



PRODI SISTEM INFORMASI ITENAS

ISA-104

PEMROGRAMAN

DASAR

(3sks)

SORTING

Sofia Umaroh



TODAY'S MATERIALS

1. Sorting
2. Sorting methods

Sorting in Array

Week #11



“

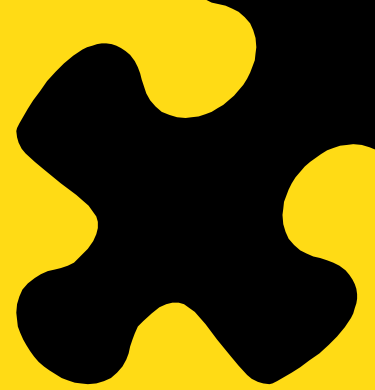
Seringkali, algoritma paling sederhana berkinerja buruk. Keutamaan mereka adalah mereka mudah ditulis, diuji, dan didebug. Algoritma yang lebih kompleks sering dibutuhkan untuk mewujudkan kinerja maksimum.

”

WHY WE NEED SORTING?

- Mengurutkan data (mis., Menempatkan data ke dalam urutan tertentu seperti naik atau turun) adalah salah satu aplikasi komputasi yang paling penting
- Mengurutkan data adalah masalah yang menarik yang telah menarik beberapa upaya penelitian paling intens di bidang ilmu komputer
- Contoh:
 - Bank mengurutkan semua transaksi harian berdasarkan no. rekening
 - Staf administrasi mengurutkan daftar hadir berdasarkan NRP
 - Etc.

PENGURUTAN DATA/ SORTING



DEFINISI

Proses menyusun kumpulan data yang seragam dengan aturan urut menaik (ascending), atau urut menurun (descending)

8	7	3	9	4	5
---	---	---	---	---	---

Unsorted Array

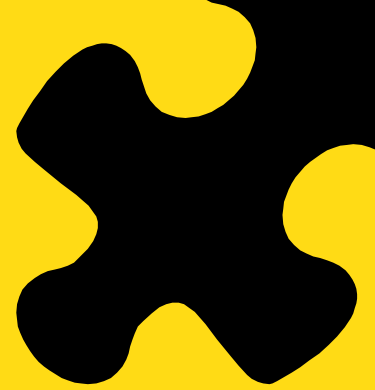
3	4	5	7	8	9
---	---	---	---	---	---

Ascending Array

9	8	7	5	4	3
---	---	---	---	---	---

Descending Array

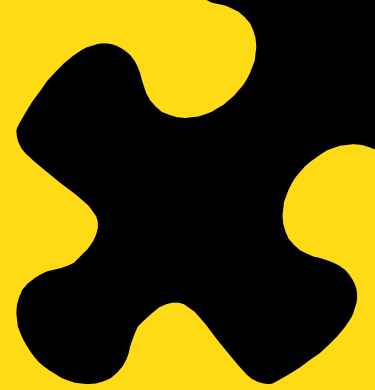
KLASIFIKASI SORTING



1) **BERDASARKAN PERBANDINGAN** (*comparison-based sorting*).

1. Pengurutan Seleksi (Selection Sort)
2. Pengurutan Sisip (Insertion Sort)
3. Pengurutan Gabung (Merge Sort)
4. Pengurutan Cepat (Quick Sort)
5. Pengurutan Himpun (Heap Sort)
6. Pengurutan Gelembung (Bubble Sort)
7. Pengurutan Shell (Shell Sort)
8. Pengurutan Pohon (Tree Sort)

KLASIFIKASI SORTING



2) BERDASARKAN PRIORITAS ANTRIAN (priority queue sorting method).

1. Pengurutan Seleksi (Selection Sort)
2. Pengurutan Himpun (Heap Sort)

3) BERDASARKAN PENYISIPAN (insert and keep sorted method).

1. Pengurutan Sisip (Insertion Sort)
2. Pengurutan Pohon (Tree Sort)

4) BERDASARKAN PEBAGIAN DAN PENGUASAAN (divide and conquer method).

1. Pengurutan Gabung (Merge Sort)
2. Pengurutan Cepat (Quick Sort)

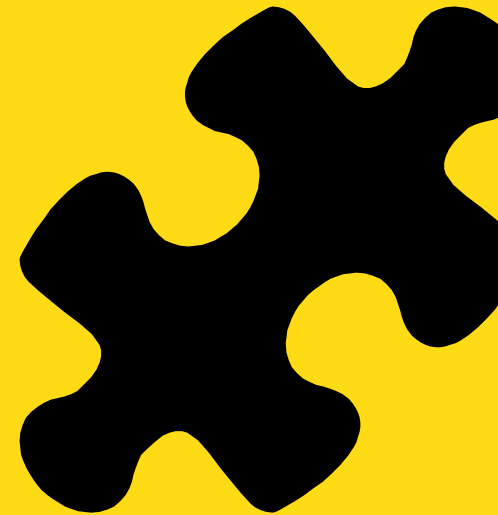


SORTING METHODS

- Insertion Sort
- Bubble Sort
- Selection Sort
- Quick Sort
- Merge Sort

1. BUBBLE SORT

- Bubble sort adalah salah satu algoritma penyortiran yang paling sederhana, tetapi bukan yang paling efisien.
- Bubble sort menukarkan dua buah elemen secara terus menerus sampai pengurutan selesai
- Selama masih terjadi penukaran ($\text{tukar} = 1$), maka terus lakukan pengurutan.





LANGKAH BUBBLE SORT

Case: Diketahui sebuah array A berisi data: 3, 2, 4, 1, 5.
Gunakan Bubble Sort untuk mengurutkan secara menaik!

First Pass:

Step 1	Step 2	Step 3	Step 4
3	2	2	2
2	3	3	3
4	4	4	1
1	1	1	4
5	5	5	5

 : an interchange

 : pair in correct order

First Pass Step: 4

First Pass Output:

2, 3, 1, 4, 5

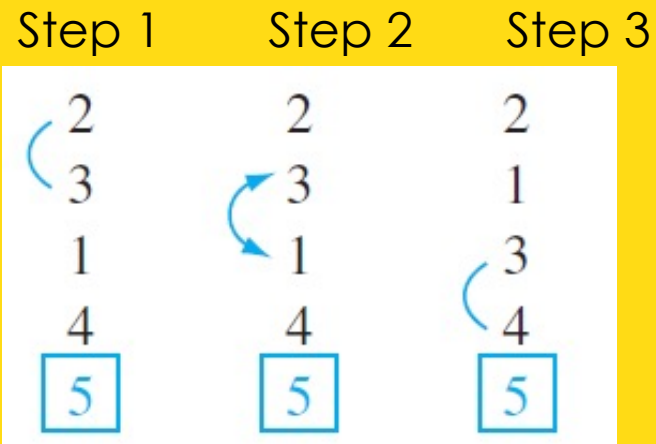
Sudah terurut?


Jika **TIDAK** lakukan **SECOND PASS.**


LANGKAH BUBBLE SORT

Case: Diketahui sebuah array A berisi data: 3, 2, 4, 1, 5.
Gunakan Bubble Sort untuk mengurutkan secara menaik!

Second Pass:



 : an interchange

 : pair in correct order

Second Pass Step: 3

First Pass Output:

2, 1, 3, 4, 5

Sudah terurut?

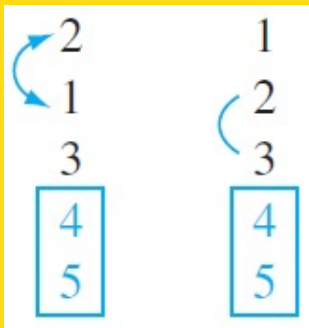
Jika **TIDAK** lakukan **THIRD PASS**.


LANGKAH BUBBLE SORT


Case: Diketahui sebuah array A berisi data: 3, 2, 4, 1, 5.
Gunakan Bubble Sort untuk mengurutkan secara menaik!

Third Pass:

Step 1 Step 2



 : an interchange

 : pair in correct order

Third Pass Step: 2

First Pass Output:

1, 2, 3, 4, 5



Sudah terurut?

Jika **TIDAK** lakukan **THIRD PASS**.

CONTOH (FIRST PASS)

LOOP	PROSES	HASIL PENGURUTAN
1	Status tukar awal = 0 Bandingkan indeks 1 dengan indeks 2. karena 5 tidak lebih besar dari 7, maka tidak ada penukaran.	 
2	Bandingkan indeks 2 dengan indeks 3. karena 7 lebih besar dari 3, maka ada penukaran. Status tukar = 1	 Tukar = 1;

CONTOH (FIRST PASS)

LOOP	PROSES	HASIL PENGURUTAN							
3	Bandungkan indeks 3 dengan indeks 4. Karena 7 lebih besar dari 4, maka ada penukaran. Status tukar = 1	<table border="1"><tr><td>5</td><td>3</td><td>7</td><td>4</td><td>9</td><td>8</td><td>6</td></tr></table> <p>1; Tukar =</p> 	5	3	7	4	9	8	6
5	3	7	4	9	8	6			
4	Bandungkan indeks 4 dengan indeks 5. karena 7 tidak lebih besar dari 9, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>5</td><td>3</td><td>4</td><td>7</td><td>9</td><td>8</td><td>6</td></tr></table> 	5	3	4	7	9	8	6
5	3	4	7	9	8	6			

CONTOH (FIRST PASS)



LOOP	PROSES	HASIL PENGURUTAN							
5	Bandungkan indeks 5 dengan indeks 6. karena 7 tidak lebih besar dari 9, maka tidak ada penukaran. Status tukar = 1	<table border="1"><tr><td>5</td><td>3</td><td>4</td><td>7</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Tukar = 1;</p> 	5	3	4	7	9	8	6
5	3	4	7	9	8	6			
6	Bandungkan indeks 6 dengan indeks 7. Karena 9 lebih besar dari 6, maka ada penukaran. Status tukar = 1	<table border="1"><tr><td>5</td><td>3</td><td>4</td><td>7</td><td>8</td><td>9</td><td>6</td></tr></table> <p>Tukar = 1;</p> 	5	3	4	7	8	9	6
5	3	4	7	8	9	6			

Status tukar terakhir FIRST LOOP = 1



CONTOH (SECOND PASS)

LOOP	PROSES	HASIL PENGURUTAN
7	Status tukar awal = 0 Bandingkan indeks 1 dengan indeks 2. karena 5 lebih besar dari 3, maka ada penukaran.	 <p>Tukar = 1;</p>
8	Bandingkan indeks 2 dengan indeks 3. karena 5 lebih besar dari 4, maka ada penukaran. Status tukar = 1	 <p>Tukar = 1;</p>

CONTOH (SECOND PASS)

LOOP	PROSES	HASIL PENGURUTAN							
9	Bandungkan indeks 3 dengan indeks 4. karena 5 tidak lebih besar dari 7, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>6</td><td>8</td><td>9</td></tr></table> 	3	4	5	7	6	8	9
3	4	5	7	6	8	9			
10	Bandungkan indeks 4 dengan indeks 5. karena 7 tidak ebih besar dari 8, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>8</td><td>6</td><td>9</td></tr></table> 	3	4	5	7	8	6	9
3	4	5	7	8	6	9			

CONTOH (SECOND PASS)



LOOP	PROSES	HASIL PENGURUTAN							
11	Bandingkan indeks 5 dengan indeks 6. karena 8 lebih besar dari 6, maka ada penukaran. Status tukar = 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>8</td><td>6</td><td>9</td></tr></table> <p>Tukar = 1;</p> 	3	4	5	7	8	6	9
3	4	5	7	8	6	9			
12	Bandingkan indeks 6 dengan indeks 7. karena 8 tidak lebih besar dari 9, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>6</td><td>8</td><td>9</td></tr></table> <p>Tukar = 1;</p> 	3	4	5	7	6	8	9
3	4	5	7	6	8	9			

Status tukar terakhir **SECOND LOOP = 1**



CONTOH (THIRD PASS)

LOOP	PROSES	HASIL PENGURUTAN
13	Status tukar awal = 0 Bandingkan indeks 1 dengan indeks 2. karena 3 tidak ebih besar dari 4, maka tidak ada penukaran.	
14	Bandingkan indeks 2 dengan indeks 3. karena 4 tidak lebih besar dari 5, maka tidak ada penukaran. Status tukar = masih 0	

CONTOH (THIRD PASS)

LOOP	PROSES	HASIL PENGURUTAN							
15	Bandungkan indeks 3 dengan indeks 4. karena 5 tidak lebih besar dari 7, maka tidak ada penukaran. Status tukar = masih 0	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>6</td><td>8</td><td>9</td></tr></table> 	3	4	5	7	6	8	9
3	4	5	7	6	8	9			
16	Bandungkan indeks 4 dengan indeks 5. karena 7 lebih besar dari 6, maka ada penukaran. Status tukar = 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>6</td><td>8</td><td>9</td></tr></table> <p>Tukar = 1;</p> 	3	4	5	7	6	8	9
3	4	5	7	6	8	9			

CONTOH (THIRD PASS)

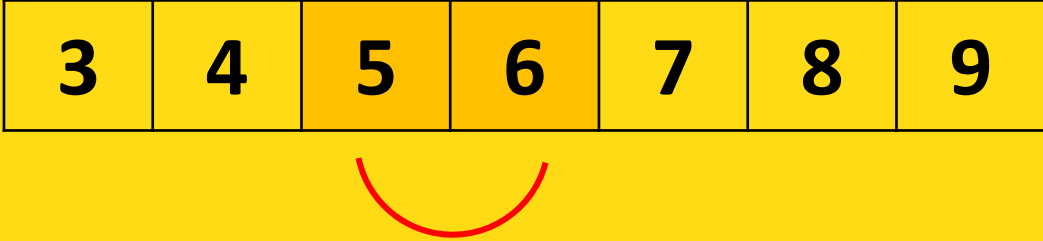
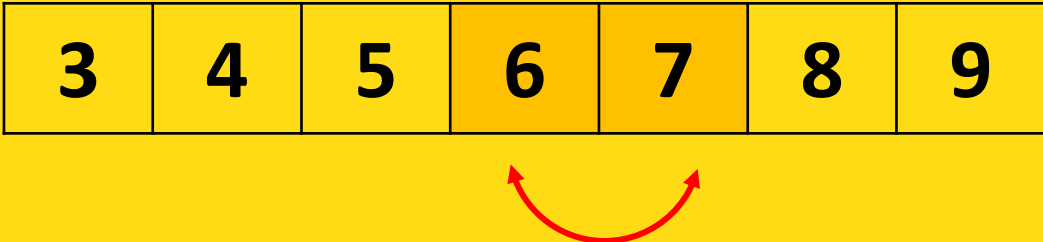
LOOP	PROSES	HASIL PENGURUTAN							
17	Bandingkan indeks 5 dengan indeks 6. karena 7 tidak lebih besar dari 8, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> 	3	4	5	6	7	8	9
3	4	5	6	7	8	9			
18	Bandingkan indeks 6 dengan indeks 7. karena 8 tidak lebih besar dari 9, maka tidak ada penukaran. Status tukar = masih 1	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> 	3	4	5	6	7	8	9
3	4	5	6	7	8	9			

Status tukar terakhir THIRD LOOP = 1



CONTOH (FORTH PASS)

LOOP	PROSES	HASIL PENGURUTAN
19	Status tukar awal = 0 Bandingkan indeks 1 dengan indeks 2. karena 3 tidak ebih besar dari 4, maka tidak ada penukaran.	
20	Bandingkan indeks 2 dengan indeks 3. karena 4 tidak lebih besar dari 5, maka tidak ada penukaran. Status tukar = masih 0	

CONTOH (FORTH PASS)

LOOP	PROSES	HASIL PENGURUTAN
21	Bandungkan indeks 3 dengan indeks 4. karena 5 tidak lebih besar dari 6, maka tidak ada penukaran. Status tukar = masih 0	
22	Bandungkan indeks 4 dengan indeks 5. karena 6 tidak lebih besar dari 7, maka tidak ada penukaran. Status tukar = masih 0	

CONTOH (FORTH PASS)

LOOP	PROSES	HASIL PENGURUTAN							
23	Bandingkan indeks 5 dengan indeks 6. karena 7 tidak lebih besar dari 8, maka tidak ada penukaran. Status tukar = masih 0	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> 	3	4	5	6	7	8	9
3	4	5	6	7	8	9			
24	Bandingkan indeks 6 dengan indeks 7. karena 8 tidak lebih besar dari 9, maka tidak ada penukaran. Status tukar = masih 0	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> 	3	4	5	6	7	8	9
3	4	5	6	7	8	9			

Status tukar terakhir FORTH LOOP = 0
SELESAI.

TOTAL LOOP: 24

ALGORITHM

BAHASA MANUSIA	ALGORITMA
Deklarasi array berisi bilangan	<pre>arrayInt : array[1..7] of <u>integer</u> arrayInt <- {5,7,3,4,9,8,6}</pre>
Deklarasi bilangan untuk counter, penyimpanan sementara (temp) dan status tukar (tukar.	<pre>i, temp, tukar: <u>integer</u></pre>

ALGORITHM

BAHASA MANUSIA	ALGORITMA
Pengulangan untuk memproses pergeseran data yang lebih besar dari dataSisip	<pre>arrayInt = {54, 22, 10, 43, 66, 48, 78} //Pengulangan Pass do //inisialisasi tukar tukar <- 0 //pengulangan untuk memeriksa apakah ada pertukaran for (i<-0 to N-1){ //jika ada nilai yang ditukar if arrayInt[i] > arrayInt[i+1] then temp <- arrayInt[i] arrayInt[i] <- arrayInt[i+1] arrayInt[i+1] <- temp tukar <- 1 {endif} {endfor} while (tukar == 1)</pre>

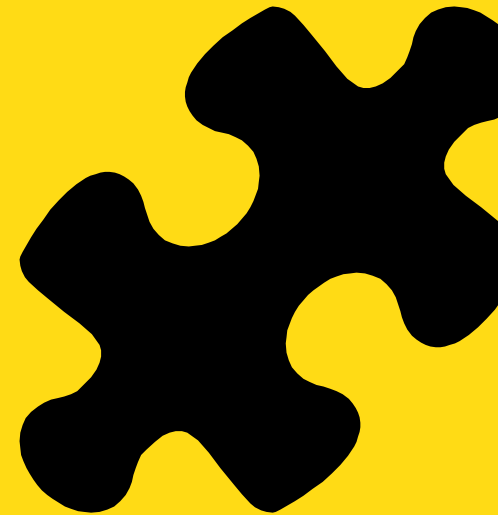
ALGORITHM

```
int a[ SIZE ] = {2, 6, 4, 8,10, 12, 89, 68, 45, 37};

// bubble sort
// loop to control number of passes
for ( pass = 1; pass < SIZE; ++pass ) {
    // loop to control number of comparisons per pass
    for (i=0; i<SIZE-1; ++i){
        // compare adjacent elements and swap them if first
        // element is greater than second element
        if (a[i]>a[i+1]){
            hold=a[i];
            a[i]= a[i+1];
            a[i+1] = hold;
        }//endif
    } // end inner for
} // end outer for
```

2. INSERTION SORT

- Metode penyisipan langsung (Insertion sort) adalah metode pengurutan data yang mengambil sebuah data sisip pada data yang diurutkan dan menggeser data yang lebih besar dari data sisip ke sebelah kanan.
- Pengulangan pada proses pengurutan dimulai dari elemen ke-2 sebanyak $N-1$
- Data sisip dimulai dari elemen array ke 2 (indeks ke-1)



CONTOH

LOOP (i)	DATA SISIP	HASIL PENGURUTAN							
1	$A[2] = 7$	<table border="1"><tr><td>5</td><td>7</td><td>3</td><td>4</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Lakukan pengecekan, apakah $data_sisip < A[2-1]$? Jika ya, maka lakukan proses penukaran: $A[2] = A[1]$ dan $A[1] = data_sisip$</p>	5	7	3	4	9	8	6
5	7	3	4	9	8	6			
2	$A[3] = 3$	<table border="1"><tr><td>5</td><td>7</td><td>3</td><td>4</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Lakukan pengecekan, 1. Apakah $data_sisip < A[3-1]$? Karena $3 < 7$, maka lakukan proses penukaran: $A[3] = A[2]$ dan $A[2] = data_sisip$ 2. Apakah $data_sisip < A[3-2]$? Karena $5 < 7$, maka lakukan proses penukaran: $A[2] = A[1]$ dan $A[1] = data_sisip$</p>	5	7	3	4	9	8	6
5	7	3	4	9	8	6			

EXAMPLE

LOOP	DATA SISIP	HASIL PENGURUTAN							
3	$A[4] = 4$	<table border="1"><tr><td>3</td><td>5</td><td>7</td><td>4</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Lakukan loop pengecekan,</p> <ol style="list-style-type: none">1. Apakah $data_sisip < A[4-1]$? Karena $4 < 7$, maka lakukan penukaran: $A[4] = A[3]$ dan $A[3] = data_sisip$2. Apakah $data_sisip < A[4-2]$? Karena $4 < 5$, maka lakukan penukaran: $A[3] = A[2]$ dan $A[2] = data_sisip$3. Apakah $data_sisip < A[4-3]$? Karena $4 > 3$, maka tidak dilakukan penukaran	3	5	7	4	9	8	6
3	5	7	4	9	8	6			

EXAMPLE

LOOP	DATA SISIP	HASIL PENGURUTAN							
4	A[5] = 9	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Lakukan loop pengecekan,</p> <ol style="list-style-type: none">1. Apakah data_sisip < A[5-1]? Karena $9 > 7$, maka tidak dilakukan penukaran2. Apakah data_sisip < A[5-2]? Karena $9 > 5$, maka tidak dilakukan penukaran3. Apakah data_sisip < A[5-3]? Karena $9 > 4$, maka tidak dilakukan penukaran4. Apakah data_sisip < A[5-4]? Karena $9 > 3$, maka tidak dilakukan penukaran	3	4	5	7	9	8	6
3	4	5	7	9	8	6			

EXAMPLE

LOOP	DATA SISIP	HASIL PENGURUTAN							
5	A[6] = 8	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>9</td><td>8</td><td>6</td></tr></table> <p>Lakukan loop pengecekan,</p> <ol style="list-style-type: none">1. Apakah data_sisip < A[6-1]? Karena $8 < 9$, maka lakukan penukaran. $A[6] = A[5]$ dan $A[5] = \text{data_sisip}$2. Apakah data_sisip < A[6-2]? Karena $8 > 7$, maka tidak dilakukan penukaran3. Apakah data_sisip < A[6-3]? Karena $8 > 5$, maka tidak dilakukan penukaran4. Apakah data_sisip < A[6-4]? Karena $9 > 4$, maka tidak dilakukan penukaran5. Apakah data_sisip < A[6-5]? Karena $8 > 4$, maka tidak dilakukan penukaran	3	4	5	7	9	8	6
3	4	5	7	9	8	6			

EXAMPLE

LOOP	DATA SISIP	HASIL PENGURUTAN							
6	A[7] = 6	<table border="1"><tr><td>3</td><td>4</td><td>5</td><td>7</td><td>8</td><td>9</td><td>6</td></tr></table> <p>Lakukan loop pengecekan,</p> <ol style="list-style-type: none">1. Apakah data_sisip < A[7-1]? Karena $6 < 9$, maka lakukan penukaran. $A[7] = A[6]$ dan $A[6] = \text{data_sisip}$2. Apakah data_sisip < A[7-2]? Karena $6 < 8$, maka lakukan penukaran. $A[6] = A[5]$ dan $A[5] = \text{data_sisip}$3. Apakah data_sisip < A[7-3]? Karena $6 < 7$, maka lakukan penukaran. $A[5] = A[4]$ dan $A[4] = \text{data_sisip}$4. Apakah data_sisip < A[7-4]? Karena $6 > 4$, maka tidak dilakukan penukaran5. Apakah data_sisip < A[6-5]? Karena $8 > 4$, maka tidak dilakukan penukaran	3	4	5	7	8	9	6
3	4	5	7	8	9	6			

EXAMPLE

LOOP	DATA SISIP	HASIL PENGURUTAN						
-	Hasil akhir	3	4	5	6	7	8	9

ALGORITHM

BAHASA MANUSIA	ALGORITMA
Deklarasi array berisi bilangan	<pre>arrayInt : array[1..7] of <u>integer</u> arrayInt <- {5,7,3,4,9,8,6}</pre>
Deklarasi bilangan untuk counter dan data sisip	<pre>i, j, dataSisip : <u>integer</u></pre>
Pengulangan untuk memproses pergeseran data yang lebih besar dari dataSisip	<pre>for (i<-1 to 7){ dataSisip <- arrayInt[i] j <- i-1 while (dataSisip < arrayInt[j]) <u>and</u> (j>=0) <u>do</u> arrayInt[j+1] <- arrayInt[j] j <- j-1 endwhile arrayInt[j+1] <- dataSisip endfor</pre>