PERTEMUNAN 6
SEARCHING

A. TUJUAN PEMBELAJARAN

Setelah menyelesaikan pertemuan ini, mahasiswa mampu mempraktekkan: mengenai pengertian seraching, dapat melakukan searching untuk beberapa metode yang ada, dan dapat mengiplementasikan searching pada metode pemrograman c++.

B. URAIAN MATERI

1. Searching

Secara umum *search* diartikan mencari data dengan cara menelusuri tempat penyimpanan data tersebut. Tempat Penyimpanan data datap berupa arraydalam memori, bias juga berada daam satu file pada *eksternal storage*.

Pencarian itu penting dan paing sering dilakukan pada engoprasian computer, secara umum pencarian dilakukan dengan memasukan dengan sebuah kata kunci. Misalnya, jika seorang manajer bank mencari transaksi tertentu pada elanggan naka yang dilakukan adalah memasukan nid nasabah, atau jika anda seorang mahasiswa anda dapat mencari data diri anda dengan menggunakan NIM. Ada berbagai jenis pencarian, seperti pencarian linier, pencarian biner,dan penarian hash, dll.

Kami biasanaya melakukan peneliususran dengan file kunci. Kunci ini adalah pengenal untuk data pencatat . contoh utamanya adalah ID,Klien, NIK,NIM dll. Output dari algoritma pencarian biasanya adalah posisi record atau isi dari sebuah catatan.

Jadi pencarian adalah sebuah dasar dalam sebuah proses pemrograman atau tindakan untuk mendapatkan suatu data dalam kumpulan data berdasarkan satu kunci atau acuan data.

2. Sequential Search

Sequential Search (pencarian beruntun) adalah metode pencarian yang paling mudah. Pencarian beruntun adalah proses membandingkan setiap elemen array satu per satu secara beruntun yang dimulai dari elemen pertama hingga elemen yang dicari ditemukan atau hingga elemen terakhir dari array. Pencarian beruntun dapat dilakukan terhadap elemen array yang belum terurut atau terhadap elemen array yang terurut. Perbedaan dari keduanya terletak pada efisiensi operasi pembandingan yang dilakukan.

Dengan kata lain sequential search akan mencari data dengan cara membandingkannya satu-persatu dengan data yang ada. Prosesnya tentu saja akan singkat jika data yang diolah sedikit, dan akan lama jika data yang diolah banyak.

Proses kerja pencarian beruntun.

Misalkan kita memiliki suatu aray yang berisi 8 elemen seperti di atas, dimana variabel yang diguunakan adallah:

Nilai : Variabel aray skumpulan bilangan

Posisi : Variabel aray yang menampung ideks dari bilanganyang dicari

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1							

Krena tidak sama maka tidak ada perubahan dan dilnjutkan ke proses prulangan kempat dengan menaikkan harga i=4(i++).

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1							

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1							

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1							

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1	2						

Nilai	10	15	9	3	25	65	15	30
Indeks	0	1	2	3	4	5	6	7
Posisi	1	6						

Maka dari proses di atas dapat kita lihat keluaran dari program adalah “Data 15 ada ditemukan sebanyak 2 yang ada pada posisi 1 dan 6”.

3. BinarySearch

Pencarian biner adalah metode untuk menemukan elemen yang diperlukan dalam array yang dipesan dengan berulang kali membagi array menjadi dua dan mencari setengahnya.

Metode ini dilakukan dimulai dengan seluruh matriks. Kemudian dipotong menjadi dua. Jika nilai data yang diperlukan lebih besar dari elemen di tengah larik, bagian atas larik diperhitungkan. Jika tidak, itu dianggap bagian bawah. Ini dilakukan terus menerus sampai nilai data yang dibutuhkan diperoleh atau array yang tersisa kosong.

Pencarian biner, juga dikenal sebagai pencarian jarak menengah, pencarian log, atau cutoff biner, adalah algoritma pencarian yang menemukan posisi nilai target dalam array yang dipesan. Pencarian biner membandingkan nilai target dengan elemen sentral dari array. Jika tidak sama, separuh di mana target tidak dapat dihilangkan dan pencarian dilanjutkan pada separuh sisanya, sekali lagi membandingkan item di tengah dengan nilai target dan mengulanginya hingga nilai target ditemukan. Jika pencarian diakhiri dengan sisa setengah kosong, target tidak ada dalam array. Meskipun idenya sederhana, mengimplementasikan pencarian biner dengan benar memerlukan perhatian pada beberapa kehalusan tentang kondisi keluaran dan penghitungan titik tengah, terutama jika nilai dalam larik tidak semuanya bilangan bulat dalam rentang.

Pencarian biner adalah algoritma pencarian paling populer. Ini efisien dan juga salah satu teknik pemecahan masalah yang paling banyak digunakan.Jika semua nama di dunia ditulis bersama secara berurutan dan Anda ingin menemukan posisi nama tertentu, pencarian biner akan mencapai ini hingga 35 iterasi.

Pencarian biner hanya bekerja pada sekumpulan item yang dipesan. Untuk menggunakan pencarian biner pada sebuah koleksi, koleksi tersebut harus diurutkan terlebih dahulu.

Saat menggunakan pencarian biner untuk melakukan operasi pada set yang dipesan, jumlah iterasi selalu dapat dikurangi berdasarkan nilai yang dicari.

Mari perhatikan urutan berikut:

0	1	2	3	4	5	6	7	8	9

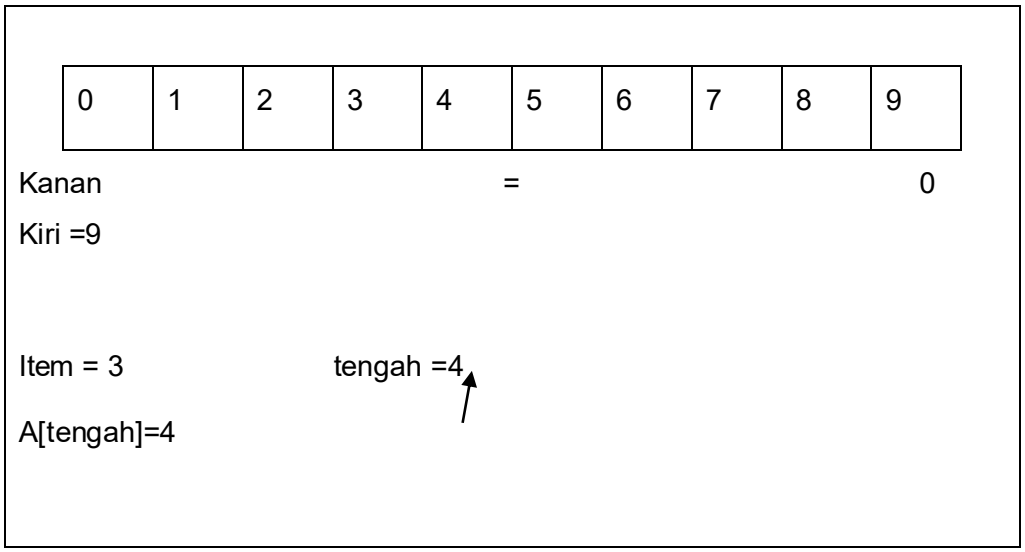
Menggunakan pencarian linier, posisi elemen 8 ditentukan pada iterasi kesembilan. Mari kita lihat bagaimana Anda bisa mengurangi jumlah iterasi menggunakan pencarian biner. Sebelum kita mulai mencari, kita perlu mengetahui awal dan akhir jangkauan. Sebut saja mereka rendah dan tinggi.

Rendah = 0

Tinggi = n-1

Sekarang bandingkan nilai pencarian Konstanta dengan elemen yang ada di median batas bawah dan atas. Jika nilai konstanta lebih tinggi, naikkan batas

bawah atau turunkan batas atas.



Mengacu pada gambar di atas, batas bawahnya adalah 0 dan batas atasnya adalah 9Median dari batas bawah dan atas adalah (batas bawah + batas atas) / 2 = 4. Di sini, di [4] = 4. Nilai 4> 2, ini adalah nilai yang Anda cari. Oleh karena itu, tidak perlu mencari elemen yang melebihi 4, karena elemen di luar jelas lebih besar dari 2.

Oleh karena itu, kita selalu dapat menurunkan batas atas matriks ke posisi elemen 4. Sekarang kita mengikuti prosedur yang sama pada matriks yang sama dengan nilai berikut:

Rendah = 0

Tinggi = 3

Ulangi prosedur ini secara berulang hingga Rendah> Tinggi. Jika kita mendapatkan key [tengah] = dalam satu iterasi, kita akan mengembalikan nilai mid. Ini adalah posisi kunci dalam matriks. Jika kunci tidak ada dalam larik, kami mengembalikan -1.

Di bawah ini adalah program yang mendemonstrasikan pencarian biner di C ++.

```
int binarySearch(int rendah,inttinggi,int kunci)
{
```

```
while(rendah<=tinggi){
    int tengah=(rendah+tinggi)/2;
if (a[tengah]<kunci){
      rendah=tengah+1;
    }
elseif(a[tengah]>kunci){
      tinggi=tengah-1;
    }
else{
return tengah;
    }
  }
return -1;
}
```

Untuk lebih jelasnya perhatikan program berikut :

```
// latihan binarisearch 1
#include<stdio.h>
#include<stdlib.h>
int binsearch(int array[],int val,intlo,int hi);
 int main()
{int len,i=0,val,key;
 int array[30];
printf("masukan nilai \n");
printf("\n tekan <0 untuk berhenti> ");
```

```
scanf("%d",&array[i]);
while(array[i]!=0)
{i++;
printf("\n tekan <0 untuk berhenti> ");
scanf("%d",&array[i]);
}
len=i; //lengthistaken as i and not as i+1 as 0 isalsostored in thearray
printf("\n **** Array **** \n");
for(i=0;i<len;i++)
printf("%d\t",array[i]);
printf("\n\nMasukan nilai yang akan di cari : ");
scanf("%d",&val);
key=binsearch(array,val,0,len-1);
//callthefunctionbinsearchwhichwillserachtheelement
if(key==-1)
printf("\n Angka tidak ada dalam array");
else
printf("\nAngka %d ditemukan pada posisi ke %d dalam array\n",val,key);
}
int binsearch(int array[],int val,intlo,int hi)
{ int mid,key;
if(lo>hi)
key=-1;
else
  { mid=(lo+hi)/2;
if(val==array[mid])
key=mid;
```

```
else
if(val<array[mid])
  {hi=mid-1;
key=binsearch(array,val,lo,hi);
  }
else
if(val>array[mid])
 { lo= mid+1;
key=binsearch(array,val,lo,hi);
 }
  }
return(key);
}
```

4. InterpolasiSearch

Algoritme pada Interpolasisearch digunakan untuk mencari nilai dalam baris elemen yang terdistribusi secara seragam dan teratur. Pencarian Interpolasi akan menuju ke lokasi yang berbeda untuk memulai, tergantung apakah nilai yang dicari lebih dekat ke akhir atau awal array, tidak seperti Pencarian Biner yang selalu mencari di tengah. Jadi, teknik ini mencoba menemukan lokasi nilai yang tepat, bukan di tengah, menggunakan rumus interpolasi. Waktu pencarian berkurang, karena mencoba menemukan perkiraan posisi terbaik setiap saat.

InterpolationSearch merupakan sebuah teknik pengembangan dari binary search. Untuk mencari Nilai Mid pada interpolation search di dapat dari :

$$Mid = \frac{kunci - data[min]}{data[max] - data[min]} x(max - min) + min$$

Contohnya saja pada nilai int array= {90, 30, 20, 40, 50,10}, data para int array harus diurutkan terlebih dahulu menggunakan teknik sorting. Sehingga array kita akan menjadi int arr[] = {10,20,30,40,50,90}. Apabila angka yang dicari adalah angka 30, berikut gambaran dari implementasi InterpolationSearch:

a. **Proses pertama**
   - (10,20,30,40,50,90)
     find=30
     low = 0
     high = N-1
     mid =((find-arr[low]) / (arr[high]-arr[low]))*(high-low) + low= 0
     (arr[mid] == 30)
     (10,20,30,40,50,90)
     (10==30) // FALSE
     low = mid + 1*Array di mulai dari index ke 0, maka index ke 3 berisi nilai 40.X

b. Proses kedua
   - (10,20,30,40,50,90)
     find=30
     mid =((find – arr[low]) / (arr[high]-arr[low])) *(high-low) + low = 1
     (arr[mid] == 30)
     (10,20,30,40,50,90)
     (20==30) // FALSE
     low = mid + 1

c. **Proses ketiga**
   - (10,20,30,40,50,90)

Pada cycle ke 3, looping ini sudah berhenti karena hasilnya sudah true dan data nya sudah ditemukan. Kita tinggal mengcheck apakah array bagian low atau high merupakan nilai yang kita cari atau bukan. Jika data ditemukan, maka program akan keluar dari looping. Jika kita ingin menampilkan index dari data yang dicari, kita tinggal menyimpan index dari array tersebut dan menampilkan nya.

```
Nama File : latihan interpolasiserach 1*/
#include <stdio.h>
#include<iomanip>
#include<conio.h>
using namespace std;
int main()
{
int arr[]={10,20,30,40,50,90};
int n = sizeof(arr)/sizeof(int);
int index=-1; //index array mulai dari 0, maka di set -1
int find=30;
int mid, low = 0, high = n-1;
while(arr[low] < find && arr[high] > find)
{
mid =((find-arr[low])/(arr[high]-arr[low]))*(high-low) + low;
if(arr[mid] < find)
{
low = mid + 1;
}
else if(arr[mid] > find){
high = mid-1;
}
else{
index=mid;
break;
```

```
}

}

if (arr[low] == find){

index = low;

}

else if(arr[high] == find){

index = high;

}
if(index==-1){

printf("Data yang anda cari tidak ditemukan\n");
}
else{
printf("Data yang anda cari berada pada index ke-%d\n",index);

}
return 0;

}
```

C. SOAL LATIHAN/TUGAS

Latihan	Petunjuk Pengerjaan Tugas
Latihan 6	1. Jelaskan pengertian,fungsi dan jenisnya dari searching ! 2. Buatlah program menggunakan sequentialserach! 3. Buatlah program menggunakan Binaryserach! 4. Buatlah program menggunakan interpolationserach! 5. Buatlah program searching mengecek apakah sebuah huruf ada dalam karakter yang telah diinput, kemudian hurufnya urutkan!

D. REFERENSI

Dordal, P. L. (2020). *An Introduction to Computer Network.* Chicago: Loyola University Chicago.

Forouzan, B. A. (2013). *Data Communications and Networking.* New York: McGraw-Hill.

Goralski, W. (2017). *The Illustrated Network.* Cambridge: Morgan Kaufmann.

Kurose, J. F., & Ross, K. W. (2017). *Computer Networking: A Top-down Approach.* Pearson.

Lowe, D. (2018). *Networking All-In-One.* Hoboken: John Wiley & Sons, Inc.

Peterson, L. L., & Davie, B. S. (2010). *Computer Networks.* Burlington: Kaufmann.

Sudiendro, H. (2013). *Teknik Dasar Telekomunikasi.* Jakarta: Kementrian Pendidikan & Kebudayaan.

Sukaridhoto, S. (2014). *Buku Jaringan Komputer I.* Surabaya: Politeknik Elektronika Negeri Surabaya (PENS).

Tanenbaum, A. S., & Wetherall, D. J. (2011). *Computer Networks.* Pearson Prentice Hall.

Pintarkom.com.(2019,25 November) Searching pada c++ pengertian dan contoh program. Diakses 18 November 2020, https://pintarkom.com/searching-pada-c-plus/