Nama : Prayudha A.L
NPM : 140810180008
Tugas 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++

```
#include <iostream>
#include <chrono>
using namespace std;

void input(int *a, int &n);
void satu(int *in, int p, int q, int r);
void msort(int *in, int p, int r);

int main()
{
    int in[100];
    int n;
    input(in, n);
    auto start = chrono::steady_clock::now();
    msort(in, 1, n);
    auto end = chrono::steady_clock::now();
    cout << "Hasil: ";
    for (int i = 0; i < n; i++)
    {
        cout << in[i] << " ";
    }

    cout << endl;
    cout << "Elapsed time in nanoseconds : "
         << chrono::duration_cast<chrono::nanoseconds>(end - start).cou
nt()
         << " ns" << endl;

    return 0;
}

void input(int *a, int &n)
{
    cout << "Input banyak data: ";
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        cout << "Input angka: ";
        cin >> a[i];
    }
```

```
}

void satu(int *in, int p, int q, int r)
{
    int n1 = q - p + 1;
    int n2 = r - q;
    int L[n1 + 1];
    int R[n2 + 1];
    for (int i = 1; i <= n1; i++)
    {
        L[i - 1] = in[(p - 1) + i - 1];
    }

    for (int j = 1; j <= n2; j++)
    {
        R[j - 1] = in[(q - 1) + j];
    }

    int i = 0;
    int j = 0;
    L[n1] = 2147483647;
    R[n2] = 2147483647;

    for (int k = (p - 1); k < r; k++)
    {
        if (L[i] <= R[j])
        {
            in[k] = L[i];
            i = i + 1;
        }
        else
        {
            in[k] = R[j];
            j = j + 1;
        }
    }
}

void msort(int *in, int p, int r)
{
    int q;
    if (p < r)
    {
        q = (p + r) / 2;
        msort(in, p, q);
        msort(in, q + 1, r);

        satu(in, p, q, r);
```

```
89.     }
90.}
91.
```

```
C:\Users\Libra\Documents\MEGA\Semester 4\Analgo\Praktikum\Analgoku\Analgoku4>"studiKasus1.exe"
Input banyak data: 20
Input angka: 5
Input angka: 4
Input angka: 3
Input angka: 2
Input angka: 1
Input angka: 10
Input angka: 9
Input angka: 8
Input angka: 7
Input angka: 6
Input angka: 11
Input angka: 13
Input angka: 12
Input angka: 15
Input angka: 14
Input angka: 17
Input angka: 16
Input angka: 18
Input angka: 20
Input angka: 19
Hasil: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

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?

   O(nlogn). (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++

  Berikut Pseudocode Selection Sort

```
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
    {pertukarkan ximaks dengan xi}
    temp ← xi
    xi ← ximaks
    ximaks ← temp
endfor
```

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

$T(n) = cn + cn-c + cn-2c + ... + 2c + cn$
$= c((n-1)(n-2)/2) + cn$
$= c((n^2-3n+2)/2) + cn$
$= c(n^2/2)-(3n/2)+1 + cn$
$= O(n^2)$

$T(n) = cn + cn-c + cn-2c + ... + 2c + cn$
$= c((n-1)(n-2)/2) + cn$
$= c((n^2-3n+2)/2) + cn$
$= c(n^2/2)-(3n/2)+1 + cn$
$= \Omega(n^2)$

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

```cpp
#include <iostream>
#include <conio.h>

using namespace std;

int data1[100], data2[100];
int n;

void tukar(int a, int b);
void selection_sort();

int main()
{
    cout << "\nMasukkan Jumlah Data : ";
```

```
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cout << "Masukkan data ke-" << i << " : ";
        cin >> data1[i];
        data2[i] = data1[i];
    }

    selection_sort();
    cout << "Data Setelah di Sort : " << endl;
    for (int i = 1; i <= n; i++)
    {
        cout << " " << data1[i];
    }

    getch();
}

void tukar(int a, int b)
{
    int t;
    t = data1[b];
    data1[b] = data1[a];
    data1[a] = t;
}

void selection_sort()
{
    int pos, i, j;
    for (i = 1; i <= n - 1; i++)
    {
        pos = i;
        for (j = i + 1; j <= n; j++)
        {
            if (data1[j] < data1[pos])
                pos = j;
        }
        if (pos != i)
            tukar(pos, i);
    }
}
```

```
C:\Users\Libra\Documents\MEGA\Semester 4\Analgo\Praktikum\Analgoku\Analgoku4>"studiKasus2.exe"

Masukkan Jumlah Data : 10
Masukkan data ke-1 : 9
Masukkan data ke-2 : 8
Masukkan data ke-3 : 7
Masukkan data ke-4 : 6
Masukkan data ke-5 : 5
Masukkan data ke-6 : 3
Masukkan data ke-7 : 4
Masukkan data ke-8 : 1
Masukkan data ke-9 : 2
Masukkan data ke-10 : 10
Data Setelah di Sort :
 1 2 3 4 5 6 7 8 9 10
```

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

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

- Selesaikan persamaan rekurensi T(n) dengan metode subtitusi 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++

Berikut Pseudocode Insertion Sort

```
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
```

Subproblem = 1
Masalah setiap subproblem = n-1
Waktu proses penggabungan = n
Waktu proses pembagian = n $T(n) = \{\Theta(1)\ T(n-1) + \Theta(n)$

T(n) = cn + cn-c +cn-2c + ... + 2c +cn <= $2cn^2 + cn^2$
= c((n-1)(n-2)/2) + cn<= $2cn^2 + cn^2$
= c((n^2-3n+2)/2) + cn<= $2cn^2 + cn^2$

$= c(n^2/2)-c(3n/2)+c+cn <= 2cn^2 + cn^2$
$= O(n^2)$

$T(n) = cn <= cn$
$= \Omega(n)$

$T(n) = (cn + cn^2)/n$
$= \Theta(n)$

```
#include <iostream>
#include <conio.h>

using namespace std;

int data1[100], data2[100], n;
void insertion_sort();

int main()
{
    cout << "Masukkan Jumlah Data : ";
    cin >> n;
    cout << endl;
    for (int i = 1; i <= n; i++)
    {
        cout << "Masukkan data ke-" << i << " : ";
        cin >> data1[i];
        data2[i] = data1[i];
    }
    insertion_sort();
    cout << "\nData Setelah di Sort : " << endl;
    for (int i = 1; i <= n; i++)
    {
        cout << data1[i] << " ";
    }
    getch();
}

void insertion_sort()
{
    int temp, i, j;
    for (i = 1; i <= n; i++)
    {
        temp = data1[i];
        j = i - 1;
        while (data1[j] > temp && j >= 0)
        {
            data1[j + 1] = data1[j];
            j--;
```

```
        }
        data1[j + 1] = temp;
    }
}
```

```
C:\Users\Libra\Documents\MEGA\Semester 4\Analgo\Praktikum\Analgoku\Analgoku4>"studiKasus3.exe"
Masukkan Jumlah Data : 10

Masukkan data ke-1 : 46
Masukkan data ke-2 : 12
Masukkan data ke-3 : 34
Masukkan data ke-4 : 28
Masukkan data ke-5 : 30
Masukkan data ke-6 : 78
Masukkan data ke-7 : 111
Masukkan data ke-8 : 234
Masukkan data ke-9 : 78
Masukkan data ke-10 : 54

Data Setelah di Sort :
12 28 30 34 46 54 78 78 111 234
```

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++

$$T(n) = \{\Theta(1)\ T(n-1) + \Theta(n)$$

T(n) = cn + cn-c +cn-2c + ... + 2c +c <= $2cn^2 + cn^2$

= c((n-1)(n-2)/2) + c<= $2cn^2 + cn^2$

= c((n^2-3n+2)/2) + c<= $2cn^2 + cn^2$

= c(n^2/2)-c(3n/2)+2c <= $2cn^2 + cn^2$

=O(n^2)

T(n) = cn + cn-c +cn-2c + ... + 2c +c <= $2cn^2 + cn^2$

= c((n-1)(n-2)/2) + c<= $2cn^2 + cn^2$

= c((n^2-3n+2)/2) + c<= $2cn^2 + cn^2$

= $c(n^2/2)-c(3n/2)+2c <= 2cn^2 + cn^2$

= $\Omega (n^2)$

$T(n) = cn^2 + cn^2 = \Theta(n^2)$

```
#include <iostream>
#include <conio.h>

using namespace std;

int main()
{
    int arr[100], n, temp;
    cout << "Masukkan banyak elemen yang akan diinput : ";
    cin >> n;

    for (int i = 0; i < n; ++i)
    {
        cout << "Masukkan Elemen ke-" << i + 1 << " : ";
        cin >> arr[i];
    }

    for (int i = 1; i < n; i++)
    {
        for (int j = 0; j < (n - 1); j++)
        {
            if (arr[j] > arr[j + 1])
            {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    cout << "\nHasil dari Bubble Sort : " << endl;
    for (int i = 0; i < n; i++)
    {
        cout << " " << arr[i];
    }
}
```

```
C:\Users\Libra\Documents\MEGA\Semester 4\Analgo\Praktikum\Analgoku\Analgoku4>"studiKasus4.exe"
Masukkan banyak elemen yang akan diinput : 10
Masukkan Elemen ke-1 : 88
Masukkan Elemen ke-2 : 22
Masukkan Elemen ke-3 : 11
Masukkan Elemen ke-4 : 44
Masukkan Elemen ke-5 : 55
Masukkan Elemen ke-6 : 66
Masukkan Elemen ke-7 : 99
Masukkan Elemen ke-8 : 62
Masukkan Elemen ke-9 : 12
Masukkan Elemen ke-10 : 52

Hasil dari Bubble Sort :
 11 12 22 44 52 55 62 66 88 99
```