Algoritma dan Pemrograman Lanjut

Pertemuan Ke-7
Pencarian *(Searching)* 2

Disusun Oleh :
Wilis Kaswidjanti, S.Si.,M.Kom.

Jurusan Teknik Informatika
Fakultas Teknologi Industri
Universitas Pembangunan Nasional "Veteran"
Yogyakarta

Algoritma dan Pemrograman Lanjut

Judul Materi : Pencarian (*Searching)* 2

Deskripsi Materi : Materi ini membahas metode searching binary search menggunakan tipe data array untuk data urut

Tujuan Instruksional Khusus :

1. Memahami dan membandingkan metode searching menggunakan tipe data array
2. Mendeskripsikan berbagai metode searching
3. Mengilustrasikan berbagai metode searching menggunakan tipe data array

Referensi :

- Buku Teks

  Munir, Rinaldi (2005), *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 2, Edisi Ketiga, Penerbit Informatika Bandung, Bab 2, hal 35-76.

  Charibaldi, N. (2004), *Modul Kuliah Algoritma Pemrograman II*, Edisi Kedua, Yogyakarta

- Buku Acuan/Referensi

  Brassard, Gilles (1999), *Fundamentals of algorithma*, PrinteceHall.

  Jarne, Stroustrup B. (1997), *C++ Programming language*, AT &T.

  Kristanto, Andri (2003), *Algoritma pemrograman C++*, Graha Ilmu.

  Schildt,Herbert (2000), *The Complete Reference C++*, McGraw-Hill.

  Sedgewick, R. (2000), *Algoritma Third edition In C part 5*, Addison Wesley.

PENCARIAN (*SEARCHING*) 2

PENDAHULUAN

Pencarian dengan menggunakan tipe data Array dapat digunakan metode Sequential Search dan Binary search. Perbedaan dari kedua teknik ini terletak pada keadaan data.

ISI

Pencarian Biner (Binary Search)

Salah satu syarat pencarian biner (*Binary Search*) dapat dilakukan adalah data sudah dalam keadaan terurut. Dengan kata lain, apabila data belum dalam keadaan urut, pencarian biner tidak dapat dilakukan.

Dalam kehidupan sehari-hari, sebenarnya kita sering menggunakan pencarian biner, misalnya pada saat ingin mencari suatu kata dalam kamus.

Langkah dari pencarian biner adalah sebagai berikut:

1. mula-mula diambil posisi awal=1 dan posisi akhir=n
2. kemudian kita cari posisi data tengah dengan rumus posisi tengah = (posisi awal + posisi akhir) div 2
3. kemudian data yang dicari dibandingkan dengan data tengah
    a. jika sama, data ditemukan, proses selesai.
    b. jika lebih kecil, proses dilakukan kembali tetapi posisi akhir dianggap sama dengan posisi tengah - 1.
    c. jika lebih besar, proses dilakukan kembali tetapi posisi awal dianggap sama dengan posisi tengah + 1.
4. ulangi langkah 2 sampai data ditemukan, atau tidak ditemukan.
5. Pencarian biner ini akan berakhir jika data ditemukan atau posisi awal lebih besar dari pada posisi akhir. Jika posisi awal sudah lebih besar daripada posisi akhir berarti data tidak ditemukan.

Contoh :

Misalkan kita ingin mencari angka 14 pada sekumpulan data urut berikut :

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
3	7	10	12	13	14	20	24	29
awal				tengah				akhir

Jawab:

1. mula-mula dicari data tengah, dengan rumus tengah = (awal+akhir) div 2 = (1+9) div 2 = 5, berarti data tengah adalah data ke-5, dengan nilai 13
2. data yang kita cari adalah 14, bandingkan nilai 14 dengan data tengah.
3. karena 14 > 13, berarti proses dilanjutkan tetapi posisi awal dianggap sama dengan posisi tengah+1 atau 6

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
3	7	10	12	13	14	20	24	29
					awal	tengah		akhir

4. data tengah yang baru didapat dari rumus (6+9) div 2 = 7, berarti data tengah yang baru adalah data ke-7, yaitu 20.
5. data yang kita cari adalah 14, bandingkan nilai 14 dengan nilai tengah.
6. karena 14 < 20, berarti proses dilanjutkan, tetapi posisi akhir dianggap sama dengan posisi tengah-1 atau 6.

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
3	7	10	12	13	14	20	24	29
					awal akhir tengah			

7. data tengah yang baru didapat dari rumus (6+6) div 2 = 6, berarti data tengah yang baru adalah data ke-6, yaitu 14
8. data yang kita cari adalah 14, bandingkan dengan data tengah, ternyata sama.
9. jadi data yang ditemukan pada indeks ke-6

Bagaimana jika data yang dicari tidak ditemukan, misalnya data yang dicari 16? Pencarian biner ini akan berakhir jika data ditemukan atau posisi awal lebih besar dari pada posisi akhir. Jika posisi awal sudah lebih besar daripada posisi akhir berarti data tidak ditemukan.

Secara umum algoritma pencarian biner, adalah sebagai berikut :
(Data diurutkan lebih dahulu, data disimpan di data array, x adalah nilai yang dicari)

1. awal ← 1
2. akhir ← N
3. ketemu ← false
4. selama (awal<=akhir) dan (not ketemu) kerjakan baris 5 sampai 8.
5. tengah ← (awal+akhir) div 2
6. jika (data [tengah] = x) maka ketemu ← true
7. jika (x < data [tengah]) maka akhir ← tengah-l
8. jika (x > data [tengah] maka awal ← tengah+1.
9. if (ketemu) maka tengah adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

Contoh 1 pemakaian pencarian binary :

```
/* BinSearch_SudahUrut
diasumsikan Array X sudah ada dan berisi data yang sudah terurut,
nilai yang dicari adalah y dan hanya ada satu */

#include <iostream.h>
typedef enum boolean {false=0,true=1};
main()
{
 int X[10]={10,20,30,40,50,60,70,80,90,100};
 int i,y,j,k;
 boolean found;
 cout << "nilai yang dicari = ";
 cin >> y;
 found = false;
 i=0;
 j=10;
 while ((!found) & (i <= j))
 {
       k=(i+j)/2;
       if (y == X[k])
            found=true;
```

```
       else
            if (y<X[k])
                 j=k-1;    //i tetap
            else
                 i=k+1; //j tetap
 }
 if (found)
     cout<< y<<"ditemukan dalam Array pada index ke-" << k;
 else
     cout << "tidak ada " << y << " dalam Array";
}
```

Contoh 2 pemakaian pencarian binary :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int binary_search (int array[], int value, int size)
{
  int found= 0;
  int high= size, low= 0, mid;
  Mid= (high+low)/2;
  printf("\n\nLooking for %d\n", value);
  while ((! Found) && (high >= low))
    {
   printf("Low %d Mid %d High %d\n", low, mid, high);
   if (value == array[mid])
    Found= 1;
   else if (value < array[mid])
    High= mid - 1;
   else
    Low= mid + 1;
   Mid= (high + low) / 2;
  }
 return (( found) ? mid: -1);
}

main()
{
 int array[100], I;
 clrscr();
 for (i=0; i<100; i++)
    array[i]= i;
 printf("Result of search %d\n",binary_search(array, 33, 100));
 printf("Result of search %d\n",binary_search(array, 75, 100));
 printf("Result of search %d\n",binary_search(array, 1001, 100));
 getche();
}
```

Program ini dimaksudkan untuk mengamati jumlah operasi searching yang harus dilakukan untuk menemukan masing-masing nilai.

PENUTUP

Metode Binary Search dapat dipakai hanya untuk data yang urut atau telah diurutkan.

SOAL-SOAL

1. Ilustrasikan pencarian nilai 25 pada kumpulan data dibawah. Gunakanlah metode Binary Search.

12	15	4	25	45	10	19	9	32	33

2. Ilustrasikan pencarian nilai 20 pada kumpulan data dibawah. Gunakanlah metode Binary Search.

12	15	4	25	45	10	19	9	32	33

3. Tulislah program untuk mengurutkan data bertipe rekaman menggunakan salah satu metode pengurutan. Tipe rekaman yang harus diurutkan mempunyai empat data, yaitu:

- nomorinduk, bertipe bilangan bulat
- nama, bertipe string
- alamat, bertipe string
- golongan, bertipe char (dapat bernilai ‘A’, …’Z’);

procedure pengurutan menerima satu parameter, yaitu bilangan bulat yang dapat bernilai 1, 2, atau 3. Apabila bernilai 1, maka data diurutkan menurut nomorinduk, apabila bernilai 2, maka data diurutkan menurut nama, dan apabila bernilai 3, maka data diurutkan menurut golongan.