**TUGAS 1**

**ANALISIS ALGORITMA**

****

Disusun Oleh:

Satrio Sadrakh Allesandro

140810160038

**PRODI S-1 TEKNIK INFORMATIKA**

**DEPARTEMEN ILMU KOMPUTER**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

**2018**

**Spesifikasi Komputer:**

Model & Manufacturer : LENOVO G50-45 (80E3)

OS : Windows 10 Enterprise 64-bit (10.0, Build 17134)

Processor : AMD A8-6410 APU with AMD Radeon R5 Graphics (4CPUs), 2.0 GHz

Memory : 4096 MB RAM

Compiler : <https://www.onlinegdb.com/online_c++_compiler#>

**Program bilangan 2 pangkat n**

**Algoritma Iteratif**

Source Code:

#include <iostream>

#include <conio.h>

#include <math.h>

using namespace std;

int iteratif (int n){

for (int x=1; x<=n; x++){

cout<<pow(2,x)<<" ";

}

}

int main(){

int n;

cout<<"Masukkan n : ";

cin>>n;

cout<<endl;

clock\_t t1,t2;

t1=clock();

iteratif (n);

t2=clock();

float diff ((float)t2-(float)t1);

cout<<endl<<endl<<"n : "<<n;

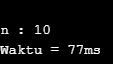
cout<<endl<<"Waktu = "<<diff<<"ms"<<endl;

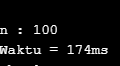
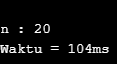
system ("pause");

return 0;

}

Hasil:



**Algoritma Rekursif**

Source Code:

#include <iostream>

#include <conio.h>

#include <math.h>

#include <time.h>

using namespace std;

int rekursif (int n){

if (n>=1){

rekursif (n-1);

cout<<pow(2,n)<<" ";

}

}

int main(){

int n;

cout<<"Masukkan n : ";

cin>>n;

cout<<endl;

clock\_t t1,t2;

t1=clock();

rekursif (n);

t2=clock();

float diff ((float)t2-(float)t1);

cout<<endl<<endl<<"n : "<<n;

cout<<endl<<"Waktu = "<<diff<<"ms"<<endl;

system ("pause");

return 0;

}

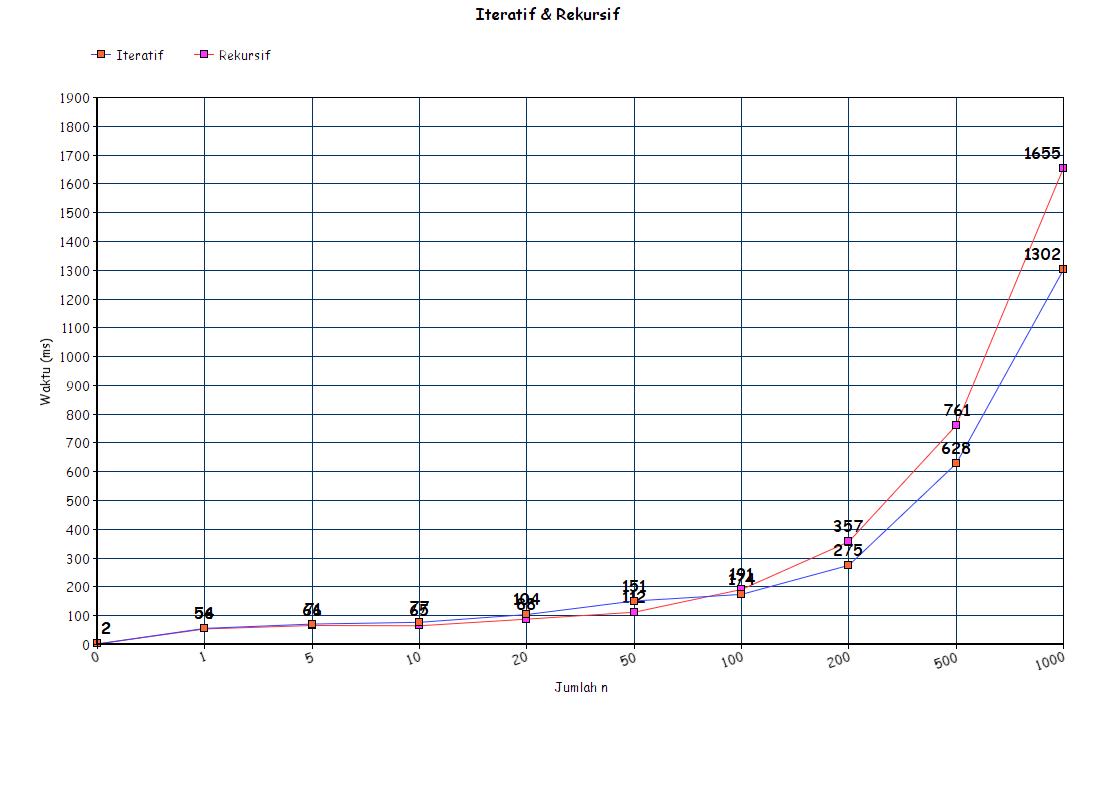
Hasil:

**Analisa & Kesimpulan**



Saat membandingkan *running time* antara algoritma iteratif dengan rekursif, terlihat bahwa bila nilai n berada di antara 0 – 50 (0 < n <= 50), maka algoritma rekursif memiliki running time yang lebih cepat disbanding dengan algoritma iteratif. Namun, sebaliknya, bila nilai n >= 100, maka algoritma iteratif memiliki running time yang lebih cepat disbanding dengan algoritma rekursif.

**Program pencarian (Searching)**

**Algoritma Pencarian Linier**

Source Code:

#include <iostream>

#include <math.h>

using namespace std;

int search(int arr[], int n, int x)

{

int i;

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

if (arr[i] == x){

return i;

}

}

return -1;

}

int main()

{

int n, x;

cout<<"Masukkan jumlah bilangan : ";

cin>>n;

cout<<"Masukkan angka yang dicari : ";

cin>>x;

int arr[n];

cout<<"Elemen array: "<<endl;

for(int x=1; x<=n; x++){

arr[x]=x;

cout<<arr[x]<<" ";

}

cout<<endl<<endl;

clock\_t t1,t2;

t1=clock();

cout<<"Jumlah bilangan: "<<n<<endl;

cout<<"Angka yang dicari: "<<x<<endl;

int result = search(arr, n, x);

if (result == -1)

cout<<"Elemen Tidak Ditemukan";

else

cout<<"Elemen berada di urutan ke: " <<result;

t2=clock();

float diff ((float)t2-(float)t1);

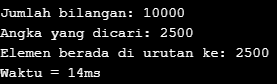
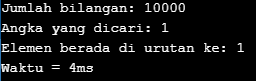
cout<<endl<<"Waktu = "<<diff<<"ms"<<endl;

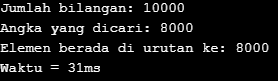
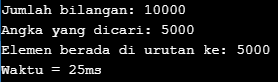
system ("pause");

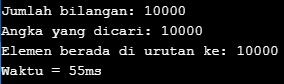
return 0;

}

Hasil:







**Algoritma Pencarian Biner**

Source Code:

#include <iostream>

#include <math.h>

using namespace std;

int binarySearch(int arr[], int l, int r, int x)

{

if (r >= l) {

int mid = l + (r - l) / 2;

if (arr[mid] == x)

return mid;

if (arr[mid] > x)

return binarySearch(arr, l, mid - 1, x);

return binarySearch(arr, mid + 1, r, x);

}

return -1;

}

int main(void)

{

int n, x;

cout<<"Masukkan jumlah bilangan : ";

cin>>n;

cout<<"Masukkan angka yang dicari : ";

cin>>x;

int arr[n];

cout<<"Elemen array: "<<endl;

for(int x=1; x<=n; x++){

arr[x]=x;

cout<<arr[x]<<" ";

}

cout<<endl<<endl;

clock\_t t1,t2;

t1=clock();

cout<<"Jumlah bilangan: "<<n<<endl;

cout<<"Angka yang dicari: "<<x<<endl;

int result = binarySearch(arr, 0, n, x);

if (result == -1)

cout<<"Elemen Tidak Ditemukan";

else

cout<<"Elemen berada di urutan ke: " <<result;

t2=clock();

float diff ((float)t2-(float)t1);

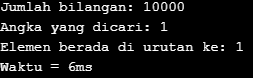
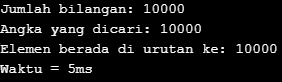
cout<<endl<<"Waktu = "<<diff<<"ms"<<endl;

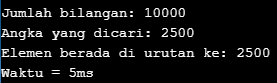
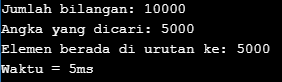
system ("pause");

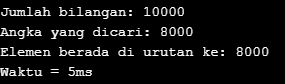
return 0;

}

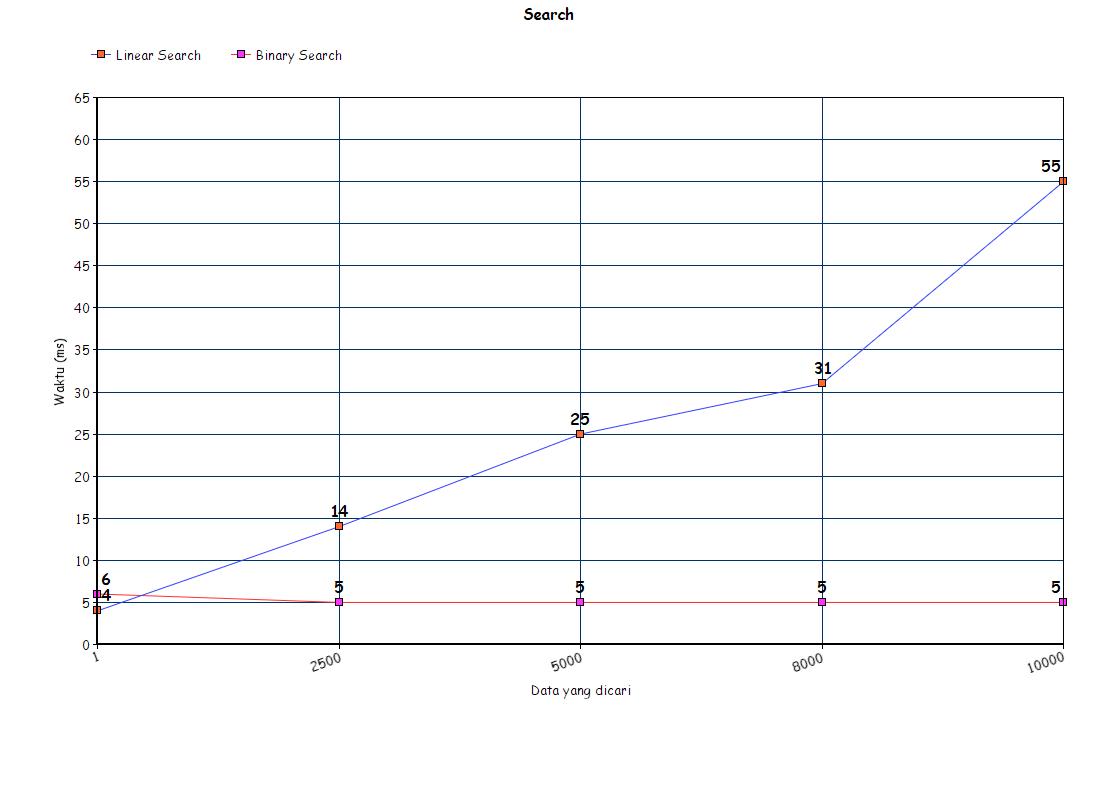
Hasil:





**Kesimpulan:**

****

Saat membandingkan *running time* antara algoritma *linear search* dengan *binary search*, terlihat bahwa algoritma *linear search* akan memiliki *running time* yang lebih cepat dibanding dengan *binary search* jika dan hanya jika data yang dicari berada di awal *list* (di gambar diwakilkan dengan data 1). Selain itu, *binary search* memiliki *running time* yang lebih cepat.