NAMA : RAHMADITYA PUTRI LAILATUL ISMI KELAS : D4 MANAJEMEN INFORMATIKA B

LAPORAN INDIVIDU CODING "BUBBLE SORT"

♣ Bubble Sort merupakan metode sorting dengan membandingkan elemen-elemen yang ada di dalam tabel. Metode ini mendasar pada pertukaran dua buah elemen untuk mencapai keadaan urut yang diinginkan.

♣ Algoritma Bubble Sort :

- ➤ Membandingkan data ke-i dengan data ke-(i+1) (tepat bersebelahan). Jika tidak sesuai maka tukar (data ke-i = data ke-(i+1) dan data ke-(i+1) = data ke-i). Apa maksudnya tidak sesuai? Jika kita menginginkan algoritme menghasilkan data dengan urutan ascending (A-Z) kondisi tidak sesuai adalah data ke-i > data ke-i+1, dan sebaliknya untuk urutan descending (A-Z).
- ➤ Membandingkan data ke-(i+1) dengan data ke-(i+2). Kita melakukan pembandingan ini sampai data terakhir. Contoh: 1 dgn 2; 2 dgn 3; 3 dgn 4; 4 dgn 5 ...; n-1 dgn n.
- ➤ Selesai satu iterasi, adalah jika kita sudah selesai membandingkan antara (n-1) dgn n. Setelah selesai satu iterasi kita lanjutkan lagi iterasi berikutnya sesuai dengan aturan ke1. mulai dari data ke-1 dgn data ke-2, dan seterusnya.
- Proses akan berhenti jika tidak ada pertukaran dalam satu iterasi

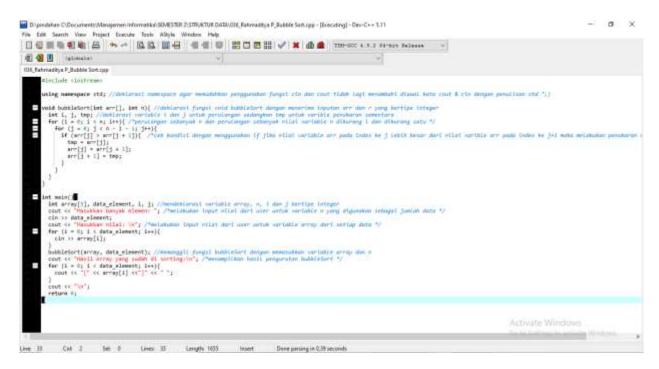
Kelebihan Bubble Sort :

- ➤ Metode Bubble Sort merupakan yang paling simple ➤ Metode Bubble Sort mudah di pahami algoritmanya ➤ Mudah untuk diubah menjadi kode.
- > Definisi terurut terdapat dengan jelas dalam algoritma.
- Cocok untuk pengurutan data dengan elemen kecil telah terurut.

Kekurangan Bubble Sort:

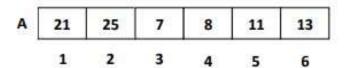
- 1. Metode Bubble Sort merupakan metode pengurutan yang paling tidak efisien
- 2. Pada saat mengurutkan data yang sangat besar akan mengalami kelambatan luar biasa

CODE BUBBLE SORT



UTPUT BUBBLE SORT

♣ Contoh proses sorting bubble sort

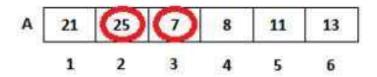


Langkah 1: dimulai dengan mengakses indeks pertama dari array dan membandingkannya dengan indeks setelahnya (indeks kedua).

A	21	25	7	8	11	13
	1	2	3	4	5	6

Karena elemen pertama tidak lebih besar dari elemen kedua, maka tidak akan dilakukan pertukaran.

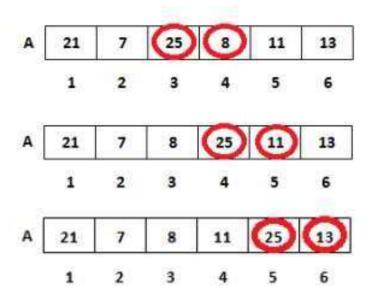
Kemudian dimulai kembali dengan membandingkan indeks kedua dengan indeks setelahnya (indeks ketiga).



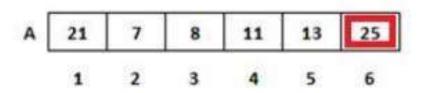
Karena elemen kedua lebih besar dari elemen ketiga, maka dilakukan pertukaran.



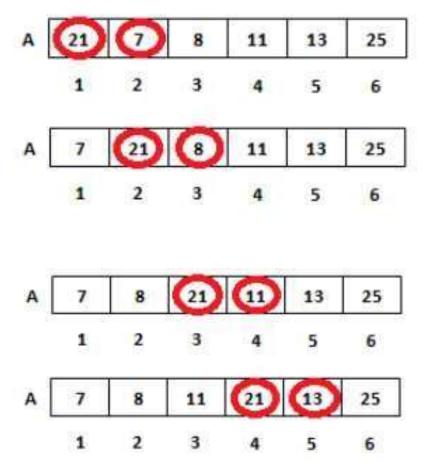
Selanjutnya, dilakukan dengan aturan yang sama. Apabila nilai di suatu indeks lebih besar dari nilai di indeks setelahnya, maka pertukaran nilai akan dilakukan.



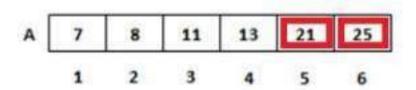
Setelah dilakukan pembandingan sebanyak 5 kali, akhirnya didapatkan hasil pengurutan langkah pertama sebagai berikut.



Langkah 2 : didapatkan hasil pembandingan sebagai berikut :



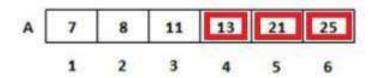
Maka, setelah dilakukan pembandingan sebanyak 4 kali, akhirnya didapatkan hasil pengurutan langkah kedua sebagai berikut.



Langkah 3 : didapatkan hasil pembandingan sebagai berikut :



Maka, setelah dilakukan pembandingan sebanyak 3 kali, akhirnya didapatkan hasil pengurutan langkah ketiga sebagai berikut.



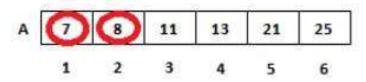
Langkah 4 : didapatkan hasil pembandingan sebagai berikut :



Maka, setelah dilakukan pembandingan sebanyak 2 kali, akhirnya didapatkan hasil pengurutan langkah keempat sebagai berikut.



Langkah 5: didapatkan hasil pembandingan sebagai berikut:



Setelah dilakukan pembandingan sebanyak 1 kali, didapatkan hasil pengurutan langkah kelima sebagai berikut.



Maka, pada akhirnya array dapat dipastikan terurut setelah langkah kelima dikerjakan.

