MAKALAH SEARCHING DALAM PEMROGRAMAN

BAHASA C



Oleh :

DIKI CANDRA

NIM 2022903430010

D4 TEKNOLOGI REKAYASA KOMPUTER JARINGAN

POLITEKNIK NEGERI LHOKSEMAWE

2022/2023

BAB I

PENDAHULUAN

A. LATAR BELAKANG

Seiring dengan berkembangnya Teknologi Informasi, yang membutuhkan kecepatan danketepatan dalam berbagai aspek kehidupan. Kita dituntut untuk bergerak cepat dalam segala hal,termasuk juga dalam bidang komunikasi informasi. Salah satunya dalam bidang statistic yangmeliputi pencarian data. Sehingga banyak programer yang membuat program yangmempermudah kerja dalam bidang statistika dan informasi, yakni dalam hal pencarian data. Dari program pencarian data terdapat algoritma-algoritma yang digunakan untuk mencari sebuah data,salah satunya adalah binary search.Binary search adalah algoritma pencarian untuk data yang terurut. Pencarian dilakukandengan cara menebak apakah data yang dicari berada ditengah-tengah data, kemudianmembandingkan data yang dicari dengan data yang ada ditengah. Bila data yang ditengah samadengan data yang dicari, berarti data ditemukan. Namun, bila data yang ditengah lebih besar daridata yang dicari, maka dapat dipastikan bahwa data yang dicari kemungkinan berada disebelahkiri dari data tengah dan data disebelah kanan data tengah dapat diabai.

*Upper bound*

dari bagiandata kiri yang baru adalah indeks dari data tengah itu sendiri. Sebaliknya, bila data yang ditengahlebih kecil dari data yang dicari, maka dapat dipastikan bahwa data yang dicari kemungkinan besar berada disebelah kanan dari data tengah*.*

*Lower bound*

dari data disebelah kanan dari datatengah adalah indeks dari data tengah itu sendiri ditambah 1. Demikian seterusnya.

B. RUMUSAN MASALAH

Berdasarkan uraian pada latar belakang masalah diatas, maka permasalahan yang diteliti dapat dirumuskan:

* Apa pengertian searching/pencarian?
* Apa saja jenis-jenis algoritma pencarian?
* Bagaimana algoritma dan contoh programnya?
* Kelemahan Searching
* Keunggulan Searching

C. TUJUAN

* Mahasiswa dapat melakukan perancangan aplikasi menggunakan struktur Searching (Pencarian)
* Mahasiswa mampu melakukan analisis pada algoritma Searchingyang dibuat
* Mahasiswa mampu mengimplementasikan algoritma Searching pada sebuahaplikasi secara tepat dan efisien
* Mahasiswamampu menjelaskan mengenai algoritma Searching.
* Mahasiswamampu membuat dan mendeklarasikan struktur algoritma Searching.
* Mahasiswamampu menerapkan dan mengimplementasikan algoritma Searching.

D. Manfaat

Manfaat yang diperoleh dari makalah tersebut adalah sebagai berikut:

1.Pembaca

A. Menjadikan pembaca makalah semata-mata agar pembaca memahami dasar-dasar pemrograman.

B. Untuk memotivasi pembaca untuk membuat makalah yang lebih baik dari Ini.

2.Pengarang

C. Untuk memberikan semangat pada mata kuliah yang sedang diambil saat ini.

D. Sebagai titik awal untuk melakukan praktikum agar praktikum selanjutnya dapat dilakukan dengan lebih baik lagi.

BAB II

PEMBAHASAN

A. PENGERTIAN SEARCHING

Pencarian (searching) merupakan proses fundamental dalam pengelolaan data. Proses pencarian adalah menemukan nilai (data) tertentu di dalam sekumpulan data yang bertipe sama (baik bertipe dasar atau bertipe bentukan). Search algoritma adalah algoritma yang menerima argument *a* dan mencoba untuk mencari record yang mana key-nya adalah Algoritma bisa mengembalikan nilai record, atau pointer ke record. Record sendiri adalah tipe data yang terdiri atas kumpulan variabel yang dapat berbeda tipenya. Setiap variabel disebut field. Sequensial Search (penelusuran sequensial) yaitu proses mengunjungi melalui suatu pohon dengan cara setiap simpul di kunjungi hanya satu kali yang disebut tree transversal / kunjungan pohon. Sedangkan Binary Search adalah penelusuran pohon biner dimana data yang dimasukkan atau yang sudah ada diurutkan terlebih dahulu.

Data dapat di simpan secara temporer dalam memori utama atau di simpan secara permanen di dalam memori sekunder (tape atau disk). Di dalam memori utama, struktur penyimpanan data yang umum adalah berupa larik atau tabel (array), sedangkan di dalam memori sekunder berupa arsip (file). Aktivitas yang berkaitan dengan pengolahan data ini sering di dahului dengan proses pencarian. Sebagai contoh, untuk mengubah (update) data tertentu, langkah pertama yang harus dilakukan adalah mencari keberadaaan data tersebut di dalam kumpulannya. Aktivitas yang awal sama juga dilakukan pada proses penambahan (insert) data yang baru. Proses penambahan data dimulai dengan mencari apakah data yang ditambahkan sudah terdapat di dalam kumpulan. Jika sudah dan mengasumsikan tidak boleh ada duplikasi data maka data tersebut tidak perlu ditambahkan, tetapi jika belum ada, maka tambahkan.

Algoritma pencarian yang akan dibicarakan dimulai dengan algoritma pencarian yang paling sederhana yaitu pencarian beruntun atau Sequential Search sampai pada algoritma pencarian yang lebih maju yaitu pencarian bagi dua (Binary Search).

B. JENIS-JENIS SEARCHING

1. Squential Search

Disebut juga sebagai metode pencarian urut adalah metode pencarian yang paling mudah. Adalah suatu 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). Sedangkan 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).

Sequential search memiliki proses sebagai berikut:

»     Tentukan banyaknya data yang akan di olah, misal banyak data adalah N.

»     Tentukan data apa yang akan dicari, misal data yang akan dicari adalah C.

»     Deklarasikan sebuah counter untuk menghitung banyak data yang ditemukan, missal counternya adalah K.

»     Inisialisasikan K =0

»     Lakukanlah perulangan sebanyak N kali

»     Dalam tiap proses perulangan tersebut periksalah apakah data yang sedang diolah sama dengan data yang dicari.

»     Jika ternyata sama K=K+1

»     Jika tidak, lanjutkan proses perulangan .

»     Setelah proses perulangan berhenti, periksalah nilai K.

»     Jika nilai K lebih dari 0, artinya data yang dicari ada dalam data /array dan tampilkan   nilai K ke layer sebagai jumlah data yang ditemukan.

»     Jika nilai K=0, artinya data yang dicari tidak ditemukan dalam data / array dan tampilkan ke layar bahwa data tidak ditemukan.

» Proses selesai. Dapat disimpulkan bahwa 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. Disarankan proses ini digunakan pada jumlah data yang sedikit saja.

**Kelebihannya**:

* Relatif lebih cepat dan efisien untuk data yang terbatas
* Algoritma sederhana

**Kekuranganya :**

* Kurang cepat untuk data dalam jumlah besar
* Beban komputasi cenderung lebih besar

CONTOH PROGRAM (*Squential*)

Contoh 1:

* Input Banyak Data: 8
* Input Kumpulan Data: 10 5 8 12 15 22 24 18
* Input Data yang Dicari: 22
* Hasil Pencarian: Data Ditemukan pada Indeks ke - 5

Kode Program:

#include <stdio.h>

//visit us @hobingoding.com

int main(){

int banyakData, i, cari;

int ketemu = 0;

printf("Pencarian Data dengan Algoritma Sequential Search\n\n");

printf("Input Banyak Data\t: ");

scanf("%d", &banyakData);

int data[banyakData];

printf("Input Kumpulan Data\t: ");

for(i = 0; i < banyakData; i++)

{

scanf("%d", &data[i]);

}

printf("Input Data yang Dicari : ");

scanf("%d", &cari);

for(i = 0; i < banyakData; i++)

{

if(cari == data[i])

{

ketemu = 1;

printf("Hasil Pencarian \t: Data Ditemukan pada Indeks ke - %d\n", i);

break;

}

}

if(ketemu == 0)

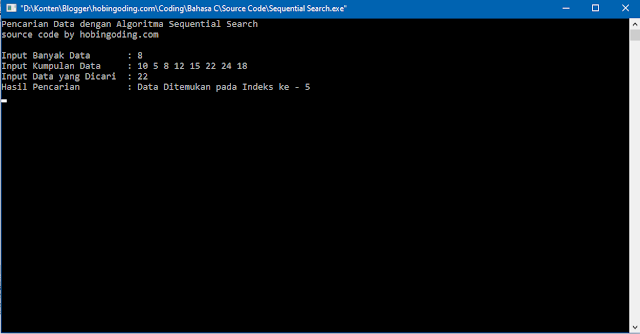
printf("Hasil Pencarian \t : Data Tidak Ditemukan\n");

getch();

return 0;

}

Output Program:



Penjelasan Program

Saya disini membuat sebuah array sepanjang / sebanyak n (int data[n]) yang dimana data yang diinput oleh user nantinya akan disimpan pada tiap-tiap indeks array (indeks ke - 0 hingga indeks n - 1).

Kemudian saya meminta inputan lagi (scanf cari) agar user dapat menginput data yang akan dicari. Lalu saya menggunakan perulangan sebanyak n data untuk mencari data yang akan dicari dengan cara membandingkan / mencocokkan data yang dicari dengan data yang ada pada masing-masing indeks array.

Sebelumnya saya telah menginsiasikan variabel ketemu = 0 (data tidak ditemukan) terlebih dahulu yang dimana variabel ini nantinya akan saya gunakan untuk melakukan pencetakan keluaran. Jika data yang dicari ada pada array / ditemukan, maka nilai ketemu akan berubah menjadi 1, kemudian dilanjutkan dengan melakukan pencetakan keterangan posisi indeks data, lalu program akan berhenti karena adanya fungsi break.

Jika data belum / tidak ditemukan, maka pencarian akan dilakukan hingga indeks terakhir. Apabila semua data telah diperiksa / dicocokkan dan data belum ketemu, maka nilai ketemu akan tetap = 0 yang mana jika ketemu = 0 maka akan dicetak keluaran data tidak ditemukan.

Contoh 2:

Kode Program:

#include <stdio.h>

int main() {

int data[100], cari, i, n;

printf("Masukkan banyaknya jumlah data : ");

scanf("%d", &n);

printf("Input setiap data integer sebanyak %d :\n", n);

**for** (i = 0; i < n; i++){

scanf("%d", &data[i]);

}

printf("Input data yang ingin dicari: ");

scanf("%d", &cari);

**for** (i = 0; i < n; i++) {

**if** (data[i] == cari) {

printf("%d berada di indeks ke %d.\n", cari, i+1);

break;

}

}

**if** (i == n){

printf("%d tidak ada.\n", cari);

}

**return** 0;

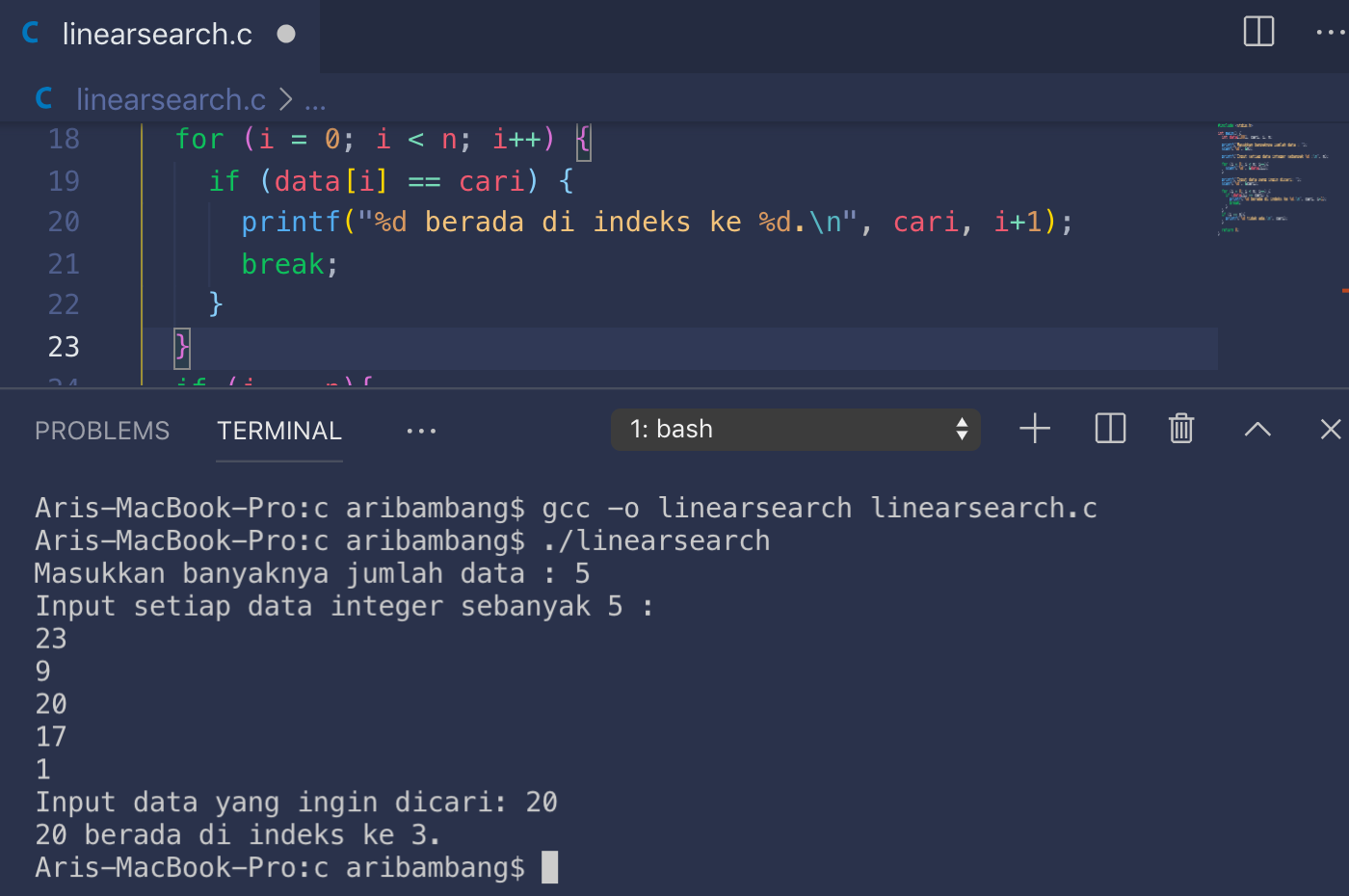
}

Penjelasan:

Berikut adalah penjelasan source code programnya :

* Baris 1 : Meng-import library stdio.h untuk input output.
* Baris 3 : Fungsi main program.
* Baris 4 : Mendeklarasikan variable data bertipe array integer untuk menampung data-data sebanyak 100 index, variable cari bertipe integer dimana nilai untuk data yang akan dicari, variable i untuk perulangan dan variable n dimana nilainya merupakan banyaknya jumlah data.
* Baris 6 – 7 : Melakukan input nilai ke variable n.
* Baris 9 – 13 : Melakukan input nilai setiap data ke dalam array.
* Baris 15 – 16 : Melakukan input nilai ke variable cari.
* Baris 18 – 23 : Melakukan perulangan sebanyak variable n dan ketika kondisi dimana nilai dari array indeks ke i sama dengan nilai variable cari maka data tersebut berada di array tersebut.
* Baris 24 – 26 : Jika perulangan sampai dengan nilai variable n makan data yang dicari tidak ada di array tersebut.

Output:



Contoh 3:

#include <stdio.h>

int linearSearch(int data[],int datayangdicari){

//looping untuk mencari datanya

int bykdata;

for(int i= 0; i<bykdata;i++){

if(data[i]==datayangdicari)return i;

}

return -1;

}

int main (){

int data[4]={9,11,1,8}; //data

int indexSearch = linearSearch(data,1); //(data, angka yang dicari)

if(indexSearch == -1){

printf("Data tidak ditemukan dalam array\n");

}

else{

printf("Data %d ditemukan dalam array\n",data[indexSearch]);

}

return 0;

}

**Penjelasan :**

* Pada function linearSearch terdapat parameter int data[], int data yangdicari yang nantinya digunakan untuk dipanggil ke function main.
* Karena linearSearch mencari data dengan memeriksa array satu per satu maka dibutuhkan looping pada function linearSearch.
* Pada function main berfungsi memanggil function linearSearch, mendeklarasikan data, dan melakukan seleksi apakah ada data yang ingin dicari?.

Output :



Kesimpulan (*Squential search*)

* Sequential Search adalah teknik pencarian data dimana data dicari secara urut daridepan ke belakang atau dari awal sampai akhir.
* Kelebihan dari proses pencarian secara sequential ini jika data yang dicari terletakdidepan, maka data akan ditemukan dengan cepat.

2. Binary Search

Metode pencarian yang kedua adalah binary search, pada metode pencarian ini, data harus diurutkan terlebih dahulu. Pada metode pencarian ini, data dibagi menjadi dua bagian (secara logika), untuk setiap tahap pencarian. Algoritma binary search :

* 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.

Pencarian Biner (Binary Search) dilakukan untuk :

* Memperkecil jumlah operasi pembandingan yang harus dilakukan antara data yang dicari dengan data yang ada di dalam tabel, khususnya untuk jumlah data yang sangat besar ukurannya.
* Beban komputasi juga lebih kecil karena pencarian dilakukan dari depan, belakang, dan tengah.
* Prinsip dasarnya adalah melakukan proses pembagian ruang pencarian secara berulang-ulang sampai data ditemukan atau sampai ruang pencarian tidak dapat dibagi lagi (berarti ada kemungkinan data tidak ditemukan).
* Syarat utama untuk pencarian biner adalah data di dalam tabel harus sudah terurut.

Kelebihannya :

* Untuk data dalam jumlah besar, waktu searching lebih cepat
* Beban komputasi lebih kecil

Kekuranganya :

* Data harus sudah di-sorting lebih dulu (dalam keadaan terurut).

Contoh Program (*binary search*)

Contoh 1:

#include<stdio.h>

int binaryIteratif(int data[],int n,int cari,int kiri,int kanan){

int mid,index=0;

while(kiri<=kanan){

mid=(kiri+kanan)/2;

if(cari==data[mid]){

return mid;

}

else if(cari>data[mid]){

kiri = mid+1;

}

else{

kanan = mid -1;

}

}

return -1;

}

int main(){

int data[100];

int n;

int cari;

int index;

int kiri,kanan;

printf("Masukkan banyaknya data : ");

scanf("%d",&n);

printf("Masukkan %d angka : ", n);

for(int i=0;i<n;i++){

scanf("%d",&data[i]);

}

printf("Masukkan angka yang ingin dicari: ");

scanf("%d",&cari);

kiri=0,kanan=n-1;

index=binaryIteratif(data,n,cari,kiri,kanan);

if(index==0){

printf("Data tidak ditemukan.\n");

}

else{

printf("Angka %d ditemukan pada index ke %d\n",cari,index);

}

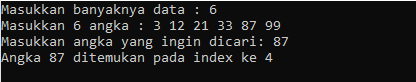
return 0;

}

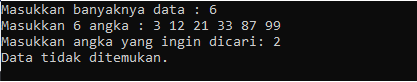
Penjelasan :

* Perbedaan paling mencolok antara binary search rekursif dan iteratif terletak pada function binarysearch dimana rekursif memanggil diri sendiri untuk mengeksekusi angka yang dicari sedangkan iteratif menggunakan looping untuk mengeksekusi angka yang dicari.
* int n berfungsi sebagai banyaknya data yang ingin dimasukkan.
* Int kiri, kanan , mid berfungsi untuk mencari angka yang di inginkan menggunakan rumus pada binary search.
* Int data[100] berfungsi untuk memasukkan data.
* Int cari berfungsi untuk memasukkan angka yang ingin di cari.

Output :



Jika data yang dicari tidak ditemukan :



Contoh 2:

#include<stdio.h>

int main()

{

//deklarasi variabel

int A[10], i,j,k,tkr,top,bottom,middle,tm;

//proses penginputan data

for(i=0;i<10;i++)

{

printf("Data ke-%d:",i+1);

scanf("%d",&A[i]);

}

printf("Masukkan data yang akan anda cari:");

scanf("%d",&k);

//proses pengurutan data

for(i=0;i<10;i++)

{

for(j=i+1;j<10;j++)

{

if (A[i]>A[j])

{

tkr=A[i];

A[i]=A[j];

A[j]=tkr;

}

}

}

//proses pencarian data

tm=0;

top=9;

bottom=0;

while(top>=bottom)

{

middle=(top+bottom)/2;

if(A[middle]==k)

{

tm++;

}

if(A[middle]<k)

{

bottom=middle+1;

}

else

{

top=middle-1;

}

}

if (tm>0)

{

printf("Data %d yang dicari ada dalam array\n",k);

}

//jika tidak ditemukan

else

{

printf("Data tidak ditemukan dalam array\n");

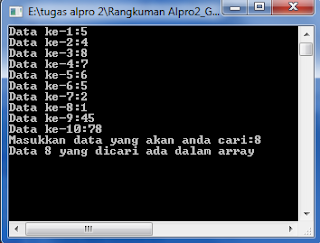
}

getch();

return 1;

}

Output :



Contoh program binary search (iteratif) :

#include<stdio.h>

int binaryIteratif(int data[],int n,int cari,int kiri,int kanan){

int mid,index=0;

while(kiri<=kanan){

mid=(kiri+kanan)/2;

if(cari==data[mid]){

return mid;

}

else if(cari>data[mid]){

kiri = mid+1;

}

else{

kanan = mid -1;

}

}

return -1;

}

int main(){

int data[100];

int n;

int cari;

int index;

int kiri,kanan;

printf("Masukkan banyaknya data : ");

scanf("%d",&n);

printf("Masukkan %d angka : ", n);

for(int i=0;i<n;i++){

scanf("%d",&data[i]);

}

printf("Masukkan angka yang ingin dicari: ");

scanf("%d",&cari);

kiri=0,kanan=n-1;

index=binaryIteratif(data,n,cari,kiri,kanan);

if(index==0){

printf("Data tidak ditemukan.\n");

}

else{

printf("Angka %d ditemukan pada index ke %d\n",cari,index);

}

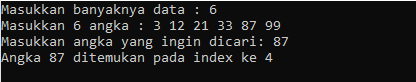
return 0;

}

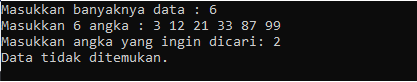
Penjelasan :

* Perbedaan paling mencolok antara binary search rekursif dan iteratif terletak pada function binarysearch dimana rekursif memanggil diri sendiri untuk mengeksekusi angka yang dicari sedangkan iteratif menggunakan looping untuk mengeksekusi angka yang dicari.
* int n berfungsi sebagai banyaknya data yang ingin dimasukkan.
* Int kiri, kanan , mid berfungsi untuk mencari angka yang di inginkan menggunakan rumus pada binary search.
* Int data[100] berfungsi untuk memasukkan data.
* Int cari berfungsi untuk memasukkan angka yang ingin di cari.

Output :



Jika data yang dicari tidak ditemukan :



Kelebihan:

* Keunggulan utama dari algoritma binary search adalah kompleksitas algoritmanya yang lebihkecil daripada kompleksitas algoritma sequential search. Hal ini menyebabkan waktu yangdibutuhkan algoritma binary search dalam mencari sebuah record dalam sebuah table, lebih kecildaripada waktu yang dibutuhkan algoritma sequential search.

Kekurangan:

* Data harus disorting dahulu dan algoritma lebih rumit, tidak baik untuk data berantai. algoritmaini hanya bisa digunakan pada tabel yang elemennya sudah terurut baik menaik maupunmenurun.

3. Interpolation search

InterpolationSearch merupakan sebuah teknik pengembangan dari binary search. Teknik binary search akan selalu memeriksa nilai tengah dari setiap array, sedangkan interpolation search dapat pergi ke lokasi yang berbeda berdasarkan key yang didapat. Jika nilai key lebih dekat ke array yang terakhir, maka teknik interpolation search akan memulai pencarian dari array yang terakhir.

Proses pencarian data ini hampir sama dengan proses pencarian binary search, pencarian ini juga dilakukan pada kumpulan data yang sudah urut. Akan tetapi jika pada binary search kita membagi data menjadi 2 bagian tiap prosesnya, pada interpolation search kita akan membagi data menurut rumus sebagai berikut:

Posisi = ( kunci – data[low] / data[high] – data[low] ) \* ( high – low ) + low

Singkatnya proses pencarian interpolation search hampir mirip dengan proses pencarian kata dikamus, yaitu kita mencari data yang dimaksud dengan cara memperkirakan letak data.

Contoh Program (*Interpolation search*)

Contoh 1:

#include<stdio.h>

int main()

{

//deklarasi variable

int A[10], i,j,k,tkr,low,high,pos,tm;

//proses penginputan data

for(i=0;i<10;i++)

{

printf("data ke-%d:",i+1);

scanf("%d",&A[i]);

}

//Input data yang akan dicari

printf("Masukkan data yang akan anda cari:");

scanf("%d",&k);

//proses pengurutan data

for(i=0;i<10;i++)

{

for(j=i+1;j<10;j++)

{

if (A[i]>A[j])

{

tkr=A[i];

A[i]=A[j];

A[j]=tkr;

}

}

}

//proses pencarian data

tm=0;

high=9;

low=0;

do

{

pos = ((k - A[low]) / (A[high] - A[low]))\*(high-low) + low;

if (A[pos] == k)

{

tm++;

break;

}

if (A[pos] > k)

high = pos-1;

else

if (A[pos] < k)

low = pos + 1;

}

while(k >= A[low] && k <= A[high]);

if (tm>0)

{

printf("data %d yang dicari ada dalam array\n",k);

}

//jika tidak ditemukan

else

{

printf("data tidak ditemukan dalam array\n");

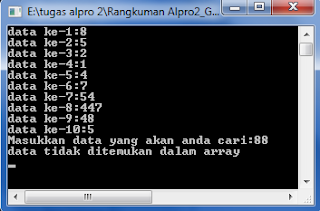
}

getch();

return 1;

}

Output:



Contoh 2:

#include<stdio.h>

int main()

{

//deklarasi variable

int A[10], i,j,k,tkr,low,high,pos,tm;

//proses penginputan data

for(i=0;i<10;i++)

{

printf(“data ke-%d:”,i+1);

scanf(“%d”,&A[i]);

}

//Input data yang akan dicari

printf(“Masukkan data yang akan anda cari:”);

scanf(“%d”,&k);

//proses pengurutan data

for(i=0;i<10;i++)

{

for(j=i+1;j<10;j++)

{

if (A[i]>A[j])

{

tkr=A[i];

A[i]=A[j];

A[j]=tkr;

}

}

}

//proses pencarian data

tm=0;

high=9;

low=0;

do

{

pos = ((k – A[low]) / (A[high] – A[low]))\*(high-low) + low;

if (A[pos] == k)

{

tm++;

break;

}

if (A[pos] > k)

high = pos-1;

else

if (A[pos] < k)

low = pos + 1;

}

while(k >= A[low] && k <= A[high]);

if (tm>0)

{

printf(“data %d yang dicari ada dalam array\n”,k);

}

//jika tidak ditemukan

else

{

printf(“data tidak ditemukan dalam array\n”);

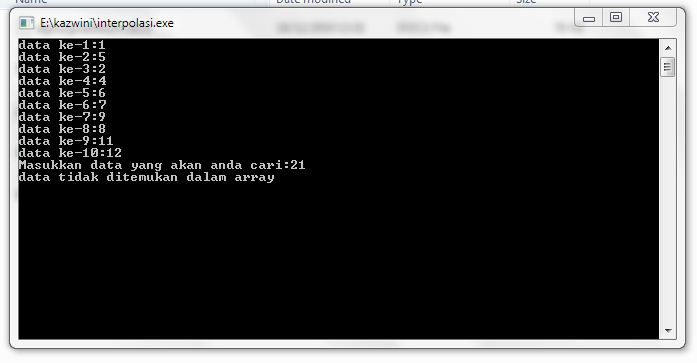
}

getch();

return 1;

}

Output:



Contoh 3:

#include<stdio.h>

#include <conio.h>

int main()

{

//deklarasi variable

int A[10], i,j,k,tkr,low,high,pos,tm;

//proses penginputan data

for(i=0;i<10;i++)

{

printf("data ke-%d : ",i+1);

scanf("%d",&A[i]);

}

//Input data yang akan dicari clrscr();

printf("\n Masukkan data yang akan anda cari : ");

scanf("%d",&k);

//proses pengurutan data

for(i=0;i<10;i++){

for(j=i+1;j<10;j++){

if (A[i]>A[j]){

tkr=A[i]; A[i]=A[j]; A[j]=tkr;

}

}

}

//proses pencarian data

tm=0;

high=9; low=0; do

{

pos = ((k - A[low]) / (A[high] - A[low]))\*(high-low) + low;

if (A[pos] == k){

tm++;

break;

}

if (A[pos] > k)

high = pos-1;

else if (A[pos] < k)

low = pos + 1;

}

while(k >= A[low] && k <= A[high]);

if (tm>0){

printf(" data %d yang dicari ada dalam array\n",k);

printf(" ===================================");

}else{

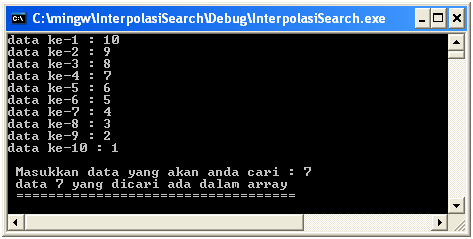
printf("data tidak ditemukan dalam array\n");}

//scanf("%d");

return(0);

}

Output:



4. Fibonacci Search

Fibonacci Search adalah pencarian sebuah elemen dalam array satu dimensi dengan menggunakan angka fibonacci sebagai titik-titik (indeks) elemen array yang isinya dibandingkan dengan nilai yang dicari.Sama halnya dengan Binary Search, Fibonacci Search juga mengharuskan data yang sudah terurut baik menaik (ascending) maupun menurun (descending).

Contoh Program (*Fibonacci search*)

Contoh 1:

#include <stdio.h>

#include <stdlib.h>

/\*program menampilkan deret bilangan fibonachi dengan menggunakan perlulangan

tampa array \*/

int main()

{

int nilai1=0,nilai2=1,nilai3,banyak; // pendeklarasian variabel yang dibutuhkan

printf( “=====================\n”

“|| Deret Fibonacci ||\n”

“=====================\n”);

printf(“\nMasukkan banyak bilangan fibonacci: “);scanf(“%d”,&banyak);

for(int batas=0;batas<banyak;batas++)

{

if(batas==0)

{

nilai3=nilai1;

printf(“%d “,nilai3);

}

else if(batas==1)

{

nilai3=nilai2;

printf(“%d “,nilai3);

}

else if(batas>1)

{

nilai3=nilai1+nilai2; // rumus untuk menentukan nilai berikutnya

nilai1=nilai2; // cara mengubah nilai posisi masing-masing bilangan

nilai2=nilai3;

printf(“%d “,nilai3);

}

}

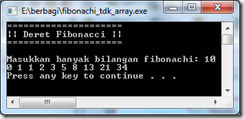
printf(“\n”);

system(“PAUSE”);

return 0;

}

Output :



Contoh 2:

#include <stdio.h>

int main (){

int i; //variabel counter perulangan

int a = 0; //variabel bilangan ke 1

int b = 1; //variabel bilangan ke 2

int c; //variabel hasil jumlah dari dua bilangan sebelumnya

int mau; //banyak bilangan yang diinginkan dalam deret

int jumlah=1; //jumlah awal -> bilangan ke 1 + bilangan ke 2 (0+1=1)

printf ("masukan banyaknya baris fibonaci yang diinginkan: ");

//user diminta memasukan banyaknya bilangan yang diinginkan dalam baris

scanf ("%d",&mau); //user memberi masukan dalam bentuk integer

printf("\n");

printf ("%d ",a); //mencetak bilangan awal (ke 1)

printf ("%d ",b); //mencetak bilangan ke 2

for (i=0;i<mau-2;i++){ //pengulangan

c=a+b;

//varibel c, hasil dari penjumlahan dua bilangan sebelumnya(a+b)

printf ("%d ",c);

//mencetak hasil jumlah dari dua bilangan sebelumnya [c]

jumlah = jumlah + c;

//penjumlahan menghitung jumlah keseluruhan bilangan dalam deret

a=b;

//variabel a berubah nilai menjadi nilai pada variabel b

b=c;

//variabel b berubah nilai menjadi nilai pada variabel c

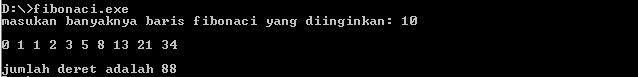
}

printf ("\n\njumlah deret adalah %d",jumlah); //mencetak hasil penjumlahan keseluruhan bilangan dalam deret”);

return 0;

}

Output :



BAB III

PENUTUP

Demikianlah Makalah ini saya buat dengan sebaik-baiknya. Terima kasih buat Orang tua yang selalu mendukung saya baik materil maupun moril,dan terima kasih untuk abang dan kakak assisten laboratorium paket applikasi, juga terima kasih buat kawan-kawan yang telah membantu dalam mengerjakan makalah ini.semoga makalah ini dapat berguna buat kita semua dan buat angkatan ke depannya kelak. lebih dan kurangnya saya mohon maaf, kepada abang dan kakak saya ucapkan terima kasih.

Wassalammu’alaikum Wr.Wb.

DAFTAR PUSTAKA

* <https://arsuwasite.wordpress.com/2012/10/04/membuat-deret-fibonacci-dengan-bahasa-c/>
* <https://renisacadel.wordpress.com/2011/06/30/deret-bilangan-fibonacci-dengan-bahasa-c/>
* <https://docplayer.info/58310348-Bab-3-searching-a-tujuan.html>
* <https://mikirinkode.com/interpolation-search/>
* <https://www.academia.edu/34659664/Makalah_Sistem_Berkas_Binary_Search>
* <https://www.sobatambisius.com/2021/09/belajar-bahasa-c-13-searching.html>
* <http://technopobia.blogspot.com/2015/05/binary-search-dengan-bahasa-c.html>
* <https://www.academia.edu/10686626/Laporan_Algoritma_dan_Struktur_Data_Modul_3>
* <https://www.sobatambisius.com/2021/09/belajar-bahasa-c-13-searching.html>
* <http://imeldapasaribu.blogspot.com/2014/06/pencarian-searching-dalam-algoritma.html>