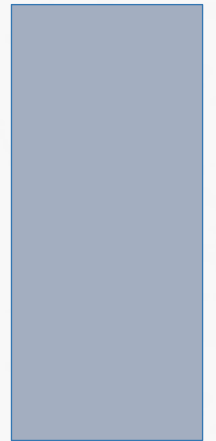


SEARCHING

ALGORITMA



SEARCHING

- Algoritma pencarian (searching algorithm) adalah algoritma yang menerima sebuah argumen 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 (unsuccessful)

JENIS SEARCHING

- Ada beberapa pencarian yang akan kita uraikan disini:
 - a) Pencarian Beruntun (Sekuensial Search)
 - b) Pencarian Bagi dua (Binary Search)

PENCARIAN BERUNTUN (SEKUENSIAL SEARCH) (1)

- Pencarian berurutan 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 dasarnya, pencarian ini hanya melakukan pengulangan dari 1 sampai dengan jumlah data.
- Pada setiap pengulangan, dibandingkan data ke- i dengan yang dicari.
- Apabila sama, berarti 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.

PENCARIAN BERUNTUN (SEKUENSIAL SEARCH) (2)

- Berikut ini adalah contoh pencarian sekuensial:

| Elemen | 13 | 16 | 14 | 21 | 76 | 15 |
|--------|----|----|----|----|----|----|
| Index | 0 | 1 | 2 | 3 | 4 | 5 |

Dari data diatas, larik L mempunyai 6 elemen. Pencarian akan dimulai dari indeks ke-0 yaitu posisi pertama. Misalkan elemen yang dicari: $X = 21$. Urutan elemen yang dibandingkan:

```
13 <> 21
16 <> 21
14 <> 21
21 = 21 (ditemukan idx = 4)
```

PENCARIAN BERUNTUN (SEKUENSIAL SEARCH) (4)

```
#include <iostream>
using namespace std;
int main() {
    int jml;
    int i ;
    int cari;
    int arr[50];
    int tanda=-1;
    cout<<"Masukkan banyaknya bilangan = " ;
    cin>>jml;
    for(i=0;i<jml;i++)
    {
        cout<<"Masukkan bilangan ke-"<<i+1<<" : ";
        cin>>arr[i];
    }
    cout<< "Isi dari array : "<<endl;
    for (i=0;i<jml;i++)
        cout<<" "<<arr[i];
    cout<<"\n\nMasukkan data yang dicari: ";
    cin>>cari;
    for (i=0;i<jml;i++)
    {
        if (cari ==arr[i])
        {
            tanda=i;break;
        }
    }
    if (tanda!=-1)
        cout<<"\n\n Data ditemukan" ;
    else
        cout<<"\n\n Data tidak ditemukan";
    return 0;
}
```

PENCARIAN BERUNTUN (SEKUENSIAL SEARCH) (5)

Kode 5.2.

```
#include <stdio.h>
#include <conio.h>
#include <iostream>
using namespace std;

main()
{
    int larik[10]={15,12,3,7,10,6,17,11,9,14}, i,n, x, posisi;

    cout<<"data yang ingin dicari ? ";cin>>x;
    i=0;
    posisi=0;

    while(i<9 && larik[i]!=x)
    {
        i++;

    }
    if (larik[i]!=x)
        cout<<"maaf data yang dicari tidak ada";
    else if (larik[i]==x)
    {
        posisi=i+1;

        cout<<"pada posisi ke "<<posisi;
    }
    getch();
}
```

Output kode 5.2.

```
C:\Windows\system32\cmd.exe - 1

D:\sd2017>g++ -o 1 search1.cpp

D:\sd2017>1
data yang ingin dicari ? 17
pada posisi ke 7
D:\sd2017>1
data yang ingin dicari ? 32
maaf data yang dicari tidak ada
```

PENCARIAN BINER (BINARY SEARCH)

(1)

- Pencarian biner adalah proses mencari data dengan membagi data atas dua bagian secara terus menerus sampai elemen yang dicari sudah ditemukan, atau indeks kiri lebih besar dari indeks kanan.
- Algoritma pencarian ini mempunyai syarat yaitu bahwa kumpulan data yang harus dilakukan pencarian harus sudah terurut terlebih dahulu.
- Karena data sudah terurut, algoritma dapat menentukan apakah nilai data yang dicari berada sebelum atau sesudah elemen larik yang sedang dibandingkan pada suatu saat.
- Dengan cara ini, algoritma dapat lebih menghemat waktu pencarian.

PENCARIAN BINER (BINARY SEARCH)

(2)

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

PENCARIAN BINER (BINARY SEARCH)

(3)

| | | | | | | | |
|------|----|----|----|----|----|----|------|
| 81 | 76 | 21 | 18 | 16 | 13 | 10 | 7 |
| 1a=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8=1b |

1. Misal elemen yang dicari adalah $X = 18$

langkah 1:

1a=1 dan 1b=8

elemen tengah $K = (1+8)/2 = 4$ (diarsir)

| | | | | | | | |
|----|----|----|----|----|----|----|---|
| 81 | 76 | 21 | 18 | 16 | 13 | 10 | 7 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

kiri kanan

Langkah 2:

$L[4] = 18$? Ya! (X ditemukan, pencarian dihentikan)

PENCARIAN BINER (BINARY SEARCH)

(4)

2. Misal elemen yang dicari adalah $X = 16$

langkah 1:

$ia=1$ dan $ib=8$

elemen tengah $K=(1+8)/2 = 4$ (diarsir)

| | | | | | | | |
|----|----|----|----|----|----|----|---|
| 81 | 76 | 21 | 18 | 16 | 13 | 10 | 7 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

kiri kanan

Langkah 2:

$L[4]=18$? Tidak !

harus diputuskan apakah pencarian akan dilakukan di bagian kiri atau kanan dengan pemeriksaan sbb:

$L[4] > 16$? Ya ! Lakukan pencarian pada larik bagian kanan dengan $ia=k+1 = 5$ dan $ib = 8$ (tetap)

PENCARIAN BINER (BINARY SEARCH)

(5)

| | | | |
|----|----|----|---|
| 16 | 13 | 10 | 7 |
| 5 | 6 | 7 | 8 |

Langkah 2.1:

ia = 5 dan ib = 8

elemen tengah $K = (5+8)/2 = 6$ (diarsir)

| | | | |
|----|----|----|---|
| 16 | 13 | 10 | 7 |
| 5 | 6 | 7 | 8 |

Kiri

kanan

Langkah 2.2:

$L[6] = 16$? Tidak !

harus diputuskan apakah pencarian akan dilakukan di bagian kiri atau kanan dengan pemeriksaan sbb:

$L[6] > 16$? Tidak ! Lakukan pencarian pada larik bagian kiri dengan
 $ia = 5$ dan $ib = k-1 = 5$

PENCARIAN BINER (BINARY SEARCH)

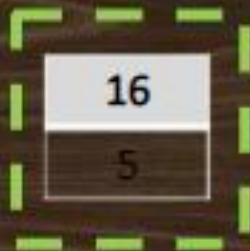
(6)



Langkah 2.1.1:

$ia = 5$ dan $ib = 5$

elemen tengah $K = (5+5)/2 = 5$ (diarsir)



Langkah 2.1.2:

$L[5] = 16$? Ya ! (X ditemukan, pencarian dihentikan)

PENCARIAN BINER (BINARY SEARCH)

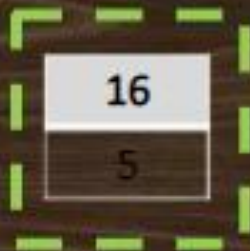
(7)



Langkah 2.1.1:

$ia = 5$ dan $ib = 5$

elemen tengah $K = (5+5)/2 = 5$ (diarsir)



Langkah 2.1.2:

$L[5] = 16$? Ya ! (X ditemukan, pencarian dihentikan)

PENCARIAN BINER (BINARY SEARCH)

(8)

```
#include<iostream>

using namespace std;

int main(){

    int n, angka[10],kiri , kanan,tengah ,temp, key;

    bool ketemu = false;

    cout<<"Masukkan jumlah data: ";
    cin>>n;

    for(int i = 0; i < n; i++)
    {
        cout<<"Angka ke-["<<i<<"]": ";
        cin>>angka[i];
    }

    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n-i-1; j++)
        {
            if(angka[j] > angka[j+1])
            {
                temp = angka[j];
                angka[j] = angka[j+1];
                angka[j+1] = temp;
            }
        }
    }
}
```


PENCARIAN BINER (BINARY SEARCH)

(9)

```
cout<<"Data yang telah diurutkan adalah : ";

for(int i = 0; i < n; i++)
{
    cout<<angka[i]<<" ";
}

cout<<"\n Masukkan angka yang dicari: ";

cin>>key;
```

```
kiri = 0;
kanan = n-1;

while(kiri<=kanan)
{
    tengah = (kiri + kanan)/2;

    if(key == angka[tengah])
    {
        ketemu = true;
        break;
    }
    else if(key < angka[tengah])
    {
        kanan = tengah - 1;
    }
    else
    {
        kiri = tengah + 1;
    }
}

if(ketemu==true)
    cout<<"Angka ditemukan!";
else
    cout<<"Angka tidak ditemukan";

return 0;
```

```
}
```