Pengurutan (*Sorting*)

Definisi Pengurutan (*Sorting*)

- Pengurutan yaitu Yaitu proses pengaturan sekumpulan objek menurut urutan atau susunan tertentu.
- Pengurutan data merupakan salah satu permasalahan umum yang juga sering dijumpai dalam pemrograman.
- Acuan pengurutan dibedakan menjadi :
  1. Ascending / menaik

     Syarat : $L[1] \leq L[2] \leq L[3] \leq \ldots \leq L[N]$
  2. Descending / menurun

     Syarat : $L[1] \geq L[2] \geq L[3] \geq \ldots \geq L[N]$

Definisi Pengurutan (*Sorting*)

- Dalam pemrograman, terdapat beberapa metode untuk melakukan pengurutan data. Namun terdapat 8 (delapan) metode yang umumnya banyak digunakan, yaitu :
  1. *Bubble Sort*
  2. *Gravitation Sort*
  3. *Selection Sort*
  4. *Insertion Sort*
  5. *Heap Sort*
  6. *Quick Sort*
  7. *Merge Sort*
  8. *Shell Sort*

Pengurutan Gelembung (*Bubble sort*)

- Diinspirasi oleh gelembung sabun yang ada dipermukaan air, dimana benda yang berat akan terbenam dan yang ringan akan terapung
- Bila pengurutan dengan acuan ascending : elemen yang bernilai besar akan "dibenamkan" melalui proses pembandingan antar elemen yang bersebelahan dan proses pertukaran
  - Proses pertukaran dilakukan sebanyak N-1 langkah, dimana N adalah ukuran array
  - Pada akhir setiap langkah ke I, array L[1..N] akan terdiri atas dua bagian yaitu :
    - Yang sudah terurut
    - Yang belum terurut
  - Setelah langkah terakhir diperoleh array L[1..N] yang terurut ascending

Pengurutan Gelembung (*Bubble sort*)

- Contoh : diurutkan secara ascending, N = 6

Lokasi	1	2	3	4	5	6
Data	25	27	10	8	76	21

Langkah 5

Lokasi	1	2	3	4	5	6
Awal	25*	27*	10	8	76	21
1	25	27*	10*	8	76	21
2	25	10	27*	8*	76	21
3	25	10	8	27*	76*	21
4	25	10	8	27	76*	21*
5	25	10	8	27	21	**76**

Langkah 4

Lokasi	1	2	3	4	5	6
Awal	25*	10*	8	27	21	**76**
1	10	25*	8*	27	21	**76**
2	10	8	25*	27*	21	**76**
3	10	8	25	27*	21*	**76**
4	10	8	25	21	**27**	**76**

Langkah 3

Lokasi	1	2	3	4	5	6
Awal	10*	8*	25	21	**27**	**76**
1	8	10*	25*	21	**27**	**76**
2	8	10	25*	21*	**27**	**76**
3	8	10	21	**25**	**27**	**76**

Pengurutan Gelembung (*Bubble sort*)

Langkah 2

Lokasi	1	2	3	4	5	6
Awal	8*	10*	21	**25**	**27**	**76**
1	8	10*	21*	**25**	**27**	**76**
2	8	10	**21**	**25**	**27**	**76**

Langkah 1

Lokasi	1	2	3	4	5	6
Awal	8*	10*	21	25	**27**	**76**
1	8	**10**	**21**	**25**	**27**	**76**

Pengurutan Gelembung (*Bubble sort*)

```
Algoritma:
Deklarasi
   I       : bilangan bulat   {untuk langkah}
   J       : bilangan bulat   {indek}
   Temp : bilangan bulat  {untuk penampung sementara}
   L           : Array [1 ..N]
   N           : bilangan bulat  {jumlah elemen array}
Deskripsi
   For I ← (N-1) downto  1 do
        For J ← 1 to I do
                        If  L[J] > L [J+1]  then
                                    Temp ← L[J]
                                    L[J] ← L[J+1]
                                    L[J+1] ← temp
                        Endif
            Endfor
   Endfor
```

Pengurutan Seleksi (*Selection Sort*)

- Yaitu memilih nilai yang maksimum/minimum dari suatu array yang akan diurutkan dan menempatkannya pada posisi awal atau akhir array; selanjutnya elemen tersebut diisolasi dan tidak disertakan pada proses berikutnya. Hal ini dilakukan secara terus menerus sampai sebanyak N-1
- Dibedakan menjadi :
  - Algoritma pengurutan maksimum

    Yaitu memilih elemen maksimum sebagai basis pengurutan
  - Algoritma pengurutan minimum

    Yaitu memilih elemen minimum sebagai basis pengurutan

Pengurutan Seleksi (*Selection Sort*)

Contoh : Diurutkan secara ascending dengan algoritma pengurutan minimum

Lokasi	1	2	3	4	5	6
Data	25	27	10	8	76	21

Langkah/ Lokasi	1	2	3	4	5	6
1	***8***	27	10	25*	76	21
2	***8***	***10***	27*	25	76	21
3	***8***	***10***	***21***	25	76	27*
4	***8***	***10***	***21***	***25****	76	27
5	***8***	***10***	***21***	***25***	***27***	76*

Pengurutan Seleksi (*Selection Sort*)

```
Algoritma :
Deklarasi :
    I       : bilangan bulat   {untuk langkah}
    J       : bilangan bulat   {indek}
    Temp : bilangan bulat  {untuk penampung sementara}
    L     : Array [1 ..N]
    N    : bilangan bulat  {jumlah elemen array}
    K    : Bilangan bulat  {menampung indek nilai terkecil}
 X   : Bilangan bulat  {menampung nilai terkecil}

Deskripsi :
    For I ← 1 to (N-1) do
        K ← I
        X ← L[I]
           For J ← ( I+1) to N do
         If  L[J] < X then
                        K ← J
                        X ← L [J]
         Endif
        Endfor
        Temp ← L[I]
        L[I] ← X
        L[K] ← temp
    Endfor
```

Pengurutan Gravitasi (Gravitasion Sort)

- Algoritma GSA (Gravitational Search Algorithm) adalah salah satu algoritma optimasi yang dapat digunakan untuk pengambilan keputusan.
- Algoritma ini terinspirasi dari sebuah teori hukum gravitasi yaitu teori Newton. Inti dari teori tersebut adalah “Setiap partikel yang ada di dunia akan saling menarik satu sama lain dengan kekuatan yang berbanding lurus dengan massa partikel dan berbanding terbalik dengan jarak antar partikel tersebut”.
- Mirip dengan Bubble Sort tetapi dimulai dari elemen pertama (paling kiri) dan dibandingkan dengan elemen di belakangnya (sebelah kanannya), sehingga pada akhir langkah pertama diperoleh elemen terakhir sudah dalam posisi terurut. Demikian seterusnya.

Pengurutan Gravitasi (Gravitasion Sort)

- Contoh : Urutkan naik elemen-elemen array A = [6, 2, 9, 3, 7, 4]

Lokasi	1	2	3	4	5	6
Data	6	2	9	3	7	4

Langkah 1

Lokasi	1	2	3	4	5	6
Awal	6*	2*	9	3	7	4
1	2	6*	9*	3	7	4
2	2	6	9*	3*	7	4
3	2	6	3	9*	7*	4
4	2	6	3	7	9*	4*
5	2	6	3	7	4	**9**

Langkah 2

Lokasi	1	2	3	4	5	6
Awal	2*	6*	3	7	4	**9**
1	2	6*	3*	7	4	**9**
2	2	3	6*	7*	4	**9**
3	2	3	6	7*	4*	**9**
4	2	3	6	4	**7**	**9**

Langkah 3

Lokasi	1	2	3	4	5	6
Awal	2*	3*	6	4	**7**	**9**
1	2	3*	6*	4	**7**	**9**
2	2	3	6*	4*	**7**	**9**
3	2	3	4	**6**	**7**	**9**

Pengurutan Gravitasi (Gravitasion Sort)

Langkah 4

Lokasi	1	2	3	4	5	6
Awal	2*	3*	4	**6**	**7**	**9**
1	2	3*	4*	**6**	**7**	**9**
2	2	3	**4**	**6**	**7**	**9**

Langkah 5

Lokasi	1	2	3	4	5	6
Awal	2*	3*	**4**	**6**	**7**	**9**
1	2	**3**	**4**	**6**	**7**	**9**

Pengurutan Menyisipkan (*Insertion Sort*)

- Yaitu metode pengurutan dengan cara menyisipkan elemen array pada posisi yang tepat
- Pada prinsipnya seperti permainan kartu : ambil kartu pertama & pegang, ambil kartu kedua dan letakkan pada posisi yang tepat / berurut, ambil kartu ketiga letakkan pada posisi yang berurut (biasa diawal, ditengah atau diakhir) dst

Pengurutan Menyisipkan (*Insertion Sort*)

Contoh :

Lokasi	1	2	3	4	5	6
Data	25	27	10	8	76	21

Langkah /Lokasi	1	2	3	4	5	6
1	25					
2	25	27				
3	10	25	27			
4	8	10	25	27		
5	8	10	25	27	76	
6	8	10	21	25	27	76

Pengurutan Menyisipkan (*Insertion Sort*)

```
Deklarasi
    I       : Bilangan bulat   {untuk langkah}
    J       : Bilangan bulat   {untuk penelusuran array}
    ketemu          : boolean {untuk menyatakan posisi penyisipan ditemukan}
    x       : Bilangan bulat {tempat sementara agar L[K] tidak ditimpa
                      selama pergeseran }
Deskripsi
    For I ← 2 to N do
        X ← L[I]
        J ← I – 1
        Ketemu ← False
        While (J ≥ 1) and (not ketemu) do
                      If  X < L[J] then
                                    L[J+1]  ← L[J]
                                    J ← J-1
                      Else ketemu ← true
                      Endif
        Endwhile
        L[J+1] ← X
    Endfor
```