DIKTAT STRUKTUR DATA

Oleh:
Tim Struktur Data IF

ARRAY STATIS (lanjutan)

OPERASI ARRAY STATIS (lanjutan)

3. **Pencarian (*searching*) array**

   Proses menemukan suatu data yang terdapat dalam suatu array. Proses ini menghasilkan nilai **benar atau salah**.

Metode Pencarian

1. Sequential / Linear Search
2. Binary Search

Metode Pencarian (lanjutan)

Sequential / Linear Search:

1. **Tanpa Boolean**
   a. Tanpa Sentinel
   b. Dengan Sentinel
2. **Dengan Boolean**

SEQUENTIAL SEARCH

Tanpa boolean tanpa sentinel:

1. Tidak menggunakan variabel boolean
2. Tidak mempunyai tambahan elemen di akhir array.

Sequential Search tanpa Sentinel

berikut ini terdapat array yang akan diproses:

Number	5	1	9	4	2
	[1]	[2]	[3]	[4]	[5]

Data yang akan dicari: **9**

- Number[1] = 9? i ← i + 1
- Number[2] = 9? i ← i + 1
- Number[3] = 9? i (STOP SEARCH)

hasil: **9 ditemukan pada indeks ke- [3]**

Sequential Search tanpa Sentinel

```
Procedure SeqSearchTanpaSentinel (Input nama_array:tipe_array)
{I.S. : elemen array [1..maks_array] sudah terdefinisi}
{F.S. : menampilkan hasil pencarian (ditemukan/tidak)}
Kamus:
    i : integer
    data_cari  :  tipedata
Algoritma:
    input(data_cari)
    i ← 1
    while(nama_array [i] ≠ data_cari) and (i < maks_array) do
       i ← i + 1
    endwhile
    if (nama_array[i] = data_cari)
    then
       output(data_cari,' ditemukan pada indeks ke-',i)
    else
       output(data_cari,' tidak ditemukan')
    endif
EndProcedure
```

SEQUENTIAL SEARCH (lanjutan)

Tanpa boolean dengan sentinel:

1. Tidak menggunakan variabel boolean
2. Mempunyai tambahan elemen di akhir array untuk menyimpan data cari apabila data cari tidak ditemukan

Sequential Search dengan Sentinel

Data yang dicari: 9

sentinel

	5	1	9	4	2	9
Number	[1]	[2]	[3]	[4]	[5]	[6]

Hasil: Data ditemukan pada indeks ke- 3

Data yang dicari: 10

sentinel

	5	1	9	4	2	10
Number	[1]	[2]	[3]	[4]	[5]	[6]

Result: Data tidak ditemukan

Sequential Search Use Sentinel

```
Procedure  SeqSearchSentinel (Input nama_array:tipe_array)
{I.S. : elemen array [1..maks_array] sudah terdefinisi}
{F.S. : menampilkan hasil pencarian (ditemukan/tidak)}
Kamus:
     i : integer
     data_cari  :  tipedata
Algoritma:
     input(data_cari)
     i ← 1
     nama_array(maks_array + 1) ← data_cari
     while (nama_array [i] ≠ data_cari) do
        i ← i + 1
     endwhile
     if (i < maks_array+1)
     then
        output(data_cari,' ditemukan pada indeks ke-',i)
     else
        output(data_cari,' tidak ditemukan')
     endif
EndProcedure
```

SEQUENTIAL SEARCH (lanjutan)

Dengan boolean:

1. Menggunakan variabel boolean
2. Menghasilkan nilai **TRUE** atau

   FALSE di akhir pencarian

Sequential Search dengan Boolean

berikut ini adalah array yang akan diproses:

Number	5	1	9	4	2
	[1]	[2]	[3]	[4]	[5]

Data yang akan dicari: **9**

- Number[1] = 9? KETEMU ← FALSE
- Number[2] = 9? KETEMU ← FALSE
- Number[3] = 9? KETEMU ← TRUE (STOP SEARCH)

hasil: **9 ditemukan pada indeks ke- 3**

Sequential Search Use Sentinel

```
Procedure SeqSearchBoolean (Input nama_array:tipe_array)
{I.S. : elemen array [1..maks_array] sudah terdefinisi}
{F.S. : menampilkan data yg dicari ditemukan atau tidak ditemukan}
Kamus:
     i : integer
     ketemu : boolean
     data_cari : tipedata
Algoritma:
     input(data_cari)
     i ← 1
     ketemu ← false
     while (not ketemu) and (i ≤ maks_array) do
         if(nama_var_array(i) = data_cari)
         then
              ketemu ←  true
         else
              i ← i + 1
         endif
     endwhile
     if (ketemu)
     then
         output(data_cari,' ditemukan pada indeks ke-',i)
     else
          output(data_cari,' tidak ditemukan')
     endif
EndProcedure
```

BINARY SEARCH

1. Data **harus terurut,** baik secara *ascending* atau *descending*
2. Mekanismenya adalah dengan cara **membagi larik menjadi dua bagian** yaitu **bagian kiri** (indeks terkecil/**la**) sampai ke indeks tengah dan **bagian kanan** mulai dari indeks tengah sampai indeks terbesar (**lb**)
3. **Indeks tengah** (k) : (la+lb) **div** 2 (posisi tengah larik)

BINARY SEARCH (lanjutan)

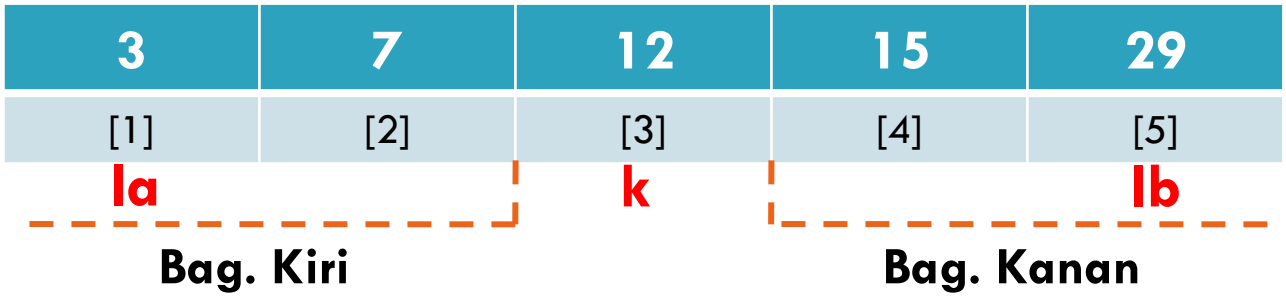
4. Jika data yang dicari **lebih kecil** dari data di posisi tengah, maka pencarian **dilanjutkan ke bagian kiri**
5. Jika data yang dicari **lebih besar** dari data di posisi tengah, maka pencarian **dilanjutkan ke bagian kanan**

KASUS BINARY SEARCH

Data yang dicari = **7**

Banyak data = **5**

Angka



Angka

KASUS BINARY SEARCH

Angka

7
[2]

lb
la
k

Jadi:

Angka 7 ditemukan pada indeks ke- 2

Binary Search

```
Procedure  BinarySearch (Input nama_array : tipe_array)
{I.S. : elemen array yang terurut secara ascending sudah terdefinisi}
{F.S. : menampilkan data yg dicari ditemukan atau tidak ditemukan}
Kamus:
      Ia, Ib, k   :  integer
      ketemu   :  boolean
      data_cari   :  tipedata
Algoritma:
      input(data_cari)
      Ia ← 1
      Ib ← maks_array
      ketemu ← false
      while (not ketemu) and (Ia ≤ Ib) do
          k  ← (Ia + Ib) div 2
          if (nama_var_array[k] = data_cari)
            then
               ketemu ←  true
            else
            if (nama_var_array[k] < data_cari)
             then
                 Ia  ←  k + 1
             else
                 Ib  ←  k – 1
            endif
          endif
      endwhile
Endprocedure
```

OPERASI ARRAY STATIS (lanjutan)

4. **Pengurutan (*sorting*) array**
   a. Bubble Sort
   b. Selection Sort
   c. Insertion Sort
   d. Radix Sort
   e. Merge Sort
   f. Quick Sort

TUGAS