**ALGORITMA PENGURUTAN (SORTING ALGORITHM)**

**SELECTION SORT**

Selection Sort mengurutkan array dengan berulang kali menemukan elemen minimum (mempertimbangkan urutan naik) dari bagian yang tidak disortir dan meletakkannya di awal. Algoritma mempertahankan dua sub-array dalam array yang diberikan.

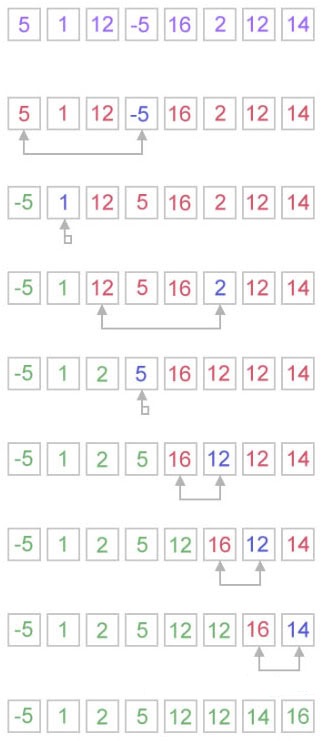
1) Subarray yang sudah diurutkan.

2) Sisa subarray yang tidak disortir.

Di setiap iterasi dari Selection Sort, elemen minimum (mempertimbangkan urutan menaik) dari subarray yang tidak disortir diambil dan dipindahkan ke subarray yang diurutkan.

Contoh 1 :

Kita dapat mengurutkan elemen array dengan menggunakan selection sort, dengan mencari elemen terkecil dan meletakkannya pada lokasi yang benar, menukar elemennya dengan bilangan yang lebiih kecil berikutnya.



arr[] = {64,25, 12, 22, 11}

// Temukan elemen minimal dalam arr[0...4] dan letakkan di awal

11 25 12 22 64

//Temukan elemen minimal dalam arr[1...4] dan letakkan di awal arr[1...4]

11 12 25 22 64

//Temukan elemen minimal dalam arr[2...4] dan letakkan di awal arr[2...4]

11 12 22 25 64

//Temukan elemen minimal dalam arr[3...4] dan letakkan di awal arr[3...4]

11 12 22 25 64

Contoh Coding :

* Jika elemen array sudah ditentukan dalam coding

**public** **class** SelectionSortExample {

**public** **static** **void** selectionSort(**int**[] arr){

**for** (**int** i = 0; i < arr.length - 1; i++)  {

**int** index = i;

**for** (**int** j = i + 1; j < arr.length; j++){

**if** (arr[j] < arr[index]){

    index = j;//searching for lowest index

  }

       }

**int** smallerNumber = arr[index];

         arr[index] = arr[i];

         arr[i] = smallerNumber;

     }

}

**public** **static** **void** main(String a[]){

**int**[] arr1 = {9,14,3,2,43,11,58,22};

        System.out.println("Before Selection Sort");

**for**(**int** i:arr1){

            System.out.print(i+" ");

        }

        System.out.println();

        selectionSort(arr1);//sorting array using selection sort

        System.out.println("After Selection Sort");

**for**(**int** i:arr1){

            System.out.print(i+" ");

        }

    }

}

**Output:**

Before Selection Sort

9 14 3 2 43 11 58 22

After Selection Sort

2 3 9 11 14 22 43 58

* Jika elemen array sudah ditentukan dalam coding dengan instansiasi objek yang menjalankan kelas SelectionSort

class SelectionSort

{

    void sort(int arr[])

    {

        int n = arr.length;

        // One by one move boundary of unsorted subarray

        for (int i = 0; i < n-1; i++)

        {

            // menemukan elemen terkecil pada array tak terurut

            int min\_idx = i;

            for (int j = i+1; j < n; j++)

                if (arr[j] < arr[min\_idx])

                    min\_idx = j;

            //menukar elemen terkecil dengan elemen pertama

            int temp = arr[min\_idx];

            arr[min\_idx] = arr[i];

            arr[i] = temp;

        }

    }

   //mencetak array

    void printArray(int arr[])

    {

        int n = arr.length;

        for (int i=0; i<n; ++i)

            System.out.print(arr[i]+" ");

        System.out.println();

    }

    public static void main(String args[])

    {

        SelectionSort ob = new SelectionSort();

        int arr[] = {64,25,12,22,11};

        ob.sort(arr);

        System.out.println("Sorted array");

        ob.printArray(arr);

    }

}

Sorted array:

11 12 22 25 64

* Jika panjang array ( banyaknya elemen) dan elemen array di masukkan sebagai input ( belum ditentukan dalam coding)

**import** java.util.Scanner;

**public** **class** SelectionSortExample2

{

**public** **static** **void** main(String args[])

   {

**int** size, i, j, temp;

**int** arr[] = **new** **int**[50];

       Scanner scan = **new** Scanner(System.in);

       System.out.print("Enter Array Size : ");

       size = scan.nextInt();

       System.out.print("Enter Array Elements : ");

**for**(i=0; i<size; i++)

       {

           arr[i] = scan.nextInt();

       }

      System.out.print("Sorting Array using Selection Sort Technique..\n");

**for**(i=0; i<size; i++)

       {

**for**(j=i+1; j<size; j++)

           {

**if**(arr[i] > arr[j])

               {

                   temp = arr[i];

                   arr[i] = arr[j];

                   arr[j] = temp;

               }

           }

       }

       System.out.print("Now the Array after Sorting is :\n");

**for**(i=0; i<size; i++)

       {

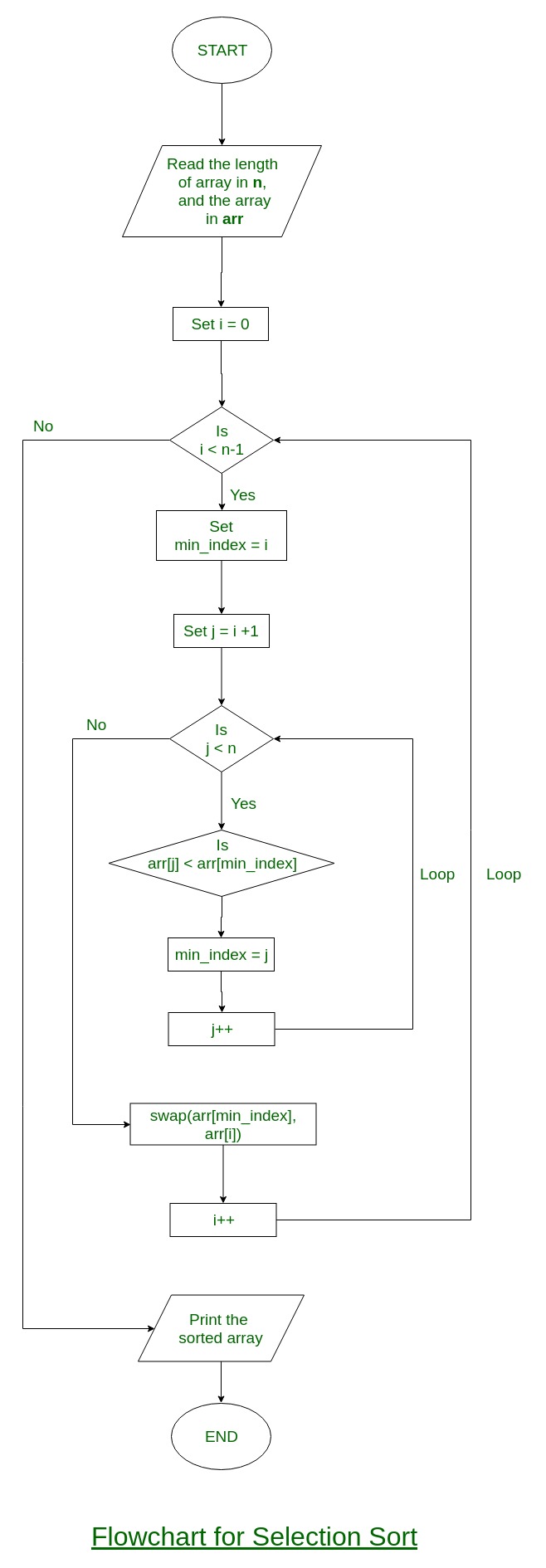
           System.out.print(arr[i]+ "  ");

       }

   }

}

Flowchart dari selesction sort



# BUBBLE SORT IN JAVA

Bubble Sort merupakan algoritma pengurutan paling tua dengan metode pengurutan paling sederhana. Pengurutan yang dilakukan dengan membandingkan masing-masing item dalam suatu list secara berpasangan, menukar item jika diperlukan, dan mengulaginya sampai akhir list secara berurutan, sehingga tidak ada lagi item yang dapat ditukar.

**Algoritma Bubble Sort :**

1. 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).
2. 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.
3. 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 ke-1. mulai dari data ke-1 dgn data ke-2, dst.
4. Proses akan berhenti jika tidak ada pertukaran dalam satu iterasi.

**Cara Kerja :**

First Pass:

( 5 1 4 2 8 ) –> ( 1 5 4 2 8 ),

algoritma membandingkan dua elemen pertaman, dan menukarnya, karena 5 > 1

( 1 5 4 2 8 ) –> ( 1 4 5 2 8 ), menukar 5 dan 4, karena 5 > 4

( 1 4 5 2 8 ) –> ( 1 4 2 5 8 ), menukar 5 dan 2, karena 5 > 2

( 1 4 2 5 8 ) –> ( 1 4 2 5 8 ), Karena elemen2 sudah terurut (8 > 5), algoritma tidak menukarkannya.

Second Pass:

( 1 4 2 5 8 ) –> ( 1 4 2 5 8 )

( 1 4 2 5 8 ) –> ( 1 2 4 5 8 ), menukar 4 dan 2, karena 4 > 2

( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )

( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )

Array sudah terurut, namun algoritma tidak mengetahui apakah sudah menyeluruh prosesnya. Algortima perlu 1 pass lagi tanpa ada penukaran untuk memastikan bahwa sudah terurut

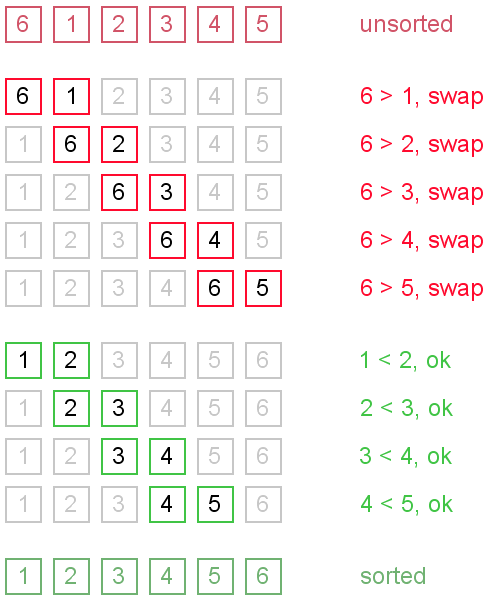
Third Pass:

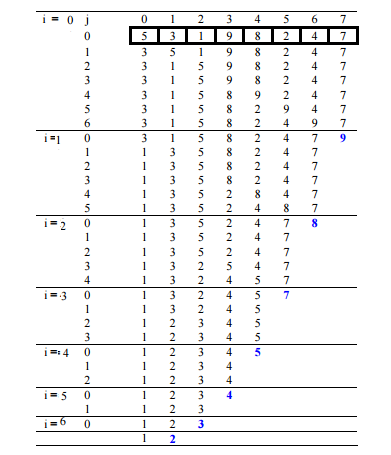
( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )

( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )

( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )

( 1 2 4 5 8 ) –> ( 1 2 4 5 8 )





Contoh Coding :

* Jika elemen array sudah ditentukan dalam coding

**public** **class** BubbleSortExample {

**static** **void** bubbleSort(**int**[] arr) {

**int** n = arr.length;

**int** temp = 0;

**for**(**int** i=0; i < n; i++){

**for**(**int** j=1; j < (n-i); j++){

**if**(arr[j-1] > arr[j]){

                                 //swap elements

                                 temp = arr[j-1];

                                 arr[j-1] = arr[j];

                                 arr[j] = temp;    }

                 }   }  }

**public** **static** **void** main(String[] args) {

**int** arr[] ={3,60,35,2,45,320,5};

                System.out.println("Array Before Bubble Sort");

**for**(**int** i=0; i < arr.length; i++){

                        System.out.print(arr[i] + " ");

               }

                System.out.println();

                bubbleSort(arr);//sorting array elements using bubble sort

                System.out.println("Array After Bubble Sort");

**for**(**int** i=0; i < arr.length; i++){

                        System.out.print(arr[i] + " ");

                }

        }   }

**Output:**

Array Before Bubble Sort

3 60 35 2 45 320 5

Array After Bubble Sort

2 3 5 35 45 60 320

* Jika elemen array sudah ditentukan dalam coding dengan instansiasi objek yang menjalankan

class BubbleSort {

    void bubbleSort(int arr[])  {

        int n = arr.length;

        for (int i = 0; i < n-1; i++)

            for (int j = 0; j < n-i-1; j++)

                if (arr[j] > arr[j+1])   {

                    // swap arr[j+1] and arr[i]

                    int temp = arr[j];

                    arr[j] = arr[j+1];

                    arr[j+1] = temp;

                }

    }

    /\* Prints the array \*/

    void printArray(int arr[])  {

        int n = arr.length;

        for (int i=0; i<n; ++i)

            System.out.print(arr[i] + " ");

        System.out.println();

    }

    public static void main(String args[]) {

        BubbleSort ob = new BubbleSort();

        int arr[] = {64, 34, 25, 12, 22, 11, 90};

        ob.bubbleSort(arr);

        System.out.println("Sorted array");

        ob.printArray(arr);

    }

}

Output:

Sorted array:

11 12 22 25 34 64 90

Kelebihan Bubble Sort :

* Metode Buble Sort merupakan metode yang paling simpel
* Metode Buble Sort mudah dipahami algoritmanya

Kelemahan Bubble Sort :

adalah pada saat mengurutkan data yang sangat besar akan mengalami kelambatan luar biasa, atau dengan kata lain kinerja memburuk cukup signifikan ketika data yang diolah jika data cukup banyak. Kelemahan lain adalah jumlah pengulangan akan tetap sama jumlahnya walaupun data sesungguhnya sudah cukup terurut. Hal ini disebabkan setiap data dibandingkan dengan setiap data yang lain untuk menentukan posisinya

**INSERTION SORT**

Insertion sort adalah sebuah metode pengurutan data dengan menempatkan setiap elemen data pada pisisinya dengan cara melakukan perbandingan dengan data – data yang ada. Ide algoritma dari metode insertion sort ini dapat dianalogikan sama seperti mengurutkan kartu, dimana jika suatu kartu dipindah tempatkan menurut posisinya, maka kartu yang lain akan bergeser mundur atau maju sesuai kondisi pemindahanan kartu tersebut. Dalam pengurutan data, metode ini dipakai bertujuan untuk menjadikan bagian sisi kiri array terurutkan sampai dengan seluruh array diurutkan.

Berikut menjelaskan bagaimana algoritma Insertion Sort bekerja dalam pengurutan kartu, Anggaplah kita ingi mengurutkan satu set kartu dari kartu yang bernilai paling kecil hingga yang paling besar.

1. Dimulai dengan posisi tangan kosong, dan semua kartu berada diatas meja. dan anggaplah kita akan menyusun kartu ke tangan kiri kita.
2. Mengamil kartu pertama dari meja dan meletakannya ke tangan kiri.
3. Mengambil kartu kedua dan membandingkannya dengan kartu yang sudah ada di tangan kiri.
4. Jika kartu yang diambil dari meja memenuhi syarat perbandingan, maka kartu tersebut akan diletakan didepan kartu yang dibandingkan, serta kartu yang lain yang telah dibandingkan akan bergeser mundur (ke belakang).

Data : 12 9 3 20 30 1

Proses Insertion Sort (Ascending)

Iterasi 1:

12  9  3  20  30  1 → (Bandingkan Data 9 dengan 12)

9  12  3  20  30  1 → (Tukar Data 9 dengan 12)

Iterasi 2:

9  12  3  20  30  1 → (Bandingkan Data 3 dengan 12)

9  3  12  20  30  1 → (Tukar 3 dengan 12. Bandingkan 3 dengan 9)

3  9  12  20  30  1 → (Tukar 3 dengan 9)

Iterasi 3:

3  9  12  20  30  1 → (Bandingkan 20 dengan 12)

3  9  12  20  30  1 → (Tidak ada pertukaran)

Iterasi 4:

3  9  12  20  30  1 → (Bandingkan 30 dengan 20)

3  9  12  20  30  1 → (Tidak ada pertukaran)

Iterasi 5:

3  9  12  20  30   1 → (Bandingkan 1 dengan 30)

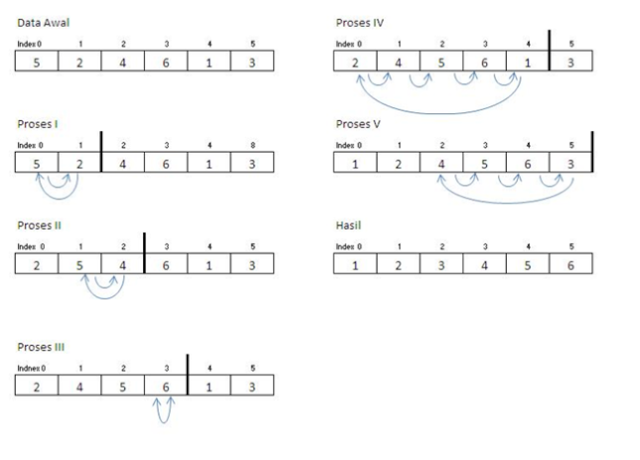
3  9  12  20  1   30 → (Tukar 1 dengan 30. Bandingkan 1 dengan 20)

3  9  12  1   20  30 → (Tukar 1 dengan 20. Bandingkan 1 dengan 12)

3  9  1  12   20  30 → (Tukar 1 dengan 12. Bandingkan 1 dengan 9)

3  1  9  12   20  30 → (Tukar 1 dengan 9. Bandingkan 1 dengan 3)

1  3  9  12   20  30 → (Tukar 1 dengan 3)

Contoh Coding :

* Jika elemen array sudah ditentukan dalam coding

**public** **class** InsertionSortExample  {

**public** **static** **void** insertionSort(**int** array[]){

**int** n = array.length;

**for** (**int** j = 1; j < n; j++) {

**int** key = array[j];

**int** i = j-1;

**while** ( (i > -1) && ( array [i] > key ) ) {

                 array [i+1] = array [i];

                 i--;

             }

             array[i+1] = key;

        }

    }

**public** **static** **void** main(String a[]){

**int**[] arr1 = {9,14,3,2,43,11,58,22};

        System.out.println("Before Insertion Sort");

**for**(**int** i:arr1){

            System.out.print(i+" ");

        }

        System.out.println();

        insertionSort(arr1);//sorting array using insertion sort

        System.out.println("After Insertion Sort");

**for**(**int** i:arr1){

            System.out.print(i+" ");

        }

    }

}

**Output:**

Before Insertion Sort

9 14 3 2 43 11 58 22

After Insertion Sort

2 3 9 11 14 22 43 58

* Jika elemen array sudah ditentukan dalam coding dengan instansiasi objek yang menjalankan

class InsertionSort {

    /\*Function to sort array using insertion sort\*/

    void sort(int arr[])

    {

        int n = arr.length;

        for (int i = 1; i < n; ++i) {

            int key = arr[i];

            int j = i - 1;

            while (j >= 0 && arr[j] > key) {

                arr[j + 1] = arr[j];

                j = j - 1;

            }

            arr[j + 1] = key;

        }

    }

    static void printArray(int arr[])

    {

        int n = arr.length;

        for (int i = 0; i < n; ++i)

            System.out.print(arr[i] + " ");

        System.out.println();

    }

    public static void main(String args[])

    {

        int arr[] = { 12, 11, 13, 5, 6 };

        InsertionSort ob = new InsertionSort();

        ob.sort(arr);

        printArray(arr);

    }

}

**Output:**

5 6 11 12 13

* Jika panjang array ( banyaknya elemen) dan elemen array di masukkan sebagai input ( belum ditentukan dalam coding)

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.IOException;

import java.util.Random;

public class InsertionSort

{

    public static void main(String[] args) throws IOException

    {

        //  Objek BufferedReader

        BufferedReader dataIn = new BufferedReader(new InputStreamReader(System.in));

        //  Input jumlah Data

        System.out.print("Masukkan jumlah Data : ");

int jlh\_data = Integer.parseInt(dataIn.readLine());

        //  Array Data untuk menampung nilai Data

        int[] data = new int[jlh\_data];

        //  Menu Pengisian data

        System.out.println("\nMenu Pengisian Data");

        System.out.println("1. Di input oleh user");

        System.out.println("2. Di isi oleh program");

        System.out.print("Pilihan : ");

int isi\_data = Integer.parseInt(dataIn.readLine());

        switch(isi\_data)

        {

            case 1  :   //  Pengisian Data oleh si User

                        System.out.println();

                        for(int a = 0; a < jlh\_data; a++)

                        {

                            System.out.print("Data ke-"+(a+1)+" : ");

data[a] = Integer.parseInt(dataIn.readLine());

                        }

                            break;

            case 2  :   //  Pengisian Data oleh program --> di isi secara acak

                        System.out.println();

                        for(int a = 0; a < jlh\_data; a++)

                            data[a] = new Random().nextInt(201);

                        //  Tampilkan Data yang di isi oleh program

                        System.out.print("Data : ");

                        for(int a = 0; a < jlh\_data; a++)

                            System.out.print(data[a]+" ");

                            break;

            default :   System.out.println("\nPilihan tidak tersedia");

        }

        //  Proses Insertion Sort

        System.out.println("\nProses Insertion Sort");

        for(int a = 0; a < jlh\_data-1; a++)

        {

            System.out.println("Iterasi "+(a+1));

            for(int b = 0; b < jlh\_data; b++)

                System.out.print(data[b]+"\t");

             System.out.print("   --> Bandingkan "+data[a+1]+" dengan "+data[a]);

              for(int b = a+1; b > 0; b--)

       {

                 String pesan = "   --> Tidak ada pertukaran";

                 if(data[b] < data[b-1])

                 {

                     pesan = "   --> "+data[b]+" tukar posisi dengan "+data[b-1];

                      //  Proses Pertukaran

                      int temp = data[b];

                     data[b] = data[b-1];

                     data[b-1] = temp;

                     System.out.println();

                     for(int c = 0; c < jlh\_data; c++){

                         System.out.print(data[c]+"\t");

                     System.out.print(pesan);

                 ` }

}

                 else

                 {

                     System.out.println();

                     for(int c = 0; c < jlh\_data; c++){

                         System.out.print(data[c]+"\t");

                      System.out.print(pesan);

                         break;

                  }

}

            }

            System.out.println("\n");

        }

        //  Tampilkan hasil Sorting

        System.out.print("\nData setelah di Sorting : ");

        for(int a = 0; a < jlh\_data; a++){

            System.out.print(data[a]+" ");

        }

    }

}