SORTING

Brigida Arie Minartiningtyas, M.Kom

Sorting

- Suatu proses pengurutan data yang sebelumnya disusun secara acak sehingga menjadi tersusun secara teratur menurut suatu aturan tertentu.
- Sorting diterapkan menggunakan tipe data array agar pemahaman serta pengimplementasiannya lebih mudah.
- Pada umumnya terdapat dua jenis pengurutan :
  - Ascending (Naik).
  - Descending (Turun).

Contoh

- Data : Array [1..6] of Byte = (22, 10, 15, 3, 8, 2);
- Data Acak : 22 10 15 3 8 2
- Terurut Ascending : 2 3 8 10 15 22
- Terurut Descending : 22 15 10 8 3 2

Metode Sorting

- Untuk melakukan proses pengurutan tersebut dapat digunakan berbagai macam cara/metode.
  - Bubble / Exchange Sort
  - Selection Sort.
  - Shell Sort.
  - Quick Sort.

Buble Sort

- Paling sederhana dan paling tidak efisien
  - memerlukan waktu yang relatif lebih lama dibandingkan dengan metode-metode yang lainnya.
  - ketidakefisienan dari bubble sort yaitu harus menyelesaikan JumMax –1 dari data.

Iterasi Ke	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]
Awal	22	10	15	3	2	8
1	10	22	15	3	2	8
	10	15	22	3	2	8
	10	15	3	22	2	8
	10	15	3	2	22	8
	10	15	3	2	8	22
2	10	15	3	2	8	22
	10	3	15	2	8	22
	10	3	2	15	8	22
	10	3	2	8	15	22
	10	3	2	8	15	22
3	3	10	2	8	15	22
	3	2	10	8	15	22
	3	2	8	10	15	22
	3	2	8	10	15	22
	3	2	8	10	15	22
4	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
5	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
	2	3	8	10	15	22
Akhir	2	3	8	10	15	22

```
for last_compare_at:=x-1 downto 1 do
        begin
            for index:=1 to last_compare_at do
                begin
                    if(a[index] > a[index + 1]) then
                                begin
                                    temp := a[index];
                                    a[index] := a[index + 1];
                                    a[index + 1] := temp;
                                end;
                end;
        end;
```

Selection Sort

- Pencarian elemen dengan nilai terkecil, kemudian dilakukan penukaran dengan elemen ke-I.
  - dicari data yang terkecil dari data pertama sampai data terakhir.
  - kemudian data tersebut kita tukar dengan data pertama.

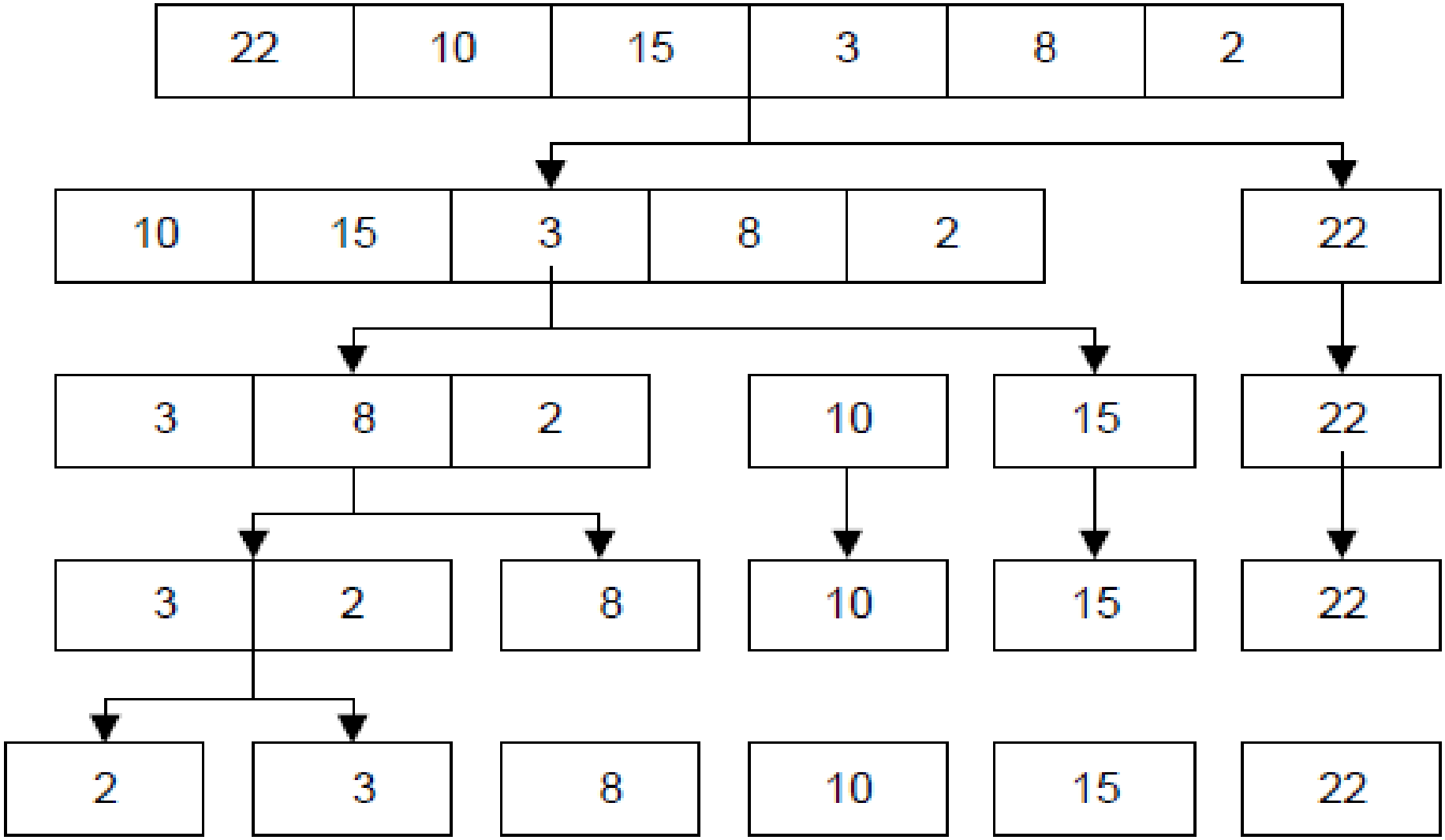
Iterasi Ke	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]
Awal	22	10	15	3	2	8
I=1, Lok=5	2	10	15	3	22	8
I=2, Lok=4	2	3	15	10	22	8
I=3, Lok=6	2	3	8	10	22	15
I=4, Lok=4	2	3	8	10	22	15
I=5, Lok=6	2	3	8	10	15	22
Akhir	2	3	8	10	15	22

```
procedure Swap(var a,b:
   Integer);
var
   t: Integer;
begin
   t := a;
   a := b;
   b := t;
end;
```

```
procedure Sorting(var d: Array100;
    c: Integer);
var
    lok, i, j: Integer;
begin
    for i:= 1 to c-1 do
    begin
            lok := i;
            for j:=i+1 to c do
            if(d[j] < d[lok]) then lok :=j;
            Swap(d[i],d[lok]);
    end;
end;
```

Quick Sort

- Misalnya kita ingin mengurutkan data A yang mempunyai N elemen. Kita pilih sembarang elemen dari data tersebut, bisanyaelemen pertama, misalnya X.
- Kemudian semua elemen tersebut disusun dengan menempatkan X pada posisi J sedemikian rupa sehingga elemen ke 1 sampai ke J-1 mempunyai nilai lebih kecil dari X dan elemen J+1 sampai ke N mempunyai nilai lebih besar dari X.
- Sampai saat ini kita sudah mempunyai dua sub data (kiri dan kanan).
- Langkah berikutnya diulang untuk setiap sub data.


22 10 15 3 8 2
10 15 3 8 2
22
3 8 2
10
15
22
3 2
8
10
15
22
2
3
8
10
15
22

```
procedure Sorting(var d: Array100; a,b: Integer);
var
    a1, b1, pivot: Integer;
begin
    a1:=a;
    b1:=b;
    pivot:= d[(a+b) div 2];
    repeat
        while(d[a1] < pivot) do Inc(a1); while(d[b1] > pivot) do
        Dec(b1);
        if (a1<=b1) then
        begin
            Swap(d[a1], d[b1]);
            Inc(a1); Dec(b1);
        end;
    until (a1 > b1);
    Tampil(d,b);
    if (a < b1) then Sorting(d,a,b1);
    if (a1 < b) then Sorting(d,a1,b);
end;
```

Searching

- Proses pencarian data dari sejumlah data yang ada.
- Pencarian data dapat dilakukan pada sejumlah data yang sudah terurut atau juga pada data yang sama sekali belum terurut.
- Metode pencarian yaitu :
  - Pencarian Berurutan (Sequential Searching).
  - Pencarian Biner (Binary Seacrh).

Sequential Searching

- Metode ini merupakan metode paling sederhana,
- Data yang dicari dibandingkan satu per satu sampai data tersebut ditemukan atau tidak ditemukan.
- Pada saat data yang dicari sudah ditemukan, maka proses pencarian langsung dihentikan.
- Tetapi jika belum ditemukan, maka pencarian diteruskan sampai seluruh data dibandingkan.

```
Procedure SEQUENTIAL_SEARCH (var Ada : Boolean;     var Posisi : Integer;
                             A : Larik;     Data : Integer;     N : Integer) ;

var   I : Integer;
begin
    Ada := false                {* pertama Dianggap  tidak ada data *}
        for   i := 1   to   N   do
        if    A[i] = Data   then
        begin                           {* Data yang dicari ketemu *}
            Ada := True;
            Posisi := i;
            Exit;
        end;
    if    Not   Ada   then
    begin                           {* Data yang dicari tidak ketemu *}
        Inc(N);
        A[N] := Data
    end
end;
```

- Urutkan dahulu sejumlah data.
- Lalu bagi dua data-data tadi dengan jumlah data yang sama pada masing-masingnya.
- Kemudian data dibandingkan dengan data terakhir dari subdata yang pertama.
- Jika data yang dicari lebih kecil, pencarian dilanjutkan pada sub data pertama dengan terlebih dahulu membagi dua lagi data-data tersebut dengan jumlah yang sama.
- Tetapi jika data yang dicari lebih besar dari data terakhir subdata pertama, berarti data yang dicari kemungkinan terletak pada subdata yang kedua.

Mencari angka 20

2	**8**	**11**	**15**	**18**	**19**	**20**	**22**	**35**	**40**	**45**

Angka 20 > 18 => berarti data 20 ada di Sub 2

2	**8**	**11**	**15**	**18**

Sub 1

19	**20**	**22**	**35**	**40**	**45**

Sub 2

Angka 20 < 22 => berarti data 20 ada di Sub 1

19	**20**	**22**

Sub 1

35	**40**	**45**

Sub 2

Angka 20 = 20 => berarti data 20 ditemukan

19	**20**

Sub 1

22

Sub 2

```
Procedure BINARY_SEARCH (var Ada : Boolean;     var Posisi : Integer;
                         A : Larik;     Data : Integer;     N : Integer) ;

var   Atas, Bawah, Tengah : Integer;
begin
    Ada := false                    {* pertama Dianggap  tidak ada data *}
    Kanan := N;                     {* Tentukan batas Kanan dan batas Kiri *}
    Kiri := 1;

    While   Kiri <= Kanan   do
        begin
            Tengah := (Atas + Bawah) div 2;
            if    Data < A[Tengah]   then
            Kanan := Tengah – 1                     {* Elemen baru dibagian pertama *}
            else if     Data > A[tengah]   then
            Kiri := Tengah + 1                {* Elemen baru dibagian kedua *}
            else
            begin                             {* Data yang dicari ketemu *}
            Ada := True;
            Posisi := Tengah;
            Bawah := Atas + 1
            end
    end
end;
```