**LAPORAN PRAKTIKUM 4**

**ANALISIS ALGORITMA**

****

Disusun oleh :

Hafidh Akhdan Najib

140810180061

**PROGRAM STUDI S-1 INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

**2020**

Studi Kasus

#### Studi Kasus 1: MERGE SORT

Setelah Anda mengetahui Algoritma Merge-Sort mengadopsi paradigma divide & conquer, lakukan Hal berikut:

1. Buat program Merge-Sort dengan bahasa C++
2. Kompleksitas waktu algoritma merge sort adalah O(n lg n). Cari tahu kecepatan komputer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

/\*

Nama : Hafidh Akhdan Najib

NPM : 140810180061

Kelas : A

\*/

#include <iostream>

using namespace std;

int a[100];

void merge(int,int,int);

void merge\_sort(int low,int high)

{ int mid;

if(low<high)

{

mid=(low+high)/2;

merge\_sort(low,mid);

merge\_sort(mid+1,high);

merge(low,mid,high);

}

}

void merge(int low,int mid,int high){

int h,i,j,b[50],k;

h=low;

i=low;

j=mid+1;

while((h<=mid)&&(j<=high))

{

if(a[h]<=a[j]){

b[i]=a[h]; h++;

}

else{

b[i]=a[j]; j++;

} i++;

}

if(h>mid){

for(k=j;k<=high;k++)

{

b[i]=a[k]; i++;

}

}

else{

for(k=h;k<=mid;k++)

{ b[i]=a[k]; i++;

}

}

for(k=low;k<=high;k++)

a[k]=b[k];

}

main(){

int num,i;

cout<<"Input Banyak Bilangan : ";cin>>num;

cout<<endl;

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

cout<<"Bilangan ke-"<<i<<" : ";cin>>a[i] ;

}

merge\_sort(1,num);

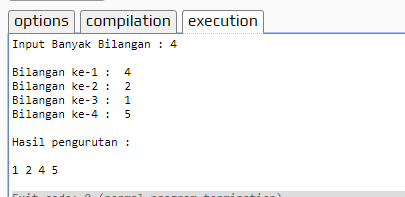
cout<<endl<<"Hasil pengurutan :"<<endl<<endl;

for(i=1;i<=num;i++)

cout<<a[i]<<" ";

cout<<endl;

}



Kompleksitas waktu algoritma merge sort adalah O(n lg n). Cari tahu kecepatan komputer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

* Menggunakan perhitungan Big O -> T(26

#### Studi Kasus 2: SELECTION SORT

Selection sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma selection sort, lakukan langkah-langkah berikut:

* Pelajari cara kerja algoritma selection sort
* Tentukan T(n) dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

T(n) = {

©(1)

n

aT ( ) + D(n)

b

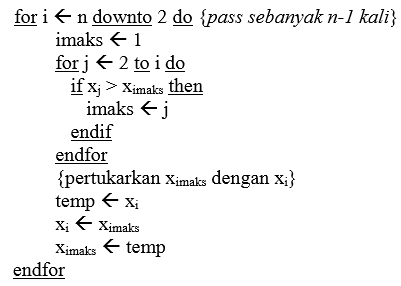
+ C(n)

### if n ≤ c

otherwise

* Selesaikan persamaan rekurensi T(n) dengan **metode recursion-tree** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
* Lakukan implementasi koding program untuk algoritma selection sort dengan menggunakan bahasa C++

**Pengerjaan :**



* + Subproblem = 1
  + Masalah setiap subproblem = n-1
  + Waktu proses pembagian = n
  + Waktu proses penggabungan = n



1. T(n) = cn + cn-c +cn-2c + ..... + 2c +cn  
   = c((n-1)(n-2)/2) + cn  
   = c((n2-3n+2)/2) + cn  
   = c(n2/2)-(3n/2)+1 + cn  
   =O(n2)
2. T(n) = cn + cn-c +cn-2c + ..... + 2c +cn  
   = c((n-1)(n-2)/2) + cn  
   = c((n2-3n+2)/2) + cn  
   = c(n2/2)-(3n/2)+1 + cn  
   = Ω (n2)
3. T(n) = cn2= Θ(n2)

/\*

Nama : Hafidh Akhdan N

NPM : 140810180061

Kelas : A

\*/

#include <iostream>

#include <conio.h>

using namespace std;

int data[50],data2[50];

int n;

void tukar(int a, int b)

{

int t;

t = data[b];

data[b] = data[a];

data[a] = t;

}

void selection\_sort()

{

int pos,i,j;

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

{

pos = i;

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

{

if(data[j] < data[pos]) pos = j;

}

if(pos != i) tukar(pos,i);

}

}

int main(){

cout<<"Input Jumlah Data : ";cin>>n;

for(int i=1;i<=n;i++)

{

cout<<"Input data ke-"<<i<<" : ";

cin>>data[i];

data2[i]=data[i];

}

selection\_sort();

cout<<"Data Setelah di Sorting : "<<endl;

for(int i=1; i<=n; i++)

{

cout<<" "<<data[i];

}

getch();

}

#### Studi Kasus 3: INSERTION SORT

Insertion sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma insertion sort, lakukan langkah-langkah berikut:

* Pelajari cara kerja algoritma insertion sort
* Tentukan T(n) dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

T(n) = {

©(1)

n

aT ( ) + D(n)

b

+ C(n)

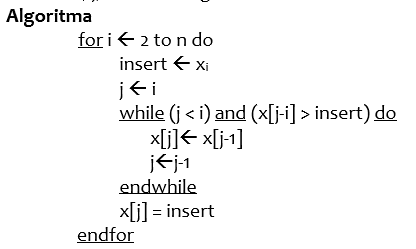
### if n ≤ c

otherwise

Selesaikan persamaan rekurensi T(n) dengan **metode subtitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ

* Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++

**Pengerjaan :**



Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses penggabungan = n

Waktu proses pembagian = n

1. T(n) = cn + cn-c +cn-2c + ..... + 2c +cn <= 2cn2 + cn2

= c((n-1)(n-2)/2) + cn<= 2cn2 + cn2

= c((n2-3n+2)/2) + cn<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+c+cn <= 2cn2 + cn2

=O(n2)

1. T(n) = cn <= cn

= Ω (n)

1. T(n) = (cn + cn2)/n   
   = Θ(n)

/\*

Nama : Hafidh Akhdan N

NPM : 140810180061

Kelas : A

\*/

#include <iostream>

#include <conio.h>

using namespace std;

int data[100],data2[100],n;

void insertion\_sort()

{

int temp,i,j;

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

temp = data[i];

j = i -1;

while(data[j]>temp && j>=0){

data[j+1] = data[j];

j--;

}

data[j+1] = temp;

}

}

int main()

{

cout<<"Masukkan Jumlah Data : "; cin>>n;

cout<<endl;

for(int i=1;i<=n;i++)

{

cout<<"Masukkan data ke-"<<i<<" : ";

cin>>data[i];

data2[i]=data[i];

}

insertion\_sort();

cout<<"\nData Setelah di Sort : "<<endl;

for(int i=1; i<=n; i++)

{

cout<<data[i]<<" ";

}

getch();

}

#### Studi Kasus 4: BUBBLE SORT

Bubble sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma bubble sort, lakukan langkah-langkah berikut:

* Pelajari cara kerja algoritma bubble sort
* Tentukan T(n) dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

T(n) = {

©(1)

n

aT ( ) + D(n)

b

+ C(n)

### if n ≤ c

otherwise

* Selesaikan persamaan rekurensi T(n) dengan **metode master** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
* Lakukan implementasi koding program untuk algoritma bubble sort dengan menggunakan bahasa C++.

**Pengerjaan :**

Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses pembagian = n

Waktu proses penggabungan = n

1. T(n) = cn + cn-c +cn-2c + ..... + 2c +c <= 2cn2 + cn2

= c((n-1)(n-2)/2) + c<= 2cn2 + cn2

= c((n2-3n+2)/2) + c<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+2c <= 2cn2 + cn2

=O(n2)

1. T(n) = cn + cn-c +cn-2c + ..... + 2c +c <= 2cn2 + cn2

= c((n-1)(n-2)/2) + c<= 2cn2 + cn2

= c((n2-3n+2)/2) + c<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+2c <= 2cn2 + cn2

= Ω (n2)

1. T(n) = cn2 + cn2

= Θ(n2)

/\*

Nama : Hafidh Akhdan Najib

NPM : 140810180061

Kelas : A

\*/

#include <iostream>

#include <conio.h>

using namespace std;

int main(){

int arr[100],n,temp;

cout<<"Massukan banyak elemen yang akan diinputkan : ";cin>>n;

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

cout<<"Masukkan Elemen ke-"<<i+1<<" : ";cin>>arr[i];

}

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

for(int j=0;j<(n-1);j++){

if(arr[j]>arr[j+1]){

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

cout << "------------------------------------------------" << endl;

cout<<"\nHasil dari Bubble Sort : "<<endl;

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

cout<<" "<<arr[i];

}

cout << "\n================================================"<<endl;

}