Tugas Kelompok

Sorting Insertion

Nama Kelompok:  
**kodijah NIM:1124102167**

**Rachmad Lukman NIM:1124102171**

**Rafli Rahman.Efendy NIM:1124102162**

Mencari atau menjelaskan **Sorting Insertion** dan penjelasan metode nya

**Sorting Insertion**

Sorting Insertion atau lebih tepatnya Insertion Sort adalah salah satu algoritma pengurutan (sorting) yang sederhana dan mudah dipahami. Algoritma ini bekerja seperti cara kita menyusun kartu di tangan saat bermain kartu: setiap elemen baru disisipkan (insertion) ke posisi yang tepat di antara elemen-elemen yang sudah diurutkan sebelumnya.

Cara Kerja Insertion Sort:

* Mulai dari elemen kedua dalam array (karena elemen pertama dianggap sudah terurut).
* Bandingkan elemen tersebut dengan elemen-elemen sebelumnya.
* Geser elemen-elemen yang lebih besar ke kanan untuk memberi tempat.
* Sisipkan elemen pada posisi yang tepat.
* Ulangi sampai semua elemen terurut.

Contoh:

* Misalnya ingin mengurutkan array: [5, 3, 4, 1]
* Langkah 1: [5, 3, 4, 1] → bandingkan 3 dengan 5 → sisipkan → [3, 5, 4, 1]
* Langkah 2: [3, 5, 4, 1] → bandingkan 4 dengan 5 → sisipkan → [3, 4, 5, 1]
* Langkah 3: [3, 4, 5, 1] → bandingkan 1 dengan semua → sisipkan → [1, 3, 4, 5]

Kelebihan:

* Mudah diimplementasikan.
* Bagus untuk data yang hampir terurut.
* Tidak membutuhkan memori tambahan (in-place sorting).

Cara kerja:

1. Subset yang diurutkan:

Algoritma memulai dengan membagi array menjadi dua subset: subset yang sudah diurutkan dan subset yang belum diurutkan.

2. Penyisipan elemen:

Elemen dari subset yang belum diurutkan diambil dan dibandingkan dengan elemen-elemen dalam subset yang sudah diurutkan.

3. Pencarian posisi:

Elemen tersebut kemudian disisipkan ke posisi yang tepat dalam subset yang sudah diurutkan, sehingga subset tersebut tetap terurut.

4. Iterasi:

Proses ini diulangi hingga seluruh array telah diurutkan.  
Kekurangan:

* Tidak efisien untuk dataset besar (kompleksitas waktu: O(n²) pada kasus terburuk).
* Lambat untuk data yang sangat besar: Kompleksitas waktu terburuknya adalah O(n^2).

Berikut gambaran implementasi Insertion Sort:

Siklus 1:

* (70, 60, 30, 50, 40, 20) -> (60, 70, 30, 50, 40, 20)
* ( 60 , 70 , 30, 50, 40, 20)

Siklus ke-2:

* ( 60 , 70 , 30, 50, 40, 20) -> ( 60 , 30, 70 , 50, 40, 20)
* ( 60 , 30, 70 , 50, 40, 20) -> (30 , 60 , 70 , 50, 40, 20)
* ( 30 , 60 , 70 , 50, 40, 20)

Siklus ke 3 :

* ( 30 , 60 , 70 , 50, 40, 20) -> ( 30 , 60 , 50, 70, 40, 20)
* ( 30 , 60 , 50, 70, 40, 20) -> ( 30 , 50, 60 , 70 , 40, 20)
* ( 30 , 50, 60 , 70 , 40, 20) -> ( 30 , 50, 60 , 70 , 40, 20)
* ( 30 , 50 , 60 , 70 , 40, 20)

Siklus ke 4 :

* ( 30 , 50 , 60, 70, 40, 20) -> ( 30 , 50 , 60, 40, 70 , 20)
* ( 30 , 50 , 60 , 40, 70 , 20) -> ( 30 , 50 , 40, 60 , 70 , 20) ( 30 , 50 , 40, 60 , 70 , 20) ->
* ( 30 , 40 , 50 , 60 , 70 , 20) ( 30 , 40, 50 , 60 , 70 , 20 ) -> ( 30 , 40, 50 , 60 , 70 , 20)
* ( 30 , 40, 50 , 60 , 70 , 20) ( 30 , 40 , 50 , 60 , 70 , 20) ( 30 , 40, 50 , 60 , 70 , 20) ( 30 , 40 , 50 , 60 , 70 , 20)

Siklus ke 5 :

( 30 , 40 , 50 , 60 , 70 , 20) -> ( 30 , 40 , 50 , 60 , 20, 70 )

( 30 , 40 , 50 , 60 , 20, 70 ) -> ( 30 , 40 , 50 , 20, 60 , 70 )

( 30 , 40 , 50 , 20, 60 , 70 ) -> ( 30 , 40 ,20, 50 , 60 , 70 )

( 30 , 40 , 20, 50 , 60 , 70 ) -> ( 30 , 20, 40 , 50 , 60 , Bahasa Indonesia: 70 )

( 30 , 20, 40 , 50 , 60 , 70 ) -> (20, 30 , 40 , 50 , 60 , 70 )

( 20 , 30 , 40 , 50 , 60 , 70 )

**Syteam FlowChart nya:**

+----------------------+

| Mulai | ← Terminator

+----------------------+

|

v

+----------------------+

| Inisialisasi N = |

| panjang array |

+----------------------+

|

v

+----------------------+

| Set I = 1 | ← Loop luar

+----------------------+

|

v

+----------------------+

| Value = Array[I] |

+----------------------+

|

v

+----------------------+

| Set J = I - 1 | ← Loop dalam

+----------------------+

|

v

+------------------------------------------+

| Apakah J >= 0 dan Array[J] > Value? | ← Kondisi

+------------------------------------------+

| |

Ya Tidak

| |

v v

+----------------------------+ +-----------------------------+

| Geser Array[J] ke Array[J+1] | | Sisipkan Value di Array[J+1]|

| J = J - 1 | +-----------------------------+

+----------------------------+ |

| |

v v

--------------->-------------------

|

v

+----------------------+

| I = I + 1 |

+----------------------+

|

v

+---------------------------+

| Apakah I < N? |

+---------------------------+

|

Ya Tidak

| |

v v

(kembali ke Value) +----------+

| Selesai |

+----------+

**Algoritma Insertion Sort (Langkah demi Langkah)**

Misalnya kita ingin mengurutkan array A dengan panjang n secara menaik (ascending):

Langkah-langkah:

* Mulai dari elemen kedua (indeks 1) hingga elemen terakhir.
* Simpan nilai elemen saat ini dalam variabel sementara (misalnya key).
* Bandingkan key dengan elemen sebelumnya dalam array.
* Selama elemen sebelumnya lebih besar dari key, geser elemen tersebut ke kanan.
* Tempatkan key di posisi yang sesuai (setelah semua elemen yang lebih besar digeser).
* Ulangi hingga seluruh array terurut.

**Pseudocode Insertion Sort:**

|  |
| --- |
| InsertionSort(A, n):  for i = 1 to n-1:  key = A[i]  j = i - 1  while j >= 0 and A[j] > key:  A[j + 1] = A[j]  j = j - 1  A[j + 1] = key |

**Penjelasan Variabel:**

* A : array yang ingin diurutkan.
* n : jumlah elemen dalam array.
* key : elemen yang sedang diperiksa/sisipkan.
* j : indeks untuk membandingkan dan menggeser elemen sebelumnya.

**Pembacaan Flowchart:**

* Inisialisasi: Tentukan panjang array (N) dan mulai dari indeks ke-1.
* Pengambilan nilai: Simpan elemen saat ini (Value) untuk dibandingkan.
* Perbandingan: Cek apakah ada elemen sebelumnya yang lebih besar.
* Pergeseran: Geser elemen ke kanan jika lebih besar dari Value.
* Penyisipan: Tempatkan Value di posisi yang benar (lebih kecil dari elemen kanan).
* Iterasi: Ulangi proses hingga semua elemen selesai diurutkan.

Contoh Eksekusi untuk Array [9, 4, 16, 1, 5]:

**Iterasi 1 (i=1):**

4 < 9 → geser 9 → sisipkan 4

→ [4, 9, 16, 1, 5]

**Iterasi 2 (i=2):**

16 > 9 → tidak geser

→ [4, 9, 16, 1, 5]

**Iterasi 3 (i=3):**

1 < 16 → geser 16

1 < 9 → geser 9

1 < 4 → geser 4

→ sisipkan 1

→ [1, 4, 9, 16, 5]

**Iterasi 4 (i=4):**

5 < 16 → geser 16

5 < 9 → geser 9

→ sisipkan 5

→ [1, 4, 5, 9, 16]