

LAPORAN TUGAS 1

Analisis Algoritma

Algoritma Pemangkatan, Linear Search, Binary Search



140810160014

SACHI HONGO

KELAS B

S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2018/2019

Tugas

- Tulis algoritma dan program memangkatkan bilangan 2 pangkat n (integer) menggunakan algoritma iteratif dan rekursif
- Buat algoritma dan program pencarian linear dan pencarian biner
- Analisis “running time” nya program-program tersebut dengan mencoba data yang berbeda dan bandingkan satu sama lain, tuliskan spesifikasi komputer yang digunakan, tuliskan kesimpulannya !!

1. Algoritma Pemangkatan

Algoritma iteratif : perulangan yang melakukan proses perulangan terhadap sekelompok intruksi. Perulangan dilakukan dalam batasan syarat tertentu. Ketika syarat tersebut tidak terpenuhi lagi maka perulangan akan berhenti.

Algoritma rekursif : salah satu metode didalam pemrograman yang mana dalam sebuah fungsi terdapat intruksi yang memanggil fungsi itu sendiri, atau lebih sering disebut memanggil dirinya sendiri.

Perbedaan :

- Iteratif menggunakan FOR, WHILE, DO-WHILE sedangkan rekursif hanya menggunakan IF.
- Iteratif dapat berjalan pada program yang terdiri dari prosedur (tidak terdapat fungsi) sedangkan rekursif merupakan fungsi.

Algoritma Pemangkatan metode Iteratif

```
/**
 *      Nama : Sachi Hongo
 *      NPM  : 140810160014
 *      Kelas : B
 *      Deskripsi : Implementasi Algoritma Pemangkatan metode Iteratif dengan running time
 */

#include<iostream>

#include <time.h>

using namespace std;
```

```
int pangkat (int n)

{int hasil=1;
for (int i=1;i<=n;i++)
{
    hasil=hasil*2;
}
return hasil;
}

int main()
{
    clock_t start, end;
    float time;
    int j;

    cout<<"PROGRAM MENGHITUNG PANGKAT"<<endl;
    cout<<endl<<"Masukkan Pangkat : ";
    cin>>j;

    start = clock();

    cout<<endl<<"Hasilnya = ";
    cout<<pangkat(j)<<endl;

    end = clock();

    //Perhitungan waktu dari clock time yang ada
    time = (float) (end - start)*1000000000000000.0;
    cout<<"Process Time : "<<time<<" s"<<endl;
}
```

Algoritma Pemangkatan metode Rekursif

```
/**
 *      Nama : Sachi Hongo
 *      NPM  : 140810160014
 *      Kelas : B
 *      Deskripsi : Implementasi Algoritma Pemangkatan metode Rekursif dengan running time
 */

#include <iostream>
#include <time.h>
using namespace std;

int Pangkat(int y){
    if (y==0){
        return 1;
    }
    else{
        return (2*Pangkat(y-1));
    }
}

int main() {

    clock_t start, end;
    float time;
    int p;

    cout<<"PROGRAM MENGHITUNG PANGKAT"<<endl;
```

```

cout<<"\nMasukkan pangkat : ";

cin>>p;

cout<<endl;

start = clock();

cout<<"Hasil = "<<Pangkat(p)<<endl;

end = clock();

//Perhitungan waktu dari clock time yang ada
time = (float) (end - start)*1000000000000000.0;

cout<<"Process Time : "<<time<<" s"<<endl;

}

```

Perbandingan Pemangkatan secara Iteratif dan Rekursif :

Iteratif

```

"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appointment.exe"
PROGRAM MENGHITUNG PANGKAT
Masukkan Pangkat : 2
Hasilnya = 4
Process Time : 1e+015 s
Process returned 0 (0x0)  execution time : 5.646 s
Press any key to continue.

```

Rekursif

```

"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appointment_rekursif.exe"
PROGRAM MENGHITUNG PANGKAT
Masukkan pangkat : 2
Hasil = 4
Process Time : 0 s
Process returned 0 (0x0)  execution time : 1.096 s
Press any key to continue.

```

Iteratif

```

"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appointment.exe"
PROGRAM MENGHITUNG PANGKAT
Masukkan Pangkat : 7
Hasilnya = 128
Process Time : 1e+016 s
Process returned 0 (0x0)  execution time : 3.649 s
Press any key to continue.

```

Rekursif

```

"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appointment_rekursif.exe"
PROGRAM MENGHITUNG PANGKAT
Masukkan pangkat : 7
Hasil = 128
Process Time : 8e+015 s
Process returned 0 (0x0)  execution time : 2.308 s
Press any key to continue.

```

Iteratif

```
"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appoinme
PROGRAM MENGHITUNG PANGKAT
Masukkan Pangkat : 11
Hasilnya = 2048
Process Time : 0 s
Process returned 0 (0x0)  execution time : 3.780 s
Press any key to continue.
```

Rekursif

```
"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appoinme
PROGRAM MENGHITUNG PANGKAT
Masukkan pangkat : 11
Hasil = 2048
Process Time : 2e+015 s
Process returned 0 (0x0)  execution time : 1.247 s
Press any key to continue.
```

Iteratif

```
"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appoinme
PROGRAM MENGHITUNG PANGKAT
Masukkan Pangkat : 29
Hasilnya = 536870912
Process Time : 0 s
Process returned 0 (0x0)  execution time : 6.672 s
Press any key to continue.
```

Rekursif

```
"E:\SACHI-140810160014\Semester 6\Analisis Algoritma\appoinme
PROGRAM MENGHITUNG PANGKAT
Masukkan pangkat : 29
Hasil = 536870912
Process Time : 0 s
Process returned 0 (0x0)  execution time : 2.577 s
Press any key to continue.
```

Dari hasil kedua program di atas, ketika inputan bernilai $N = 2, 7, 11, 29$ perbedaan di antara kedua algoritma bisa dilihat serta memiliki waktu penyelesaian yang berbeda-beda di setiap input. Hal ini mungkin terjadi karena jumlah data yang berbeda-beda dan pengerjaan compiler yang menghitung waktu sesuai input yang diberikan.

Spesifikasi Komputer :

Prosesor Intel Core™ i7-6500U CPU @2,50 GHz 2.59 GHz
Ram 8.00 GB

Analisis :

- Waktu Proses yang dibutuhkan oleh algoritma pemangkatan iteratif dan rekursif dilihat memberikan hasil yang berbeda-beda dengan inputan yang telah dicoba
- Spesifikasi Komputer saat ini cukup memberikan peranan penting dalam kebutuhan waktu proses, semakin mumpuni suatu spesifikasi komputer maka waktu proses yang dibutuhkan pun semakin sedikit

2. Algoritma Pencarian Linear dan Biner

Linear/Sequential Search

```
/**
 *      Nama : Sachi Hongo
 *      NPM  : 140810160014
 *      Kelas : B
 *      Deskripsi : Implementasi Algoritma Linear/Sequential Search dengan running time
 **/

//Menyertakan library
#include <iostream>
#include <time.h>

//Menentukan banyak data
#define MAX 500000
using namespace std;

int main()
{
    clock_t start, end;
    float time;

    //Deklarasi variabel
    int item;
    int nomor[MAX];
    bool found = false;

    //Input Data Nomor
    for(int i=0; i<=MAX ;i++){
```

```
        nomor[i] = i+1;
    }

    //Input key untuk pencarian
    cout<<"Masukkan Nomor Pencarian : ";
    cin >> item;

    start = clock();

    //Lakukan iterasi sebanyak jumlah data yang ada
    for(int i=0;i<MAX;i++)
    {
        //Logika ketika key sama dengan data pada array pada indeks i, maka data ditemukan
        if(item==nomor[i])
        {
            //Tampilan ketika data ditemukan
            found = true;

            cout << "Data ditemukan pada lokasi " << i+1 << endl;

            break;
        }
    }

    //Logika jika data tidak ditemukan
    if(found=false)
    {
        //Tampilan ketika data tidak ditemukan
        cout<<"Data tidak ditemukan";
    }

    //Akhir clock time setelah proses
    end = clock();
```



```
//Perhitungan waktu dari clock time yang ada  
time = (float) (end - start)*1000000000000000.0;  
cout<<"Process Time : "<<time<<" s";  
}
```

Binary Search

```
/**  
 *      Nama : Sachi Hongo  
 *      NPM  : 140810160014  
 *      Kelas : B  
 *      Deskripsi : Implementasi Algoritma Binary Search dengan Running Time  
 **/  
  
//Menyertakan library yang dibutuhkan  
#include <iostream>  
#include <time.h>  
  
//Menentukan banyak data  
#define MAX 500000  
  
//namespace  
using namespace std;  
int main()  
{  
    //Deklarasi clock  
    clock_t start, end;  
    double time;
```

```

//Deklarasi variabel

int nomor[MAX];

int key, low = 0, high = MAX;

bool found = false;


//Input Data Nomor

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

    nomor[i] = i+1;

}


//Input key untuk pencarian

cout<<"Masukkan Nomor Pencarian : ";

cin>>key;


//Start clock time setelah input

start = clock();

while(low<=high)

{

    //Deklarasi nilai tengah

    int mid = (low+high)/2;

    //Logika jika key sama dengan data pada array pada indeks i, maka data ditemukan

    if(key == nomor[mid])

    {

        //Tampilan ketika data ditemukan

        cout<<"Data ditemukan pada lokasi "<<mid+1<<endl;

        found = true;

        break;

    }

}

```

```

//jika key < data pada array maka high = mid - 1
else if(key<nomor[mid])
{
    high=mid - 1;
}
//jika sebaliknya key > data pada array maka low = mid + 1
else
{
    low = mid + 1;
}

if(found=false)
{
    //Tampilan ketika data tidak ditemukan
    cout<<"Data tidak ditemukan";
}
}

//Akhir clock time setelah proses
end = clock();

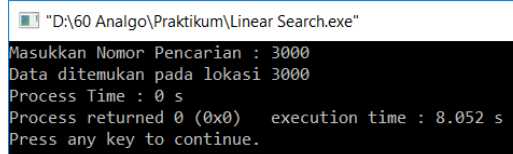
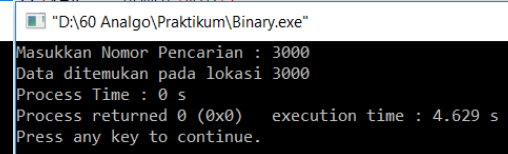
//Perhitungan waktu dari clock time yang ada
time = (float) (end - start)*1000000000000000.0;
cout<<"Process Time : "<<time<<" s";
}

```

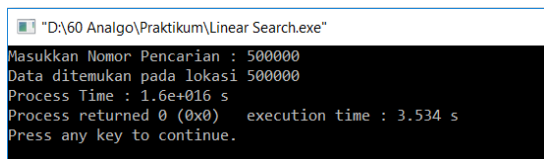
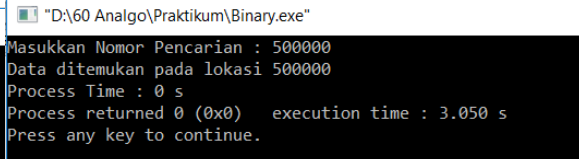
Perbandingan Linear Search dan Binary Search :

Para proses time Linear Search dan Binary search waktu proses yang dibutuhkan sangat cepat, sehingga sulit mendapatkan waktu proses yang dibutuhkan.

Pada pencarian angka-angka kecil, tidak terlihat perbedaan antara binary search dan linear search. Keduanya mencantumkan nilai 0 walau telah dikalikan dengan 1.000.000.000.000.000,0.

Linear Search	Binary Search
 <pre>"D:\60 Analgo\Praktikum\Linear Search.exe" Masukkan Nomor Pencarian : 3000 Data ditemukan pada lokasi 3000 Process Time : 0 s Process returned 0 (0x0) execution time : 8.052 s Press any key to continue.</pre>	 <pre>"D:\60 Analgo\Praktikum\Binary.exe" Masukkan Nomor Pencarian : 3000 Data ditemukan pada lokasi 3000 Process Time : 0 s Process returned 0 (0x0) execution time : 4.629 s Press any key to continue.</pre>

Sedangkan di beberapa kesempatan untuk pencarian angka – angka besar pada linear search didapat perbedaan hasil. Sementara pada binary search tidak didapatkan hasil. Namun perbedaan tidak selalu muncul dan terkadang tetap menghasilkan 0.

Linear Search	Binary Search
 <pre>"D:\60 Analgo\Praktikum\Linear Search.exe" Masukkan Nomor Pencarian : 500000 Data ditemukan pada lokasi 500000 Process Time : 1.6e+016 s Process returned 0 (0x0) execution time : 3.534 s Press any key to continue.</pre>	 <pre>"D:\60 Analgo\Praktikum\Binary.exe" Masukkan Nomor Pencarian : 500000 Data ditemukan pada lokasi 500000 Process Time : 0 s Process returned 0 (0x0) execution time : 3.050 s Press any key to continue.</pre>

Spesifikasi Komputer :

Prosesor Intel Core™ i7-6500U CPU @2,50 GHz 2.59 GHz

Ram 8.00 GB

Analisis :

- Waktu Proses yang dibutuhkan oleh Binary Search lebih sedikit jika dibandingkan dengan waktu proses yang dibutuhkan oleh Linear Search.
- Best Case pada Linear Search terjadi ketika data yang dicari berada pada urutan awal
- Best Case pada Binary Search terjadi ketika data yang dicari berada pada urutan tengah
- Spesifikasi Komputer saat ini cukup memberikan peranan penting dalam kebutuhan waktu proses, semakin mumpuni suatu spesifikasi komputer maka waktu proses yang dibutuhkan pun semakin sedikit