Pertemuan XIII, XIV

- **PENGURUTAN**
  - Pengertian
    - ✓ Algoritma Pengurutan dibuat untuk menghasilkan kumpulan data yang terurut.

  - Jenis
    Ada banyak jenis pengurutan. Tiga jenis yang paling sederhana adalah *Bubble Sort*, *Selection Sort*, dan *Insertion Sort*.

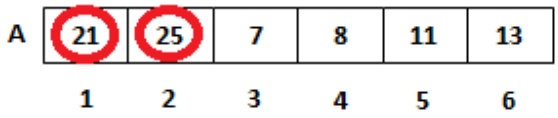
    - ✓ **Bubble Sort (**Pengurutan Gelembung / Pemberatan**)**
      Konsep pengurutan ini dilakukan dengan cara membandingkan setiap elemen dengan elemen setelahnya. Untuk membentuk urutan menaik atau menurun, elemen dipertukarkan sesuai aturan.

      Pembandingan seluruh elemen dilakukan sebanyak (N-1) kali.

      Contoh :
      Misalnya kita akan mengurutkan sebuah array A yang memiliki 3 elemen. Dalam urutan menaik, maka elemen pertama harus lebih kecil dari elemen terakhir.

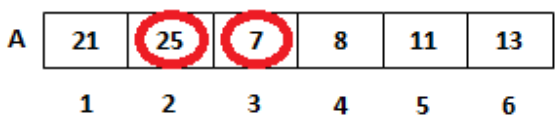
A	21	25	7	8	11	13
	1	2	3	4	5	6

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

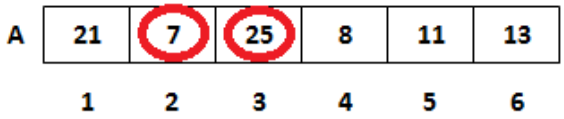


      Karena elemen pertama tidak lebih besar dari elemen kedua, maka tidak akan dilakukan pertukaran.

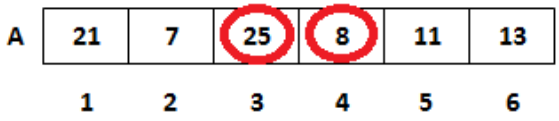
      Kemudian dimulai kembali dengan membandingkan indeks kedua dengan indeks setelahnya (indeks ketiga).

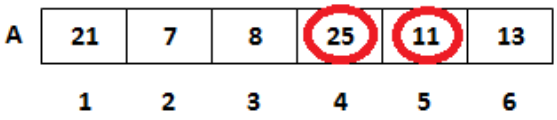


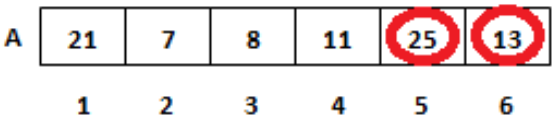
Karena elemen kedua lebih besar dari elemen ketiga, maka dilakukan pertukaran.



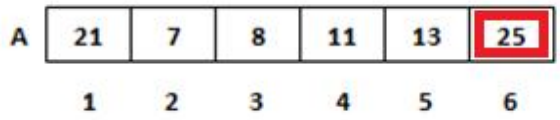
Selanjutnya, dilakukan dengan aturan yang sama. Apabila nilai di suatu indeks lebih besar dari nilai di indeks setelahnya, maka pertukaran nilai akan dilakukan.



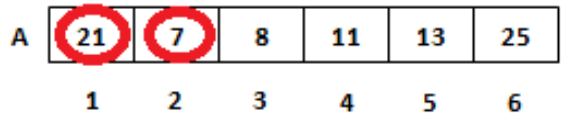


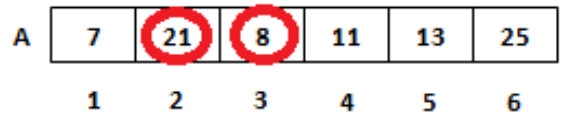


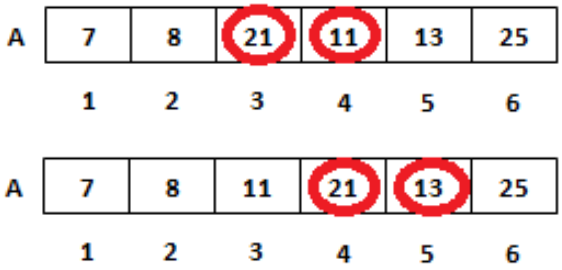
Setelah dilakukan pembandingan sebanyak 5 kali, akhirnya didapatkan hasil pengurutan langkah pertama sebagai berikut.



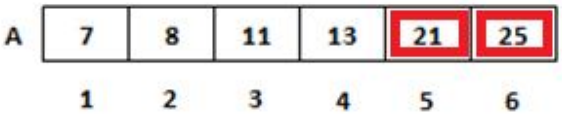
Langkah 2 : didapatkan hasil pembandingan sebagai berikut :



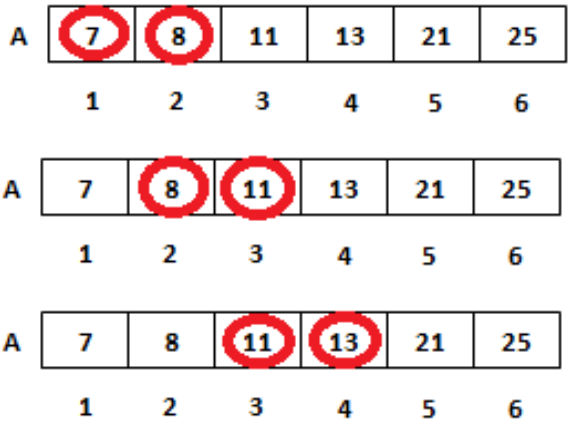




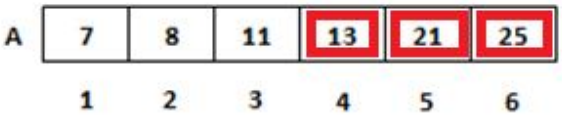
Maka, setelah dilakukan pembandingan sebanyak 4 kali, akhirnya didapatkan hasil pengurutan langkah kedua sebagai berikut.



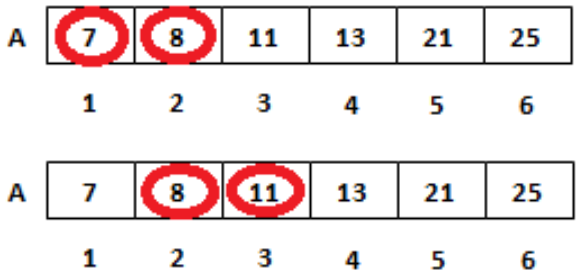
Langkah 3 : didapatkan hasil pembandingan sebagai berikut :



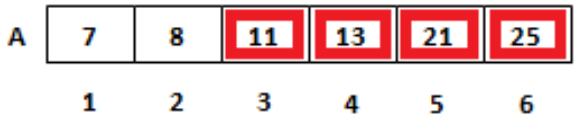
Maka, setelah dilakukan pembandingan sebanyak 3 kali, akhirnya didapatkan hasil pengurutan langkah ketiga sebagai berikut.



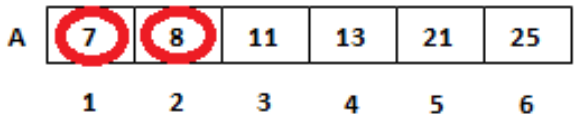
Langkah 4 : didapatkan hasil pembandingan sebagai berikut :



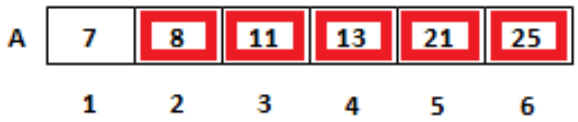
Maka, setelah dilakukan pembandingan sebanyak 2 kali, akhirnya didapatkan hasil pengurutan langkah keempat sebagai berikut.



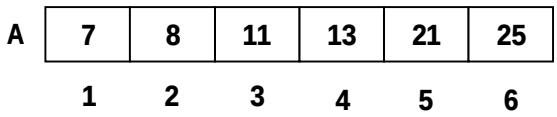
Langkah 5 : didapatkan hasil pembandingan sebagai berikut :



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



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



ALGORITMA

```
{Bubble Sort / Pengurutan Pemberatan}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

DEKLARASI

const Nmaks      = 100
type larik       = array[1..Nmaks] of integer

i,j,N,temp             : integer
A                      : larik

ALGORITMA

read(N)
for i ← 1 to N do
   write('Data ke-',i,'   : ')
   read(A[i])
endfor

{PROSES PENGURUTAN}
for i←1 to N-1 do
     for j←1 to N-i do
      if A[j] > A[j+1] then
            {Tukar}
            Temp   ← A[j]
            A[j]   ← A[j+1]
            A[j+1] ← temp
      endif
   endfor
endfor
```

PASCAL

```
{Bubble Sort / Pengurutan Gelembung}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

const Nmaks = 100;
type larik = array[1..Nmaks] of integer;

var
     i,j,N,temp              : integer;
     A                       : larik;

begin
     writeln();
     writeln('BUBBLE SORT');
```

```
    writeln('_____________________________________');
    writeln();
    writeln('Masukan Data');
    writeln('------------');
    write('Banyak Data : '); readln(N);
    writeln();

  for i:=1 to N do
      begin
        write('Data ke-',i,'   : ');
        readln(A[i]);
      end;
    writeln();


  {PROSES PENGURUTAN}
  for i:=1 to N-1 do
    begin
       for j:=1 to N-i do
          begin
            if A[j] > A[j+1] then
              begin
                {Tukar}
                Temp   := A[j];
                A[j]   := A[j+1];
                A[j+1] := temp;
              end;
          end;
    end;


    writeln();
    writeln('_____________________________________');
    writeln();
    writeln('Hasil');
    writeln('-----');

    for i:=1 to N do
     write(A[i],' | ');

    writeln();
    writeln('_____________________________________');
    readln();
end.
```

✓ **Selection Sort**

Pengurutan ini dilakukan dengan mencari terlebih dulu nilai minimum atau maksimum dari array.

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

- **Selection Sort Maksimum**

Langkah 1 : tentukan nilai Maksimum dari keenam data dalam array.
Setelah dilakukan pembandingan semua elemen, ternyata 25 adalah nilai terbesar. Maka, nilai maksimum tersebut akan ditempatkan di posisi indeks keenam (posisi elemen terakhir dari array).

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

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

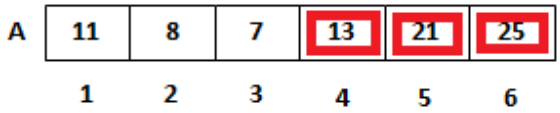
Langkah 2 : nilai maksimum dari kelima data yang tersisa (tidak termasuk data ke-6) adalah 21. Maka, nilai maksimum tersebut akan ditempatkan di posisi indeks kelima.

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

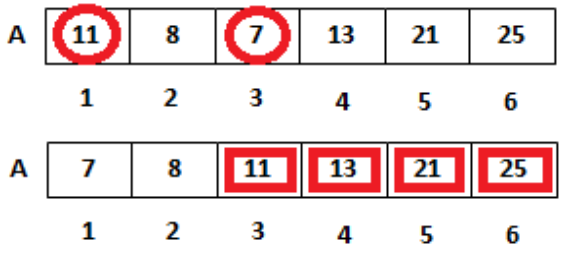
A	11	13	7	8	21	25
	1	2	3	4	5	6

Langkah 3 : nilai maksimum dari keempat data yang tersisa (tidak termasuk data ke-5 dan 6) adalah 13. Maka, nilai maksimum tersebut akan ditempatkan di posisi indeks keempat.

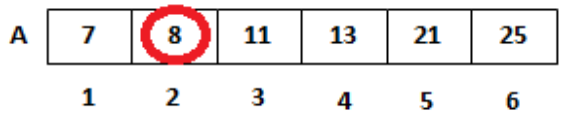
A	11	13	7	8	21	25
	1	2	3	4	5	6



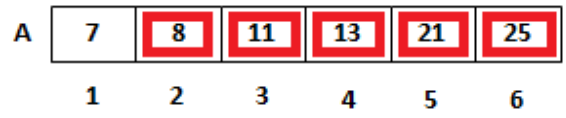
Langkah 4 : nilai maksimum dari ketiga data yang tersisa (tidak termasuk data ke-4,5, dan 6) adalah 11. Maka, nilai maksimum tersebut akan ditempatkan di posisi indeks ketiga.



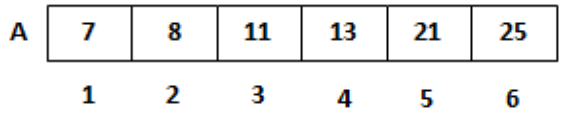
Langkah 5 : nilai maksimum dari kedua data yang tersisa (tidak termasuk data ke-3,4,5, dan 6) adalah 8. Maka, nilai maksimum tersebut akan ditempatkan di posisi indeks kedua. Pada contoh kasus, di tahap ini tidak perlu lagi dilakukan pertukaran.



Setelah dilakukan pembandingan dengan nilai maksimum sebanyak 5 kali, didapatkan hasil pengurutan langkah kelima sebagai berikut.



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



ALGORITMA

```
{Selection Sort / Pengurutan Sisip Maksimum}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

DEKLARASI

const Nmaks      = 100
type larik       = array[1..Nmaks] of integer

i,j,temp,N,maks        : integer
A                      : larik

ALGORITMA

read(N)
for i ← 1 to N do
   write('Data ke-',i,'   : ')
   read(A[i])
endfor

{PROSES PENGURUTAN}
for i←1 to N-1 do
   maks ← 1
  for j←2 to N-i+1 do
      if A[j] > A[maks] then
         maks ← j
      endif
  endfor

  {Tukar}
  Temp     ← A[N-i+1]
  A[N-i+1] ← A[maks]
  A[maks]  ← temp
endfor
```

PASCAL

```
{Selection Sort / Pengurutan Sisip}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

const Nmaks = 100;
type larik = array[1..Nmaks] of integer;

var
     i,j,temp,N,maks        : integer;
     A                      : larik;
```

```
begin
     writeln();
     writeln('SELECTION SORT');
     writeln('_____________________________________');
     writeln();
     writeln('Masukan Data');
     writeln('------------');
     write('Banyak Data : '); readln(N);
     writeln();

  for i:=1 to N do
       begin
         write('Data ke-',i,'   : ');
         readln(A[i]);
       end;
     writeln();


   {PROSES PENGURUTAN}
   for i:=1 to N-1 do
     begin
      maks := 1;
      for j:=2 to N-i+1 do
         begin
          if A[j] > A[maks] then
             maks := j;
         end;

      {Tukar}
      Temp     := A[N-i+1];
      A[N-i+1] := A[maks];
      A[maks]  := temp;
    end;

  writeln();
  writeln('_____________________________________');
  writeln();
  writeln('Hasil');
  writeln('-----');

  for i:=1 to N do
     write(A[i],' | ');

   writeln();
   writeln('_____________________________________');
   readln();
end.
```

- **Selection Sort Minimum**

Selection Sort Minimum, memiliki konsep yang sama dengan Selection Sort Maksimum, hanya saja pengurutan dilakukan dengan nilai minimum dari array.

ALGORITMA

```
{Selection Sort / Pengurutan Sisip Minimum}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

DEKLARASI

const Nmaks      = 100
type larik       = array[1..Nmaks] of integer

i,j,temp,N,min         : integer
A                      : larik

ALGORITMA

read(N)
for i ← 1 to N do
   write('Data ke-',i,'   : ')
   read(A[i])
endfor

{PROSES PENGURUTAN}
for i←1 to N-1 do
   min ← N
  for j←N-1 downto i do
      if A[j] < A[min] then
         min ← j
      endif
  endfor

  {Tukar}
  Temp     ← A[i]
  A[i]     ← A[min]
  A[min]   ← temp
endfor
```

PASCAL

```
{Selection Sort / Pengurutan Sisip}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

const Nmaks = 100;
```

```
type larik = array[1..Nmaks] of integer;

var
     i,j,temp,N,min         : integer;
     A                      : larik;

begin
     writeln();
     writeln('SELECTION SORT');
     writeln('_____________________________________');
     writeln();
     writeln('Masukan Data');
     writeln('------------');
     write('Banyak Data : '); readln(N);
     writeln();

  for i:=1 to N do
       begin
         write('Data ke-',i,'   : ');
         readln(A[i]);
       end;
     writeln();


   {PROSES PENGURUTAN}
   for i:=1 to N-1 do
     begin
      min := N;
      for j:=N-1 downto i do
         begin
          if A[j] < A[min] then
             min := j;
         end;

      {Tukar}
      Temp        := A[i];
      A[i]        := A[min];
      A[min]      := temp;
    end;

   writeln();
   writeln('_____________________________________');
   writeln();
   writeln('Hasil');
   writeln('-----');

   for i:=1 to N do
      write(A[i],' | ');

    writeln();
    writeln('_____________________________________');
    readln();
end.
```

✓ Insertion Sort

Konsep pengurutan ini dilakukan dengan menyisipkan dan menggeser data di dalam array. Data yang ada di setiap indeks array disisipkan ke dalam susunan data yang sudah terurut.

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

Langkah 1 : dilakukan dengan menganggap data di indeks pertama sudah terurut.

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

Kemudian, bandingkan data di indeks kedua dengan data yang sudah terurut.

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

Karena data kedua lebih besar dari indeks pertama, maka posisinya tidak berubah.

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

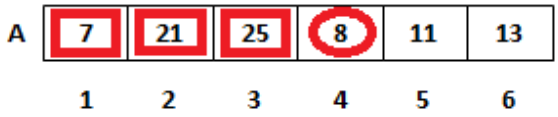
Langkah 2 : bandingkan data di indeks ketiga dengan data yang sudah terurut.

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

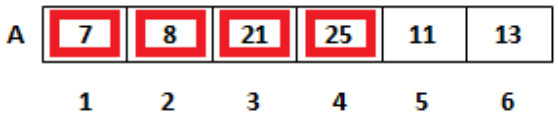
Karena data ketiga lebih kecil dari indeks pertama, maka data tersebut dipindahkan ke indeks pertama, sementara data yang sudah terurut sebelumnya, digeser ke kanan.

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

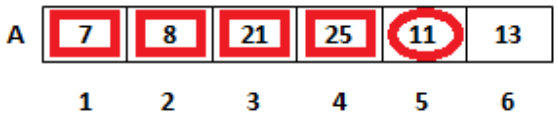
Langkah 3 : bandingkan data di indeks keempat dengan data yang sudah terurut.



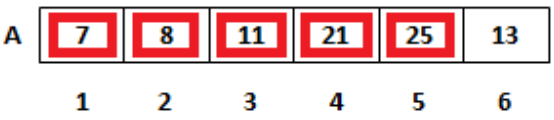
Karena data keempat lebih kecil dari indeks kedua, maka data tersebut dipindahkan ke indeks kedua, sementara data yang sudah terurut sebelumnya, digeser ke kanan dari indeks kedua.



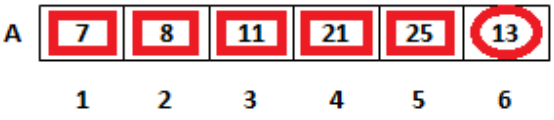
Langkah 4 : bandingkan data di indeks kelima dengan data yang sudah terurut.



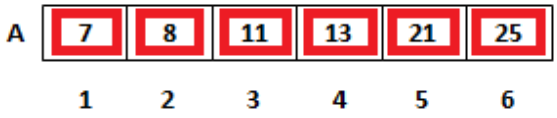
Karena data kelima lebih kecil dari indeks ketiga, maka data tersebut dipindahkan ke indeks ketiga, sementara data yang sudah terurut sebelumnya, digeser ke kanan dari indeks ketiga.



Langkah 5 : bandingkan data di indeks keenam dengan data yang sudah terurut.



Karena data keenam lebih kecil dari data keempat, maka data tersebut dipindahkan ke indeks keempat, sementara data yang sudah terurut sebelumnya, digeser ke kanan dari indeks keempat.



Dengan demikian, keseluruhan data terurut pada langkah kelima.

ALGORITMA

```
{Insertion Sort / Pengurutan Sisip}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

DEKLARASI

const Nmaks    = 100
type larik     = array[1..Nmaks] of integer

ketemu                    : boolean
i,j,N,X                   : integer
A                         : larik

ALGORITMA

read(N)
for i ← 1 to N do
   write('Data ke-',i,'   : ')
   read(A[i])
endfor

{PROSES PENGURUTAN}
for i←2 to N do

   {Masukkan nilai A[i] ke variabel bantuan X}
   x ← A[i]
   {Inisialisasi variabel ketemu dan j}
   ketemu ← FALSE
   j ← i - 1
   while ketemu and j>=1 do

      if A[j] > x then
         {geser}
         A[j+1] ← A[j]
         j ← j-1
      else
         ketemu ← TRUE
      endif

   endwhile
   {Sisipkan X ke indeks setelah j}
   A[j+1] ← x
endfor
```

PASCAL

```
{Insertion Sort / Pengurutan Sisip}

program Pengurutan;
{I.S : Data Array sudah terdefinisi}
{F.S : Data Array yang sudah diurutkan akan ditampilkan}

const Nmaks = 100;
type larik = array[1..Nmaks] of integer;

var
     ketemu                  : boolean;
     i,j,N,X                 : integer;
     A                       : larik;

begin
     writeln();
     writeln('INSERTION SORT');
     writeln('_____________________________________');
     writeln();
     writeln('Masukan Data');
     writeln('------------');
     write('Banyak Data : '); readln(N);
     writeln();

     for i:=1 to N do
       begin
         write('Data ke-',i,'   : ');
         readln(A[i]);
       end;
     writeln();

     {PROSES PENGURUTAN}
     for i:=2 to N do
        begin
            {Masukkan nilai A[i] ke variabel bantuan X}
            x := A[i];
            {Inisialisasi variabel ketemu dan j}
            ketemu := FALSE;
            j := i - 1;

            while (j>=1) and (not ketemu) do
               begin
                     if A[j] > x then
                     begin
                           {geser}
                           A[j+1] := A[j];
                           j := j-1;
                     end
                     else
                           ketemu := TRUE;
               end;
            {Sisipkan X ke indeks setelah j}
            A[j+1] := x;
        end;
```

```
        writeln();
        writeln('_____________________________________');
        writeln();
        writeln('Hasil');
        writeln('-----');

        for i:=1 to N do
            write(A[i],' | ');
        writeln();
        writeln('_____________________________________');
        readln();
end.
```