**LAPORAN**

**SRUKTUR DATA**

**“PRAKTIKUM & TUGAS BKPM”**



**Oleh :**

**Nama : Ishaq Husainy Nuryanto**

**GOLONGAN : MIF B**

**NIM : E31180742**

**PROGRAM STUDI MANAJEMEN INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI JEMBER**

**2019**

# PRAKTIKUM 4 : Searching Array (Sequential Search)

1. **Tujuan Instruksional Khusus**

* Mahasiswa mampu membuat program searching (Sequential Search)
* Mahasiswa mampu menyebutkan kelebihan dan kekurangan metode sequential searching.

1. **Dasar Teori**

**Pengertian Searching**

• Pada suatu data seringkali dibutuhkan pembacaan kembali informasi (retrieval information) dengan cara searching

• Searching adalah pencarian data dengan menelusuri tempat pencarian data tersebut. • Tempat pencarian data tersebut dapat berupa array dalam memori, bisa juga pada file pada external storage.

**Sequential Search**

Sequential Search 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

1. **Pelaksanaan Praktikum**

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int Data[MAX];

//deklarasi sequential

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;

}

int main()

{

int i;

//pembangkit bilangan random

srand(0);

//membangkitkan bilangan integer random

printf("\nDATA : ");

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

{

Data[i] = rand()/1000+1;

printf("%d ", Data[i]);

}

//deklarasi kunci untuk mencari bilangan

int Kunci;

printf("\nKunci : ");

scanf("%d", &Kunci);

//pemanggilan fungsi Sequential

int ketemu = SequentialSearch(Kunci);

if(ketemu>=0)

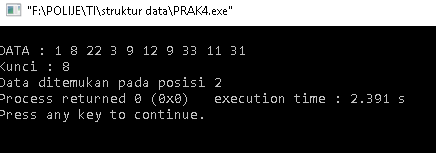
printf("Data ditemukan pada posisi %d", ketemu+1);

else

printf("Data tidak ditemukan");

}

**OUPUT :**



**Penjelasan :** Program tersebut merupakan contoh penggunaan Sequential Search yaitu merupakan 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.

**TUGAS**

1. **Buat Program untuk melakukan pencarian nama yang telah anda inputkan! (minimal 5 nama).**

#include <stdio.h>

#include <stdlib.h>

struct data{

char nama[30];

int id;

};

//inisialisasi array of struct//

struct data orang[5];

int main(){

int i, idCari;

//perulangan untuk input data//

for (i=0;i<5;i++){

orang[i].id = i;

printf("Masukkan nama pada id ke %i: ", i);

gets(orang[i].nama);

fflush(stdin);

}

//input id yang dicari//

printf("Masukkan id nama yang dicari: ");

scanf("%d", &idCari); //mencari dg memasukkan id//

//pencarian data dg sequential search//

for(i=1;i<=5;i++){

if(orang[i+1].id == idCari){

//Menampilkan nama sesuai Id//

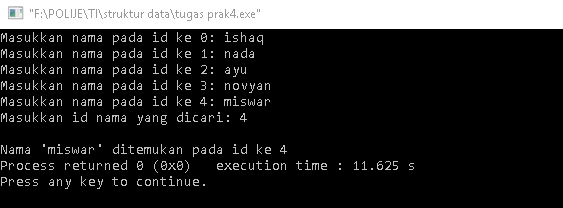
printf("\nNama '%s' ditemukan pada id ke %i", orang[i+1].nama, i+1);

}

}

}

**OUTPUT :**



**Penjelasan :** Pada program diatas kita dapat menginputkan nama sejumlah banyaknya array, lalu dengan menggunakan metode sequential search kita dapat mencari Nama berdasarkan id indeks nya.

**PRAKTIKUM 5 : Searching Array (Binary Search)**

1. **Tujuan Instruksional Khusus**

* Mahasiswa mampu membuat program dan menggunakan metode Binary Search
* Mahasiswa mampu menyebutkan kelebihan dan kekurangan metode Binary Search

1. **Dasar Teori**

Binary Search Adalah teknik pencarian data dalam dengan cara membagi data menjadi dua bagian setiap kali terjadi proses pengurutan.Data yang ada harus diurutkan terlebih dahulu berdasarkan suatu urutan tertentu yang dijadikan kunci pencarian.

1. **Pelaksanaan Praktikum**

#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

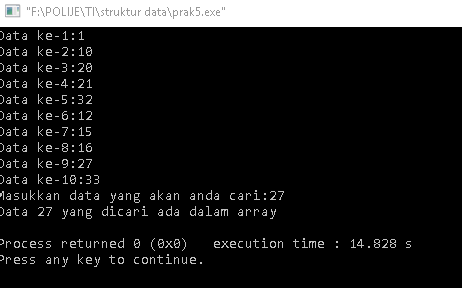
{

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

}

}

**OUTPUT :**



**Penjelasan :**

Program diatas adalah Program Binary Search yang cara kerjanya menggunakan teknik pencarian data dalam dengan cara membagi data menjadi dua bagian setiap kali terjadi proses pengurutan. Data yang ada harus diurutkan terlebih dahulu berdasarkan suatu urutan tertentu yang dijadikan kunci pencarian.

**TUGAS**

1. **Ganti coding program diatas agar array yang ada bisa ditentukan ukurannya oleh user (array dinamis) [gunakan template program yang telah ada].**

#include<stdio.h>

int main()

{

//deklarasi variabel

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

//proses penginputan data

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

scanf("%d",&jumlahData);

for(i=0;i<jumlahData;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;

}

}

//Jika ketemu

if (tm>0)

{

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

}

//jika tidak ditemukan

else

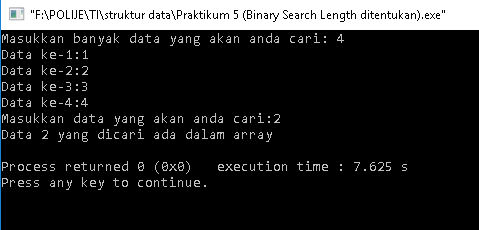
{

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

}

}

**OUTPUT :**



**Penjelasan :** Secara kinerja program ini sama halnya dengan latihan praktikum diatas karena sama-sama menggunakan Binary Search, namun perbedaannya terdapat pada cara menginputkan jumlah data yang dicari pada program tugas kita bisa menulis sendiri banyak data yang ingin dicari.

**PRAKTIKUM 6 : Buble Sorting**

1. **Tujuan Instruksional Khusus**

* Memahami karakteristik algoritma pengurutan buble sort.
* Mengimplementasikan algoritma buble sort dalam bentuk flowchart.
* Membuat diagram alir dan mengimplementasikan algoritma pada suatu permasalahan

1. **Dasar Teori**

Pengurutan data (sorting) didefinisikan sebagai suatu proses untuk menyusun kembali humpunan obyek menggunakan aturan tertentu. Ada dua macam urutan yang biasa digunakan dalam proses pengurutan yaitu

* urut naik (ascending) yaitu dari data yang mempunyai nilai paling kecil sampai paling besar
* urut turun (descending) yaitu data yang mempunyai nilai paling besar sampai paling kecil.

Bubble Sort Metode gelembung (bubble sort) sering juga disebut dengan metode penukaran (exchange sort) adalah metode yang mengurutkan data dengan cara membandingkan masing-masing elemen, kemudian melakukan penukaran bila perlu.

1. **Pelaksanaan Praktikum**

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int Data[MAX];

// Prosedur menukar data

void Tukar (int \*a, int \*b)

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

// Prosedur pengurutan metode gelembung

void BubbleSort()

{

int i, j;

for(i=1; i<MAX-1; i++)

for(j=MAX-1; j>=i; j--)

if(Data[j-1] > Data[j])

Tukar(&Data[j-1], &Data[j]);

}

int main()

{

int i;

srand(0);

// Membangkitkan bilangan acak

printf("DATA SEBELUM TERURUT");

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

{

Data[i] = (int) rand()/1000+1;

printf("\nData ke %d : %d ", i, Data[i]);

}

BubbleSort();

// Data setelah terurut

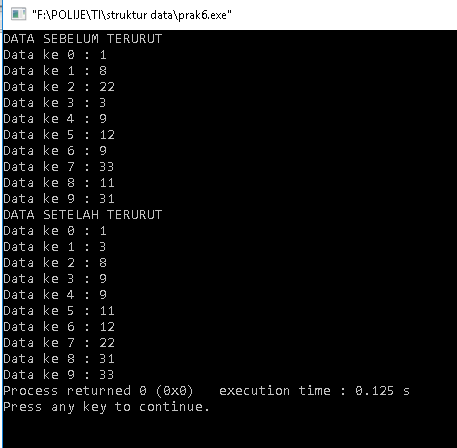
printf("\nDATA SETELAH TERURUT");

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

printf("\nData ke %d : %d ", i, Data[i]);

}

**OUTPUT :**



**Penjelasan :**

Program tersebut menggunakan metode sorting Bubble Sort Metode gelembung (bubble sort) Ascending atau sering juga disebut dengan metode penukaran (exchange sort) yang dimana sebuah metode yang mengurutkan data dengan cara membandingkan masing-masing elemen, kemudian melakukan penukaran bila perlu.

Metode ini merupakan metode pengurutan yang paling mudah, namun paling lambat dibandingkan dengan yang lain. Bubble sort akan melakukan pengurutan data dengan cara membandingkan elemen sekarang dengan elemen berikutnya. Bisa dilakukan baik dari kepala array maupun ekor array.

**TUGAS**

1. **Buatlan pengurutan 10 bilangan ganjil acak menggunakan metode buble sort.**

#include <stdio.h>

#include <stdlib.h>

#define MAX 16

int Data[MAX];

// Prosedur menukar data

void Tukar (int \*a, int \*b)

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

// Prosedur pengurutan metode gelembung

void BubbleSort()

{

int i, j;

for(i=1; i<MAX-1; i++)

for(j=MAX-1; j>=i; j--)

if(Data[j-1] > Data[j])

Tukar(&Data[j-1], &Data[j]);

}

int main()

{

int i;

srand(0);

// Membangkitkan bilangan acak

printf("DATA SEBELUM TERURUT\n");

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

{

int bilRand = (int) rand()%100;

if (bilRand%2 == 1 && bilRand != 0)

Data[i] = bilRand;

if (Data[i]%2 == 1)

printf("%d ", Data[i]);

}

BubbleSort();

// Data setelah terurut

printf("\nDATA SETELAH TERURUT\n");

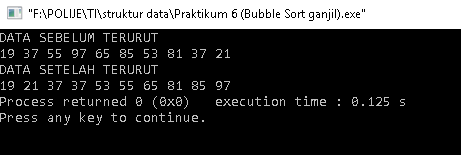
for(i=0; i<MAX; i++)

if(Data[i]%2==1)

printf("%d ", Data[i]);

}

**OUTPUT :**



**Penjelasan :**

Secara teknis sama dengan pelaksanaan Praktikum dengan menggunakan Bubble Sort untuk mengurutkan data namun pada program tugas bilangan random yang ditampilkan khusus bilangan random ganjil yang berjumlah 10 bilangan.

**PRAKTIKUM 7 : Selection Sorting**

1. **Tujuan Instruksional Khusus**

* Memahami karakteristik algoritma pengurutan selection sort.
* Mengimplementasikan algoritma selection sort dalam bentuk flowchart.
* Membuat diagram alir dan mengimplementasikan algoritma pada suatu permasalahan.

1. **Dasar Teori**

Pengurutan data (sorting) didefinisikan sebagai suatu proses untuk menyusun kembali humpunan obyek menggunakan aturan tertentu. Ada dua macam urutan yang biasa digunakan dalam proses pengurutan yaitu :

* urut naik (ascending) yaitu dari data yang mempunyai nilai paling kecil sampai paling besar
* urut turun (descending) yaitu data yang mempunyai nilai paling besar sampai paling kecil

**Selection Sort**

Selection Sort merupakan kombinasi sorting dan searching. Untuk setiap proses, akan dilakukan dengan mencari elemen dari posisi yang belum diurutkan dan kemudian memilih elemen yang memiliki nilai terkecil atau terbesar yang akan ditukarkan ke posisi yang tepat di dalam array.

1. **Pelaksanaan Praktikum**

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int Data[MAX];

// Fungsi pertukaran bilangan

void Tukar (int \*a, int \*b)

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

// Fungsi pengurutan penyisipan biner

void SelectionSort()

{

int i, j, k;

for(i=0; i<MAX-1;i++){

k = i;

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

if(Data[k] > Data[j])

k = j;

Tukar(&Data[i], &Data[k]);

}

}

int main()

{

int i;

srand(0);

// Membangkitkan bilangan acak

printf("DATA SEBELUM TERURUT");

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

Data[i] = (int) rand()/1000+1;

printf("\nData ke %d : %d ", i, Data[i]);

}

SelectionSort();

// Data setelah terurut

printf("\nDATA SETELAH TERURUT");

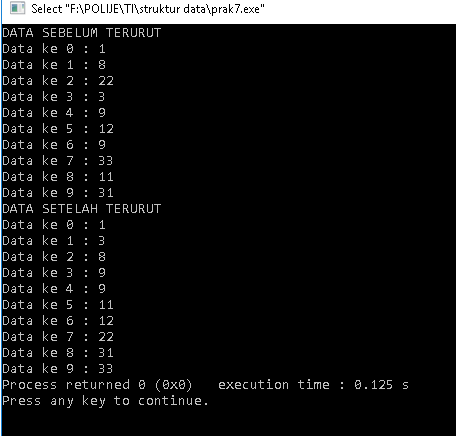
for(i=0; i<MAX; i++){

printf("\nData ke %d : %d ", i, Data[i]);

}

}

**OUTPUT :**



**Penjelasan :**

Program ini menggunakan metode Selection Sort yang merupakan merupakan kombinasi sorting dan searching. Jadi untuk setiap proses, akan dilakukan dengan mencari elemen dari posisi yang belum diurutkan dan kemudian memilih elemen yang memiliki nilai terkecil atau terbesar yang akan ditukarkan ke posisi yang tepat di dalam array.

Misalnya untuk putaran pertama, akan dicari data dengan nilai terkecil dan data ini akan ditempatkan pada indeks terkecil, pada putaran kedua akan dicari data kedua terkecil, dan akan ditempatkan di indeks kedua, negitu seterusnya hingga tidak ada data yang dicari lagi. Selama proses, pembandingan dan pengubahan hanya dilakukan pada indeks pembanding saja, pertukaran data secara fisik terjadi pada akhir proses.

**TUGAS**

1. **Buatlah Algoritma dan Flowchart pengurut 20 bilangan ganjil acak menggunakan metode selection sort.**

#include <stdio.h>

#include <stdlib.h>

#define MAX 34

int Data[MAX];

//fungsi tukar

void Tukar (int \*a, int \*b)

{

//penampungan sementara

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

//fungsi selectionsort

void SelectionSort()

{

int i, j, k;

for(i=0; i<MAX-1;i++){

k = i;

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

if(Data[k] > Data[j])

k = j;

//pemanggilan fungsi tukar

Tukar(&Data[i], &Data[k]);

}

}

int main()

{

int i;

srand(0);

printf("DATA SEBELUM TERURUT\n\n");

//pemanggilan bilangan ganjil random

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

int bilRand = (int) rand()%100;

if (bilRand%2 == 1 && bilRand != 0)

Data[i] = bilRand;

if (Data[i]%2 == 1)

printf("%d ", Data[i]);

}

//pemanggilan selectionsort

SelectionSort();

printf("\n\nDATA SETELAH TERURUT\n\n");

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

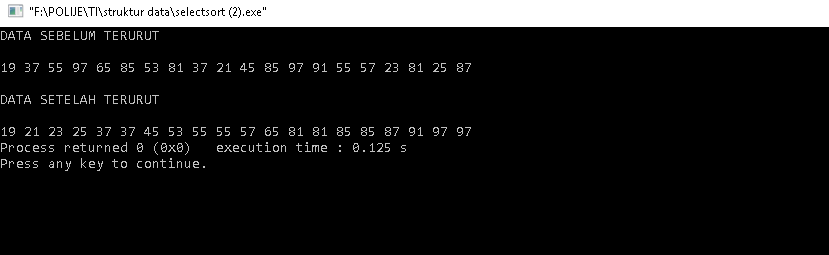
if (Data[i]%2 == 1)

printf("%d ", Data[i]);

}

}

**OUTPUT :**



**Penjelasan :**

Program ini menggunakan metode Selection Sort yang merupakan merupakan kombinasi sorting dan searching untuk mengurutkan 20 bilangan ganjil random (acak ). Jadi untuk setiap proses, akan dilakukan dengan mencari elemen dari posisi yang belum diurutkan dan kemudian memilih elemen yang memiliki nilai terkecil atau terbesar yang akan ditukarkan ke posisi yang tepat di dalam array.

Misalnya untuk putaran pertama, akan dicari data dengan nilai terkecil dan data ini akan ditempatkan pada indeks terkecil, pada putaran kedua akan dicari data kedua terkecil, dan akan ditempatkan di indeks kedua, negitu seterusnya hingga tidak ada data yang dicari lagi. Selama proses, pembandingan dan pengubahan hanya dilakukan pada indeks pembanding saja, pertukaran data secara fisik terjadi pada akhir proses.