data ke-1 : 2
Masukkan data ke-2 : 6
Masukkan data ke-3 : 5
Masukkan data ke-4 : 8
Masukkan data ke-5 : 4
-----
Data Setelah di Sort :
2 4 5 6 8
=====

```

Studi Kasus 3: INSERTION SORT

Insertion sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma insertion sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma insertion sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode substitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++

Jawaban Studi Kasus 3

Algoritma

```
for i ← 2 to n do
    insert ← xi
    j ← i
    while (j < i) and (x[j-1] > insert) do
        x[j] ← x[j-1]
        j ← j-1
    endwhile
    x[j] = insert
endfor
```

Subproblem = 1
Masalah setiap subproblem = n-1
Waktu proses penggabungan = n
Waktu proses pembagian = n

$$T(n) = \{\theta(1) T(n-1) + \theta(n)\}$$

$$\begin{aligned} T(n) &= cn + cn - c + cn - 2c + \dots + 2c + cn \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + cn \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + cn \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + c + cn \leq 2cn^2 + cn^2 \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn \leq cn \\ &= \Omega(n) \end{aligned}$$

$$\begin{aligned} T(n) &= (cn + cn^2)/n \\ &= \Theta(n) \end{aligned}$$

```
1. /*
2. Nama      = Nadhifal A. Rendusara
3. NPM       = 140810180048
4. Kelas    = B
5. */
6. #include <iostream>
7. #include <conio.h>
```



```

8. using namespace std;
9.
10. int data[100],data2[100],n;
11.
12. void insertionSort();
13.
14. int main(){
15.     cout<<"\nPROGRAM MENGURUTKAN DENGAN INSERTION SORT"<<endl;
16.     cout<<"-----"<<endl;;
17.     cout<<"\tMasukkan Jumlah Data : ";cin>>n;
18.     cout<<endl;
19.     for(int i=1;i<=n;i++){
20.         cout<<"\tMasukkan data ke-"<<i<<" : ";
21.         cin>>data[i];
22.         data2[i]=data[i];
23.     }
24.     cout<<"\n-----" << endl;
25.     insertionSort();
26.     cout<<"\tData Setelah di Sort : "<<endl;
27.     cout<<"\t";
28.     for(int i=1; i<=n; i++){
29.         cout<<data[i]<<" ";
30.     }
31.     cout<<"\n-----"<<endl;;
32.     getch();
33. }
34.
35. void insertionSort(){
36.     int temp,i,j;
37.     for(i=1;i<=n;i++){
38.         temp = data[i];
39.         j = i -1;
40.         while(data[j]>temp && j>=0){
41.             data[j+1] = data[j];
42.             j--;
43.         }
44.         data[j+1] = temp;
45.     }
46. }

```

```

D:\College Shits\SEM 4\AnalGo Prak\STUFF\Insertion.exe
PROGRAM MENGURUTKAN DENGAN INSERTION SORT
-----
Masukkan Jumlah Data : 10

Masukkan data ke-1 : 0
Masukkan data ke-2 : 1
Masukkan data ke-3 : 2
Masukkan data ke-4 : 5
Masukkan data ke-5 : 5
Masukkan data ke-6 : 1
Masukkan data ke-7 : 8
Masukkan data ke-8 : 9
Masukkan data ke-9 : 6
Masukkan data ke-10 : 7

-----
Data Setelah di Sort :
0 1 1 2 5 5 6 7 8 9
-----

```

Studi Kasus 4: BUBBLE SORT

Bubble sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma bubble sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma bubble sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode master** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma bubble sort dengan menggunakan bahasa C++

Jawaban Studi Kasus 4

Subproblem = 1
Masalah setiap subproblem = $n-1$
Waktu proses pembagian = n
Waktu proses penggabungan = n

$$T(n) = \{\theta(1) T(n-1) + \theta(n)\}$$

$$\begin{aligned} T(n) &= cn + cn - c + cn - 2c + \dots + 2c + c \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + c \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + c \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + 2c \leq 2cn^2 + cn^2 \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn + cn - c + cn - 2c + \dots + 2c + c \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + c \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + c \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + 2c \leq 2cn^2 + cn^2 \\ &= \Omega(n^2) \end{aligned}$$

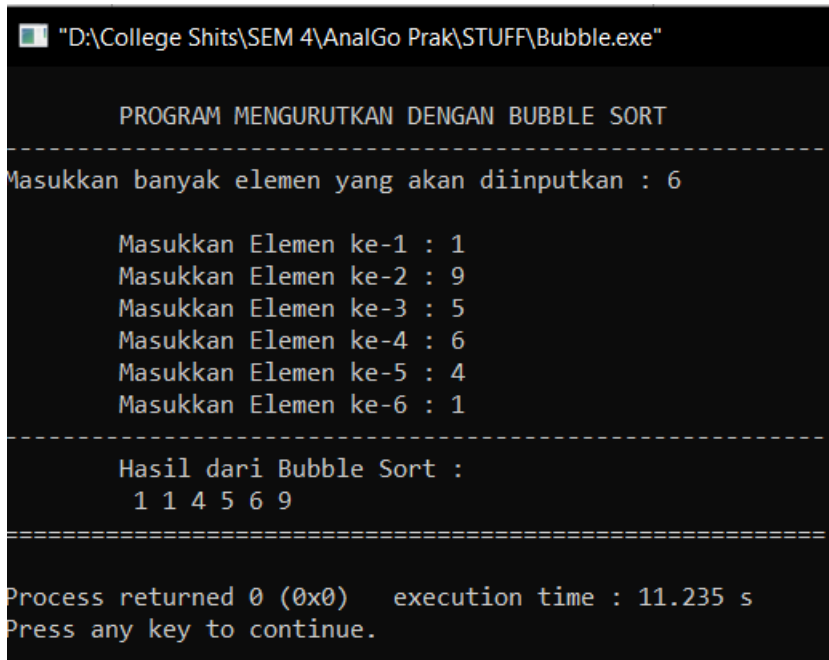
$$\begin{aligned} T(n) &= cn^2 + cn^2 \\ &= \Theta(n^2) \end{aligned}$$

```
1. /*
2. Nama      = Nadhifal A. Rendusara
3. NPM       = 140810180048
4. Kelas    = B
5. */
6. #include <iostream>
7. #include <conio.h>
8. using namespace std;
9.
10. int main(){
11.     int arr[100],n,temp;
12.     cout<<"\n\tPROGRAM MENGURUTKAN DENGAN BUBBLE SORT"<<endl;
13.     cout<<"-----"<<endl;;
14.     cout<<"Masukkan banyak elemen yang akan diinputkan : ";cin>>n;
15.     cout<<endl;
```

```

16.     for(int i=0;i<n;++i){
17.         cout<<"\tMasukkan Elemen ke-"<<i+1<<" : ";cin>>arr[i];
18.     }
19.
20.     for(int i=1;i<n;i++){
21.         for(int j=0;j<(n-1);j++){
22.             if(arr[j]>arr[j+1]){
23.                 temp=arr[j];
24.                 arr[j]=arr[j+1];
25.                 arr[j+1]=temp;
26.             }
27.         }
28.     }
29.     cout<<"-----" << endl;
30.     cout<<"\tHasil dari Bubble Sort : "<<endl;
31.     cout<<"\t";
32.     for(int i=0;i<n;i++){
33.         cout<<" "<<arr[i];
34.     }
35.     cout<<"\n===== "<<endl;
36. }

```



```

"D:\College Shits\SEM 4\AnalGo Prak\STUFF\Bubble.exe"

PROGRAM MENGURUTKAN DENGAN BUBBLE SORT
-----
Masukkan banyak elemen yang akan diinputkan : 6

Masukkan Elemen ke-1 : 1
Masukkan Elemen ke-2 : 9
Masukkan Elemen ke-3 : 5
Masukkan Elemen ke-4 : 6
Masukkan Elemen ke-5 : 4
Masukkan Elemen ke-6 : 1
-----

Hasil dari Bubble Sort :
1 1 4 5 6 9
=====

Process returned 0 (0x0)   execution time : 11.235 s
Press any key to continue.

```