**SEARCHING**

* Searching adalah pencarian data dengan cara menelusuri data-data tersebut.
* Tempat pencarian data dapat berupa array dalam memori, bisa juga pada file pada external storage.

**Sequential Search**

* Adalah teknik pencarian data dalam array ( 1 dimensi ) yang akan menelusuri semua elemen-elemen array dari awal sampai akhir, dimana data-data **tidak perlu** diurutkan terlebih dahulu.
* Kemungkinan terbaik (best case) adalah jika data yang dicari terletak di indeks array terdepan (elemen array pertama) sehingga waktu yang dibutuhkan untuk pencarian data sangat sebentar (minimal).
* Kemungkinan terburuk (worst case) adalah jika data yang dicari terletak di indeks array terakhir (elemen array terakhir) sehingga waktu yang dibutuhkan untuk pencarian data sangat lama (maksimal).
* Misalnya terdapat array satu dimensi sebagai berikut:

0 1 2 3 4 5 6 7 indeks

**8 10 6 2 10 7 1 100** value

* Kemudian program akan meminta data yang akan dicari, misalnya 6.
* Jika ada maka akan ditampilkan tulisan “ADA”, sedangkan jika tidak ada maka akan ditampilkan tulisan “TIDAK ADA”.

**Keterangan**

* dilakukan perulangan untuk mengakses semua elemen array data satu persatu berdasarkan indeksnya.
* menggunakan variabel “ket” yang berguna untuk menadai ada atau tidaknya data yang dicari dalam array data. Hanya bernilai 0 atau 1.
* “ket” pertama diinisialiasasi dengan nilai 0.
* Jika ditemukan, maka “ket” akan diset menjadi 1, jika tidak ada maka “ket” akan tetap bernilai 0.
* Semua elemen array data akan dibandingkan satu persatu dengan data yang dicari dan diinputkan oleh user.

**Binary Search**

* teknik pencarian data dengan cara membagi data menjadi dua bagian setiap kali terjadi proses pengurutan.
* **SYARAT:** “Data harus diurutkan terlebih dahulu berdasarkan suatu urutan tertentu yang dijadikan kunci pencarian.”
* Prinsip pencarian biner adalah:
  + Data diambil dari posisi 1 sampai posisi akhir N
  + Kemudian cari posisi data tengah dengan rumus (posisi awal + posisi akhir) / 2
  + Kemudian **data yang dicari dibandingkan dengan data yang di tengah**, apakah **sama** atau **lebih kecil**, atau **lebih besar**?
  + Jika lebih besar, maka proses pencarian dicari dengan posisi awal adalah posisi tengah + 1
  + Jika lebih kecil, maka proses pencarian dicari dengan posisi akhir adalah posisi tengah – 1
  + Jika data sama, berarti ketemu.

**Contoh Data:**

* Misalnya data yang dicari **17**

0 1 2 3 4 5 6 7 8

**1 8 10 13 15 17 20 30 40**

**A B C**

Karena 17 > 15 (data tengah), maka: awal = tengah + 1

0 1 2 3 4 5 6 7 8

**1 8 10 13 15 17 20 30 40**

**A B C**

Karena 17 < 20 (data tengah), maka: akhir = tengah – 1

0 1 2 3 4 5 6 7 8

**1 8 10 13 15 17 20 30 40**

**A=B=C**

Karena 17 = 17 (data tengah), maka KETEMU!

**Interpolation Search**

* dilakukan pada data yang sudah terurut berdasarkan kunci tertentu
* dilakukan dengan perkiraan letak data.
* Contoh:
  + jika kita hendak mencari suatu nama di dalam buku telepon,
  + misal yang berawalan dengan huruf W, maka kita tidak akan mencarinya dari awal buku, tapi kita langsung membukanya pada 2/3 atau ¾ dari tebal buku.
* Jadi kita mencari data secara relatif terhadap jumlah data.
* Rumus posisi relatif kunci pencarian dihitung dengan rumus:



Kasus: misalkan terdapat data sbb:

|  |  |  |
| --- | --- | --- |
| **Kode** | **Judul** | **Pengarang** |
| 025 | The C++ Programming | James Wood |
| 034 | Mastering Delphi 6 | Marcopolo |
| 041 | Professional C# | Simon Webe |
| 056 | Pure JavaScript v2 | Michael Bolton |
| 063 | Advanced JSP & Servlet | David Dunn |
| 072 | Calculus Make it Easy | Gunner Christian |
| 080 | Matrix | James |
| 088 | Visual Basic 2005 Express | Antonie |
| 096 | Artificial Life : Volume 1 | Gloria Virginia |

**Kasus #1:**

* Kunci Pencarian? **088**
* Low? **0**
* High? **8**
* Posisi = **int((088 - 025) / (096 - 025) \* (8 - 0) + 0) = [7]**
* Kode[7] = kunci pencarian, data ditemukan: **Visual Basic 2005 Express**

**Kasus #2:**

* Kunci Pencarian? **060**
* Low? **0**
* High? **8**
* Posisi = **int((060 – 025) / (096 – 025) \* (8 – 0) + 0) = [3]**
* Kunci[3] < kunci pencarian, maka teruskan
* Low = **3 + 1 = 4**
* High = **8**
* Posisi = **int((060 – 025) / (096 – 025) \* (8 – 4) + 4) = [5]**
* Ternyata Kunci[5] adalah **063** yang lebih besar daripada **060**.
* Berarti tidak ada kode **060.**

**Keterangan Proses Interpolasi Search**

* Low adalah indeks terdepan/terendah suatu array.
* High adalah indeks teratas/tertinggi suatu array.
* Kunci [i] 🡺 menandakan indeks ke-i
* jika data pada kunci[i] atau indeks[i], tidak ditemukan; maka:
  + nilai Low adalah nilai i dari kunci[i] (dimana data tidak ditemukan) DITAMBAH 1(satu) 🡺 Low = i+1
  + High tetap indeks teratas suatu array
  + Kemudian diteruskan proses perhitungan posisi relatif