LAPORAN TUGAS BESAR
ANALISIS DAN KOMPLEKSITAS ALGORITMA

SELECTION SORT DAN MERGE SORT

Disusun oleh :

ANDANG SUDRAJAD	1303194073
FRANSISCO	1303194088

UNIVERSITAS TELKOM
BANDUNG
2020

I. DEFINISI DAN PSEUDOCODE

A. SELECTION SORT

Selection sort merupakan sebuah teknik pengurutan dengan cara mencari nilai tertinggi/terendah di dalam array kemudian menempatkan nilai tersebut di tempat semestinya. Algorithma ini dapat mengurutkan data dari besar ke kecil (Descending) dan kecil ke besar (Ascending).

Untuk Pseudocode dari Selection Sort sendiri adalah sebagai berikut :

```
Deklarasi:
i : integer
j : integer
n : integer
min : integer
temp : integer
A : Array[0…n-1] of integer

for i ← 0 to n – 2 do
  min ← i
   for j ← i + 1 to n – 1 do
   if A[j] < A[min]
     min ← j
   endif
   endfor
  swap A[i] and A[min]
endfor
```

B. MERGE SORT

Merge Sort merupakan algoritma pengurutan dalam ilmu komputer yang dirancang untuk memenuhi kebutuhan pengurutan atas suatu rangkaian data yang tidak memungkinkan untuk ditampung dalam memori komputer karena jumlahnya yang terlalu besar.

Algoritma pengurutan data Merge Sort dilakukan dengan menggunakan cara divide and conquer yaitu dengan memecah kemudian menyelesaikan setiap bagian kemudian menggabungkannya kembali. Pertama data dipecah menjadi 2 bagian dimana bagian pertama merupakan setengah (jika data genap) atau setengah minus satu (jika data ganjil) dari seluruh data, kemudian dilakukan pemecahan kembali untuk masing-masing blok sampai hanya terdiri dari satu data tiap blok. Setelah itu digabungkan kembali dengan membandingkan pada blok yang sama apakah data pertama lebih besar daripada data ke-tengah+1, jika ya maka data ke-tengah+1 dipindah sebagai data pertama, kemudian data ke-pertama sampai ke-tengah digeser menjadi data ke-dua sampai ke-tengah+1, demikian seterusnya sampai menjadi satu blok utuh seperti awalnya. Sehingga metode merge sort merupakan metode yang membutuhkan fungsi rekursi untuk penyelesaiannya.

Untuk Pseudocode dari Merge Sort sendiri adalah sebagai berikut :

```
function mergesort(m)
    var list left, right
if length(m) ≤ 1
    return m
else
    middle = length(m) / 2
    for each x in m up to middle
        add x to left
    for each x in m after middle
        add x to right
    left = mergesort(left)
    right = mergesort(right)
    result = merge(left, right)
    return result
```

II. C(n), T(n) dan KELAS EFISIENSI

A. SELECTION SORT

C(n) = ½ n^2
T(n) = ½ n^2 – ½ n
θϵ(n^2)

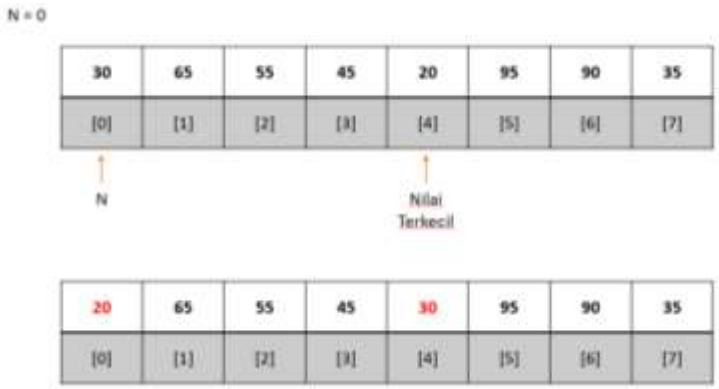
B. MERGE SORT

C(n) = 6n log2n
T(n) = 6n log2 n + 6n
θϵ(n log2 n)

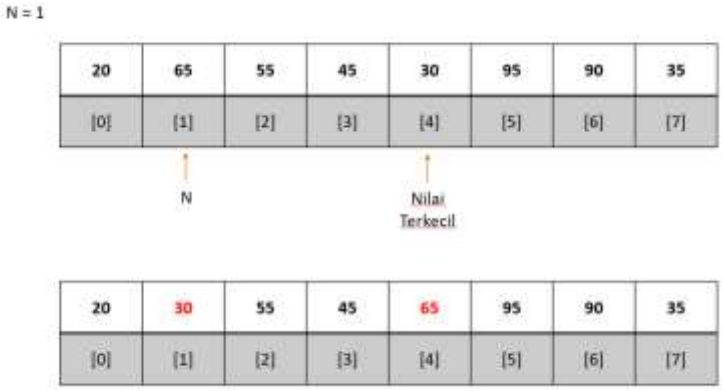
III. ILUSTRASI PROSES PENGURUTAN

A. SELECTION SORT

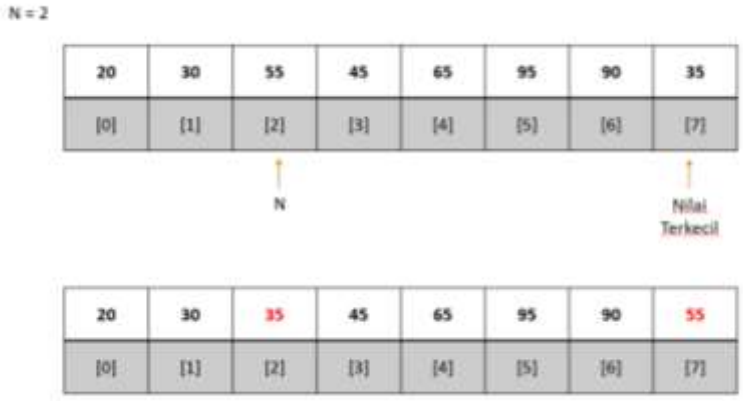
Untuk N = 0



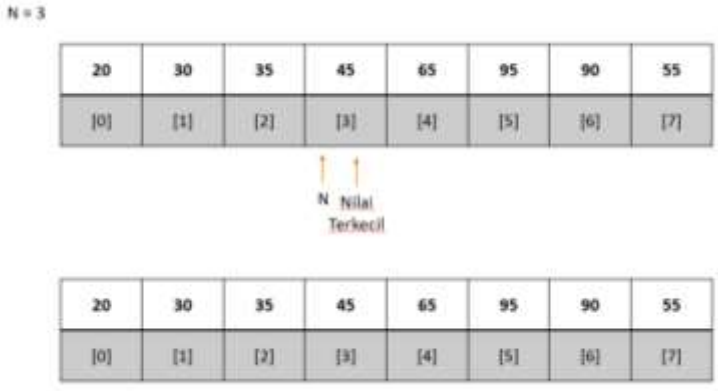
Untuk N = 1



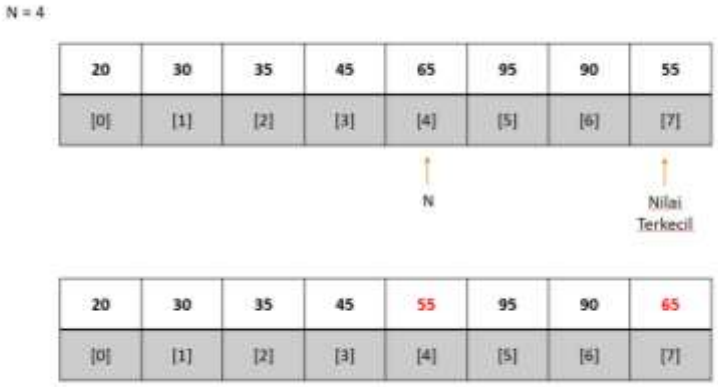
Untuk N = 2



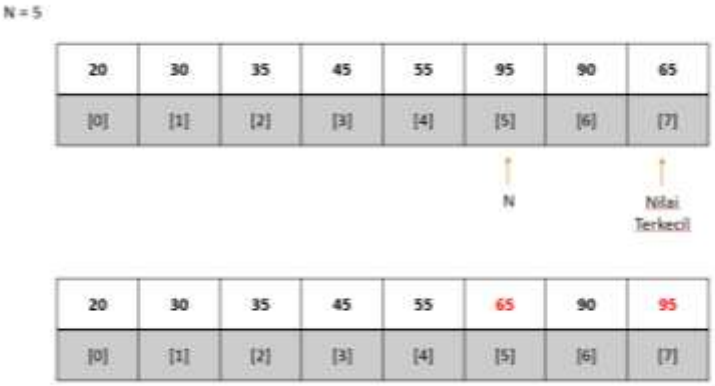
Untuk N = 3



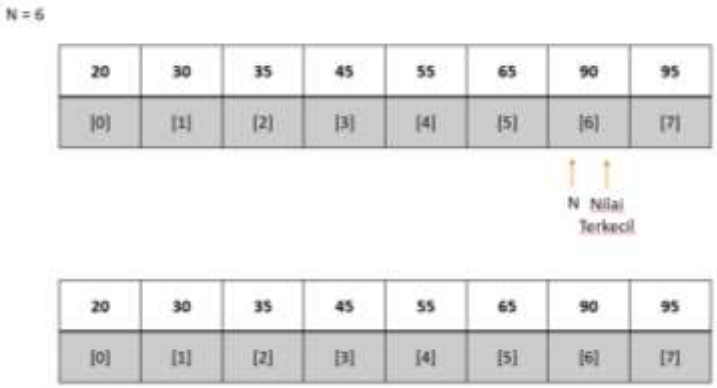
Untuk N = 4



Untuk N = 5



Untuk N = 6

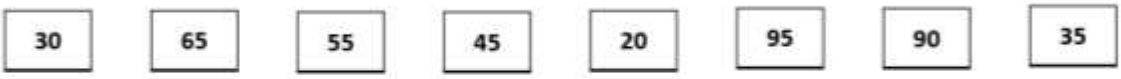


B. MERGE SORT

Pertama, Array akan dibagi menjadi 2 bagian {30,65,55,45} dan {20,95,90,35}

Kedua, Array tersebut kemudian dipisahkan kembali menjadi {30,65}, {55,45}, {20,95}, dan{90,35}

Ketiga, Array tersebut kemudian dipisahkan kembali menjadi {30} , {65}, {55}, {45}, {20}, {95}, {90}, dan{35}



Sebuah Array baru akan dibentuk sebagai penggabungan dari setiap dua Array dan diurutkan, sehingga masing-masing array memiliki nilai {30,65}, {45,55}, {20,95}, dan {35,90}

Sebuah Array baru akan dibentuk sebagai penggabungan dari setiap dua Array dan diurutkan, sehingga masing-masing array memiliki nilai {30,45,55,65}, dan {20,35,90,95}

Langkah berikutnya adalah penggabungan dari masing-masing larik ke dalam larik baru yang dibuat sebelumnya, sehingga memiliki nilai {20,30,35,45,55,65,90,95}

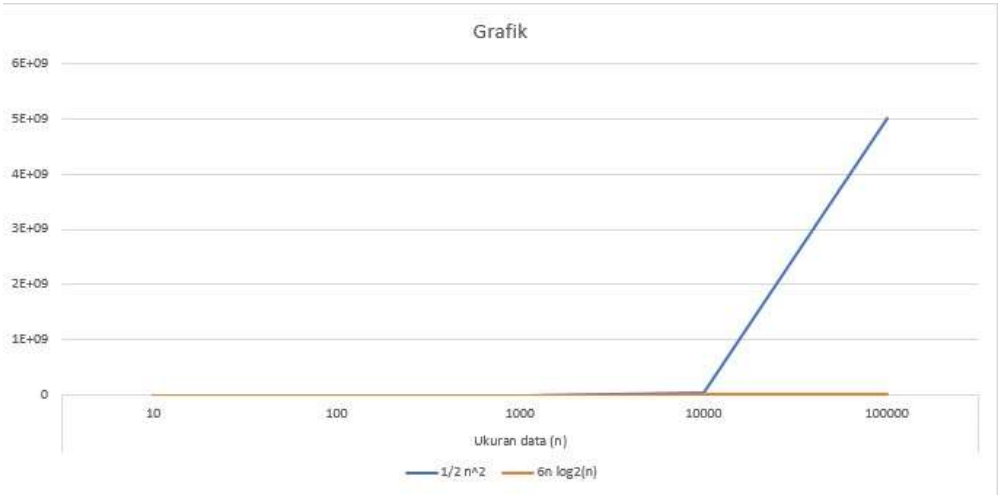
IV. TABEL DAN GRAFIK

Berikut adalah tabel dan grafik perbandingan perubahan running time dari algoritma Selection Sort dengan Merge Sort

Tabel :

	10	**100**	**1000**	**100000**	**100000**
½ n^2	50	5000	500000	50000000	5000000000
6n log2(n)	199	3986	59794	797262	9965780

Grafik :



Melalui tabel dan grafik perbandingan diatas dapat diketahui bahwa algoritma Selection Sort memiliki waktu yang lebih efisien jika memiliki nilai input yang lebih sedikit, sedangkan algoritma Merge Sort memiliki waktu lebih efisien jika memiliki nilai input yang lebih besar atau lebih banyak.

URL GITHUB : https://github.com/andangsudrajad/tubesaka/blob/main/inicodesourcenya.py

URL VIDEO : https://www.youtube.com/watch?v=Al82df6Ynzg&list=PLrXa1Xw1ZYCnxduuB3vMbbJKLHl46JIMp&index=35