13 SEARCHING

Searching (Pencarian) merupakan proses menemukan suatu nilai(data) tertentu di dalam sekumpulan data yang bertipe sama(baik bertipe dasar atau bertipe bentukan)

Metode-metode Pencarian :

1. Pencarian Beruntun(*SEQUENTIAL SEARCH*)

Proses pencarian dengan metode Sequential Search adalah dengan melakukan perbandingan nilai yang dicari dengan setiap elemen pada array mulai dari indeks terkecil sampai indeks terbesar. Pencarian dihentikan jika nilai yang dicari telah ditemukan atau semua elemen array sudah diperiksa.

Ilustrasi Kasus :

Terdapat sebuah Array *TabInt* yang terdiri dari n=10 elemen

Isi Tabel	7	5	23	1	15	8	17	75	10	4
Indeks	1	2	3	4	5	6	7	8	9	10

Kasus 1 : Misalkan nilai yang dicari adalah x = 7
Elemen yang dibandingkan : 7 (ditemukan)
Karena data langsung ditemukan maka pengulangan dihentikan
indeks larik yang dikembalikan : idx = 1

Kasus 2 : Misalkan nilai yang dicari adalah x = 17
Elemen yang dibandingkan : 7, 5, 23, 1, 15, 8, 17 (ditemukan)
Setelah data ditemukan pengulangan dihentikan
indeks larik yang dikembalikan : idx = 7

Kasus 3 : Misalkan nilai yang dicari adalah x = 25
Elemen yang dibandingkan: 7, 5, 23, 1, 15, 8, 17, 75, 10, 4 (tidak ditemukan)
Karena pencarian data sudah mencapai indeks terbesar dan nilai yang dicari belum ditemukan maka pencarian dihentikan
indeks larik yang dikembalikan : idx = 0

Algoritma Sequential Search

```
Algoritma Sequential_Search
Kamus
    const Nmax=100
    type TabInteger = array[1..NMax] of integer

    TabInt : TabInteger
    jml_data, x, i : integer
Deskripsi
    read(jml_data)                    {Banyaknya integer}
    for i ← 1 to jml_data do          {Awal Proses Input Data ke Array}
        read(TabInt[i])
    endfor                                        {Akhir Proses Input Data ke Array}

    read (x)                  {membaca data yang akan dicari}
    i ← 1                                         {Awal Prosedur Sequential Search}
    while (i<jml_data) and (TabInt[i] ≠ x) do
            i ← i+1
    endwhile
    { perulangan berhenti jika i ≥ jml_data atau TabInt[i] = x}

    if (TabInt[i] = x) then
        write('Data ditemukan pada posisi ke - ',i)
    else
        write('Data tidak ditemukan!')  {Akhir Prosedur Sequential Search}
    endif
```

Salah satu ciri program yang baik adalah program tersebut memiliki sifat prosedural. Tujuannya untuk memudahkan dalam pengembangan program, menghemat ukuran program(jika ada beberapa instruksi yang sama digunakan pada beberapa tempat dalam program), mempermudah pembacaan program, dan mempermudah pendeteksian kesalahan pada program. Algoritma di atas bukan algoritm yang prosedural karena belum membagi bagian-bagian tertentunya menjadi prosedur atau fungsi. Algoritma berikut ini merupakan modifikasi dari algoritma di atas dan sifatnya lebih prosedural

Implementasi Algoritma Sequential Search dengan Prosedur

```
procedure InputData(input n : integer , output T : TabInteger)
Kamus
    i : integer
Deskripsi
        for i ← 1 to n do
            read(T[i])
        endfor


procedure SeqSearch(input T : TabInteger; input n , x : integer ;
                                output idx : integer)
Kamus
    i : integer
Deskripsi
        i ← 1
        while (i<n) and (T[i]≠x) do
                i ← i+1
        endwhile

        if (T[i] = x) then
            idx ← i
        else
            idx ← 0
        endif
```

```
Algoritma Sequential_Search2
Kamus
      const Nmax=100
      type TabInteger = array[1..NMax] of integer
      TabInt : TabInteger
      jml_data, data, indeks : integer
      cari : char
Deskripsi
    read(jml_data)          {banyaknya data yang diinputkan}
    InputData(jml_data,TabInt) {panggil prosedur InputData}

  repeat
        read(data)          {memasukkan data yang akan dicari}
        SeqSearch(TabInt, jml_data, data, indeks) {panggil prosedur SeqSearch}
        if (indeks = 0) then
            write('Data tidak ditemukan!')
        else
            write('Data ditemukan pada posisi ke-',indeks)
        endif
        write('Cari data lagi(y/t) ? ') read (cari)
  until (cari='t') or (cari='T')
```

Algoritma Sequential_Seach2 memanggil dua prosedur yaitu prosedur InputData(dengan parameter jml_data sebagai input untuk **parameter input n** dan TabInt sebagai output untuk **parameter output T**) dan prosedur SeqSearch(dengan parameter TabInt sebagai input untuk T, jml_data sebagai input untuk n, data sebagai input untuk x, dan indeks sebagai output untuk idx)

Algoritma pencarian dengan metode Sequential Search memiliki banyak versi tergantung kebutuhan akan output, kreatifitas pembuat algoritma, dan faktor lainnya. Procedure di bawah ini adalah versi lain dari algoritma Sequential Search

Prosedur Sequential Search versi Boolean

```
procedure SeqSearch2(input T : TabInteger ;  input n , x : integer ;
                                output idx : integer ; output found : boolean)
Kamus
    i : integer
Deskripsi
    i ← 1
    found ← false
    while (i ≤ n) and (not found) do
            if (T[i] = x) then  {jika isi T[i] = x, nilai found diubah menjadi true}
                found←true
            else          {jika isi T[i] ≠ x, pencacah indeks array bertambah 1}
                i ← i+1
            endif
    endwhile
    {perulangan berhenti jika i > n atau found=true}
    if (found) then                {jika found=true}
        idx ← i
    else                                        {jika i > n}
        idx ← 0
    endif
```

2. Pencarian Bagi Dua(Binary Search)

Pencarian bagi dua adalah metode pencarian yang diterapkan pada sekumpulan data yang sudah terurut baik menaik maupun menurun. Maksud dari metode ini adalah mempersingkat waktu pencarian data/nilai pada tabel.

Proses pencarian :

1. Bandingkan nilai yang kita cari(x) dengan data yang berada pada posisi tengah. Jika sama, maka pencarian selesai.
2. Jika x lebih kecil daripada data pada posisi tengah, pencarian dilakukan pada daerah yang data-datanya lebih kecil dari data tengah
3. Jika x lebih besar daripada data pada posisi tengah, pencarian dilakukan pada daerah yang data-datanya lebih besar dari data tengah

4. Kembali ke proses nomor 1 jika masih ada daerah pencarian. Jika tidak ada, berarti data tidak ditemukan.

Prosedur Binary Search

```
Procedure BinarySearch(input T : TabInteger; input n , x : integer ;
                       output idx : integer)
Kamus
    BatasAtas, BatasBawah, Tengah : integer
Deskripsi
    BatasAtas ← 1
    BatasBawah ← n
    while (BatasAtas ≤ BatasBawah) and (T[Tengah]≠x) do
        Tengah ← (BatasAtas + BatasBawah) div 2
        if (T[Tengah] > x) then
            BatasBawah ← Tengah-1
        else
            if (T[Tengah] < x) then
                BatasAtas ← Tengah+1
            endif
        endif
    endwhile

    if (T[Tengah]=x) then
        idx ← Tengah
    else
        idx ← 0
    endif
```

LATIHAN SOAL

1. Prosedur sequential versi boolean dan prosedur binary search pada contoh pembahasan yang sudah dibahas, implementasikan kedalam algoritma dengan perosedur seperti contoh algoritma sequential search yang sudah dibahas.
2. Perhatikanlah pada semua prosedur search yang sudah dibahas, dipakai kamus umum sebagai berikut.

```
Kamus Umum
    constant Nmax : integer = 100
    type TabInt : array [1..Nmax+1] of integer
    { jika diperlukan sebuah tabel, maka akan dibuat deklarasi sebagai berikut}
    T : TabInt {tabel integer}
    N : integer {indeks efektif, 1 ≤ N ≤ Nmax+1}
```

Jika prosedur diparametrisasi seperti pada spesifikasi yang diberikan, maka T dan N menjadi dua buah parameter. Padahal, nilai T dan N sebenarnya erat kaitannya satu sama lain. Deklarasi TabInt akan lebih baik jika dibungkus menjadi sebuah type komposisi sebagai berikut :

Kamus Umum

```
constant Nmax : integer = 100
type TabInt : < TI: array [1..Nmax+1] of integer
                N : integer {indeks efektif, }
                {maksimum tabel yang terdefinisi, 1 ≤ N ≤ Nmax+1}
{ jika diperlukan sebuah tabel, maka akan dibuat deklarasi sebagai berikut}
T : TabInt {tabel integer}
```

Sebagai latihan, tuliskan ulang semua prosedur yang pernah didefinisikan dengan deklarasi type komposisi ini.