[Searching](http://nbbajry.blog.com/2011/03/31/searching/)

Searching adalah pencarian data dengan cara menelusuri data-data tersebut. Pada suatu data seringkali dibutuhkan pembacaan kembali informasi (retrieval information) dengan cara searching. Tempat pencarian data dapat berupa array dalam memori, bisa juga pada file pada external storage.

Searching adalah satu keterampilan penting yang harus dikuasai oleh siapapun yang senantiasa bersinggungan dengan dunia IT. Dengan ketrampilan yang satu ini, maka pada prinsipnya apapun yang kita inginkan dari  dunia maya bisa kita dapatkan. Hanya saja ketrampilan ini tidak banyak diperhatikan oleh setiap orang. Khususnya lagi oleh para mahasiswa.

Pencarian ( searching ) dibagi menjadi 2 yaitu :

1. Pencarian Berurutan (sequential searching)
2. Pencarian Biner (Binary Search)

**Pencarian (Searching)**

Dalam kehidupan sehari-hari sebenarnya kita sering melakukan pencarian data. Sebagai contoh, jika kita menggunakan kamus untuk mencari kata-kata dalam Bahasa Inggris yang belum diketahui terjemahannya dalam Bahasa Indonesia. Contoh lain dalam saat kita menggunakan telepon untuk mencari nomor telepon teman atau kenalan dan masih banyak contoh yang lain.

Pencarian data juga sering disebut table look-up atau storage and retrieval information adalah suatu proses untuk mengumpulkan sejumlah informasi di dalam pengingat computer dan kemudian mencari kembali informasi yang diperlukan secepat mungkin.

Algoritma pencarian ( search algorithm ) adalah algoritma yang menerima sebuah argument kunci dan dengan langkah-langkah tertentu akan mencari rekaman dengan kunci tersebut. Setelah proses pencarian dilaksanakan, akan diperoleh salah satu dari dua kemungkinan, yaitu data yang dicari ditemukan ( successful) atau tidak ditemukan ( unsuccessfull).

Metode pencarian data dapat dilakukan dengan dua cara yaitu **pencarian internal** ( internal searching) dan **pencarian eksternal** (external searching). Pada pencarian internal, semua pada rekaman yang diketahui berada dalam pengingat computer sedangkan pada pencarian eksternal, tidak semua rekaman yang diketahui berada dalam pengingat computer, tetapi ada sejumlah rekaman yang tersimpan dalam penyimpanan luar misalnya pita atau cakram magnetis.

Selain itu data pencarian juga dapat dikelompokkan menjadi pencarian statis (static searching) dan pencarian dinamis (dynamic searching). Pada pencarian statis, banyaknya rekaman yang diketahui dianggap tetap, pada pencarian dinamis, banyaknya rekaman yang diketahui bisa berubah-ubah yang disebabkan oleh penambahan atau penghapusan suatu rekaman.

Ada dua macam teknik pencarian yaitu pencarian sekuensial dan pencarian biner. Perbedaan dari dua teknik ini terletak pada keadaan data. Pencarian sekuensial digunakan apabila data dalam keadaan acak atau tidak terurut. Sebaliknya, pencarian biner digunakan pada data yang sudah dalam keadaan urut.

**Pencarian Berurutan (sequential searching)**

Pencarian berurutan juga sering disebut pencarian linear merupakan metode pencarian yang paling sederhana. Pencarian berurutan menggunakan prinsip sebagai berikut : data yang ada dibandingkan satu per satu secara berurutan dengan yang dicari sampai data tersebut ditemukan atau tidak ditemukan.

Pada dasranya, pencarian ini hanya melakukan pengulangan dari 1smapai dengan sejumlah data. Pada setiap pengulangan, dibandingkan dat ke-idengan yang dicari. Apabila sama, data telah ditemukan. Sebaliknya apabila sampai akhir pengulangan tidak ada data yang sama, berarti data tidak ada. Pada kasus yang paling buruk, untuk N elemen data harus dilakukan pencarian sebanyak N kali pula.

Algoritma pencarian berurutan dapat dituliskan sebagai berikut :

1          i ← 0

2          ketemu ← false

3          Selama (tidak ketemu) dan (i <= N) kerjakan baris 4

4          Jika (Data[i] = x) maka ketemu ← true, jika tidak i ← i+1

5          jika (ketemu) maka i adalah indeks dari data yang dicari, jika data tidak ditemukan

Dibawah ini merupakan fungsi untuk mencari data yang menggunakan pencarian sekuensial.

|  |
| --- |
| int SequentialSearch (int x)  {  int i =0;  bool ketemu = false;  while ((! ketemu ) && (i < Max)){  if (data[i] == x)  ketemu = true;  else  i++;  }  if (ketemu)  return i;  else  return -1;  } |

**Deklarasi Fungsi Untuk Mencari Data dengan Metode Sekuensial**

Fungsi diatas akan mengembalikan indeks dari data yang dicari. Apabila data yang ditemukan maka fungsi diatas akan mengembalikan nilai -1.

**Pencarian Biner (Binary Search)**

Salah satu syarat agar pencarian biner dapat dilakukan adalah data sudah dalam keadaan urut. Dengan kata lain, apabila data belum dalam keadaan urut, pencarian biner tidak dapat dilakukan. Dalam keadaan sehari-hari, sebenarnya kita juga sering menggunakan pencarian biner. Misalnya saat ingin mencari suatu kata dalam kamus.

Prinsip dalam pencarian biner dapat dijelaskan sebagai berikut : mula-mula diambil posisi awal 0 dan posisi akhir = N-1, kemudian dicari posisi data tengah dengan rumus (posisi awal + posisi akhir) / 2. Kemudian data yang dicari dibandingkan dengan dat tengah. Jika lebih kecil, proses dilakukan kembali tetapi posisi akhir dianggap sama dengan posisi tengah -1. Jika lebih besar, proses dilakukan kembali tetapi posisi awal dianggap sama dengan posisi tengah + 1. Demikian seterusnya sampai data tengah sama dengan yang dicari.

Untuk lebih jelasnya perhatikan contoh berikut. Misalnya ingin mencari data 17 pada sekumpulan data berikut :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 9 | 11 | 12 | 15 | 17 | 20 | 23 | 31 | 35 |

Awal                                              tengah                                                            akhir

Mula-mula dicari data tengah, dengan rumus ( 0 + 9 ) / 2 = 4. Berarti data tengah adalah data ke-4, yaitu 15. Data yang dicari, yaitu 17, dibandingkan dengan data tengah ini. Karena 17 > 15, berarti proses dilanjutkan tetapi kali ini posisi awal dianggap sama dengan posisi tengah + 1 atau 5.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 9 | 11 | 12 | 15 | 17 | 20 | 23 | 31 | 35 |

Awal                    tengah                 akhir

Data tengah yang baru didapat dengan rumus (5+9) / 2 = 7. Berarti data tengah yang baru adalah data ke-7, yaitu 23. Data yang dicari yaitu 17 dibandingkan dengan data tengah ini. Karena 17 < 23, berarti proses dilanjutkan tetapi kali ini posisi akhir dianggap sama dengan posisi tengah -1 atau 6.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 9 | 11 | 12 | 15 | 17 | 20 | 23 | 31 | 35 |

Awal=tengah  akhir

Data tengah yang baru didapat dengan rumus ( 5+6 ) / 2 =5. Berarti data tengah yang baru adalah data ke-5, yaitu 17. Data yang dicari dibandingkan dengan data tengah ini dan ternyata yang sama. Jadi data ditemukan pada indeks ke-5.

Pencarian biner ini akan berakhir jika data ditemukan jika posisi awal lebih besar daripada posisi akhir. Jika posisi sudah lebih besar daripada posisi akhir berarti data tidak ditemukan.

Untuk lebih jelasnya perhatikan contoh pencarian data 16 pada data diatas. Prosesnya hamper sama dengan pencarian data 17. Tetapi setelah posisi awal 5 dan posisi akhir 6, data tidak ditemukan dan 16 < 17, maka posisi akhir menjadi posisi tengah -1 atau = 4 sedangkan posisi awal = 5.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 9 | 11 | 12 | 15 | 17 | 20 | 23 | 31 | 35 |

Akhir      awal

Disini dapat dilihat bahwa posisi awal lebih besar daripada posisi akhir, yang artinya data tidak ditemukan.

Algoritma pencarian biner dapat dituliskan sebagai berikut :

1          L ← 0

2          R ← N – 1

3          ketemu ← false

4          selama (L <= R) dan (tidak ketemu) kerjakan baris 5 sampai dengan 8

5          m ← (L + R) / 2

6          jika (Data[m] = x) maka ketemu ← true

7          jika (x < Data[m]) maka R ← M – 1

8          jika (ketemu) maka m adalah indeks dari data yang dicari, jika data tidak ditemukan.

Dibawah ini merupakan fungsi untuk mencari data menggunakan pencarian biner.

|  |
| --- |
| int BinarySearch ( int x )  {  int L = 0, R = Max-1, m;  bool ketemu = false;  while ( ( L <= R ) && ( ! ketemu ) )  {  m = ( L + R ) / 2;  if ( Data [m] == x )  ketemu = true;  else if ( x < data [m] )  R = m – 1;  else  L = m + 1;  }  if ( ketemu )  return m;  else  return -1;  } |

**Deklarasi Fungsi Pencarian Data dengan Metode Biner**

Fungsi diatas akan mengembalikan indeks dari data yang dicari. Apabila data tidak ditemukan maka fungsi diatas akan mengembalikan nilai -1.

Jumlah pembandingan minimum pada pencarian biner adalah 1 kali, yaitu apabila data yang dicari tepat berada di tengah-tengah. Jumlah pembandingan maksimum yang dilakukan dengan pencarian biner dapat dicari menggunakan rumus logaritma, yaitu :

*C*= 2log(N)

**kesimpulan**

1. Algoritma pencarian berurutan digunakan untuk mencari data pada sekumpulan data       atau rekaman yang masih acak.
2. Algoritma pencarian biner digunakan untuk mencari data pada sekumpulan data atau    rekaman yang sudah dalam keadaan terurut.

program searching;  
uses crt;

const  
nmax = 50;

type  
rdata = record  
nip : integer;  
nama : string[30];  
bagian : string[20];  
gaji : longint;  
end;  
adata = array [1..nmax] of rdata;

var  
data : adata;  
plh, ch : char;  
n : integer;

{ini buat ngurutkan data terlebih dulu…}  
procedure insertsort(var a :adata; n : integer);  
var  
pass, i : integer;  
temp, temp4 : integer;  
temp1, temp2 : string;  
begin  
for pass := 2 to n do  
begin

temp := a[pass].nip;  
temp1 := a[pass].nama;  
temp2 := a[pass].bagian;  
temp4 := a[pass].gaji;  
i := pass-1;

while ((temp1)) do

begin  
a[i+1].nip := a[i].nip;  
a[i+1].nama := a[i].nama;  
a[i+1].bagian := a[i].bagian;  
a[i+1].gaji := a[i].gaji;  
i := i-1;  
end;

if (temp < a[i].nip) then  
begin  
a[i+1].nip := a[i].nip;  
a[i+1].nama := a[i].nama;  
a[i+1].bagian := a[i].bagian;  
a[i+1].gaji := a[i].gaji;  
i := i-1;  
end;

a[i+1].nip := temp;  
a[i+1].nama := temp1;  
a[i+1].bagian := temp2;  
a[i+1].gaji := temp4;

end;  
end;

{prosedur untuk meminta input data}  
procedure inputData(var A:adata; var ndata : integer);  
var  
i : integer;  
begin  
clrscr;  
writeln(‘— Input Data —’);  
repeat  
writeln;  
write(‘Berapa banyak data? ‘);  
readln(ndata);  
if ndata>0 then  
begin  
writeln;  
for i:=1 to ndata do  
begin  
writeln(‘Pegawai Ke-’,i);  
write(‘NIP : ‘); readln(A[i].nip);  
write(‘Nama : ‘); readln(A[i].nama);  
write(‘Bagian : ‘); readln(A[i].bagian);  
write(‘Gaji : ‘); readln(A[i].gaji);  
writeln;  
end;  
end  
else  
writeln(‘Input salah !!!’)  
until(n>0);  
end;

function menu : char;  
var  
c : char;  
begin  
clrscr;  
writeln(‘— Menu Searching —’);  
writeln;  
writeln(’1. Sequential Searching’);  
writeln(’2. Binary Searching’);  
writeln(’3. Keluar Program’);  
repeat  
writeln;  
write(‘Pilih >> ‘);  
c := readkey;  
writeln(c);  
if not (c in ['1','2','3']) then  
writeln(‘Input Salah !!!’);  
until(c in ['1','2','3']);  
menu := c;  
end;

procedure sequential(A : adata; ndata : integer);  
var  
i : integer;  
nonip :integer;  
begin  
clrscr;  
writeln(‘— Sequential Searching —’);  
writeln;  
write(‘Masukkan NIP pegawai yang ingin dicari : ‘);  
readln(nonip);  
insertsort(a,n);  
writeln;

{rutin kode utama}  
i:=1;  
while ((A[i].nip<=ndata)) do  
i := i+1;  
if (A[i].nip=nonip) then  
begin  
writeln(‘Data ditemukan !!!’);  
writeln(‘NIP : ‘,A[i].nip);  
writeln(‘Nama : ‘,A[i].nama);  
writeln(‘Bagian : ‘,A[i].bagian);  
writeln(‘Gaji : ‘,A[i].gaji);  
end  
else  
writeln(‘Data tidak ditemukan !!!’);  
end;

procedure binary(A : adata; ndata : integer);  
var  
bawah, tengah, atas : integer;  
nonip : integer;  
found : boolean;  
begin  
clrscr;  
writeln(‘— Binary Searching —’);  
writeln;  
write(‘Masukkan NIP pegawai yang ingin dicari : ‘);  
readln(nonip);  
insertsort(a,n);  
writeln;

{rutin kode utama}  
bawah := 1;  
atas := ndata;  
found := false;  
while ((not found) and (atas>=bawah)) do  
begin  
tengah := (bawah+atas) div 2;  
if (A[tengah].nip=nonip) then  
found := true  
else  
if (A[tengah].nip>nonip) then  
atas := tengah-1  
else  
bawah := tengah+1  
end;

if (found) then  
begin  
writeln(‘Data ditemukan !!!’);  
writeln(‘NIP : ‘,A[tengah].nip);  
writeln(‘Nama : ‘,A[tengah].nama);  
writeln(‘Bagian : ‘,A[tengah].bagian);  
writeln(‘Gaji : ‘,A[tengah].gaji);  
end  
else  
writeln(‘Data tidak ditemukan !!!’);  
end;

{program utama}  
begin  
inputData(data,n);  
ch := ’2′;  
repeat  
if (ch=’2′) then  
plh := menu;

case plh of  
’1′ : sequential(data,n);  
’2′ : binary(data,n);  
’3′ : exit;  
end;  
writeln;  
repeat  
writeln;  
writeln(‘Tekan >> 1: Cari lagi!, 2: Kembali ke Menu, 3: Keluar Program’);  
write(‘ >> ‘);  
ch := readkey;  
if not (ch in ['1','2','3']) then  
writeln(‘Input Salah !!!’);  
until(ch in['1','2','3']);  
until(ch=’3′);  
end.