TUGAS 4 PRAKTIKUM

REKURENSI DAN PARADIGMA ALGORITMA DIVIDE & CONQUER

Disusun sebagai salah satu tugas

mata kuliah Analisis Algoritma



Patricia Joanne

140810160065

Dikumpulkan tanggal

27 Maret 2019

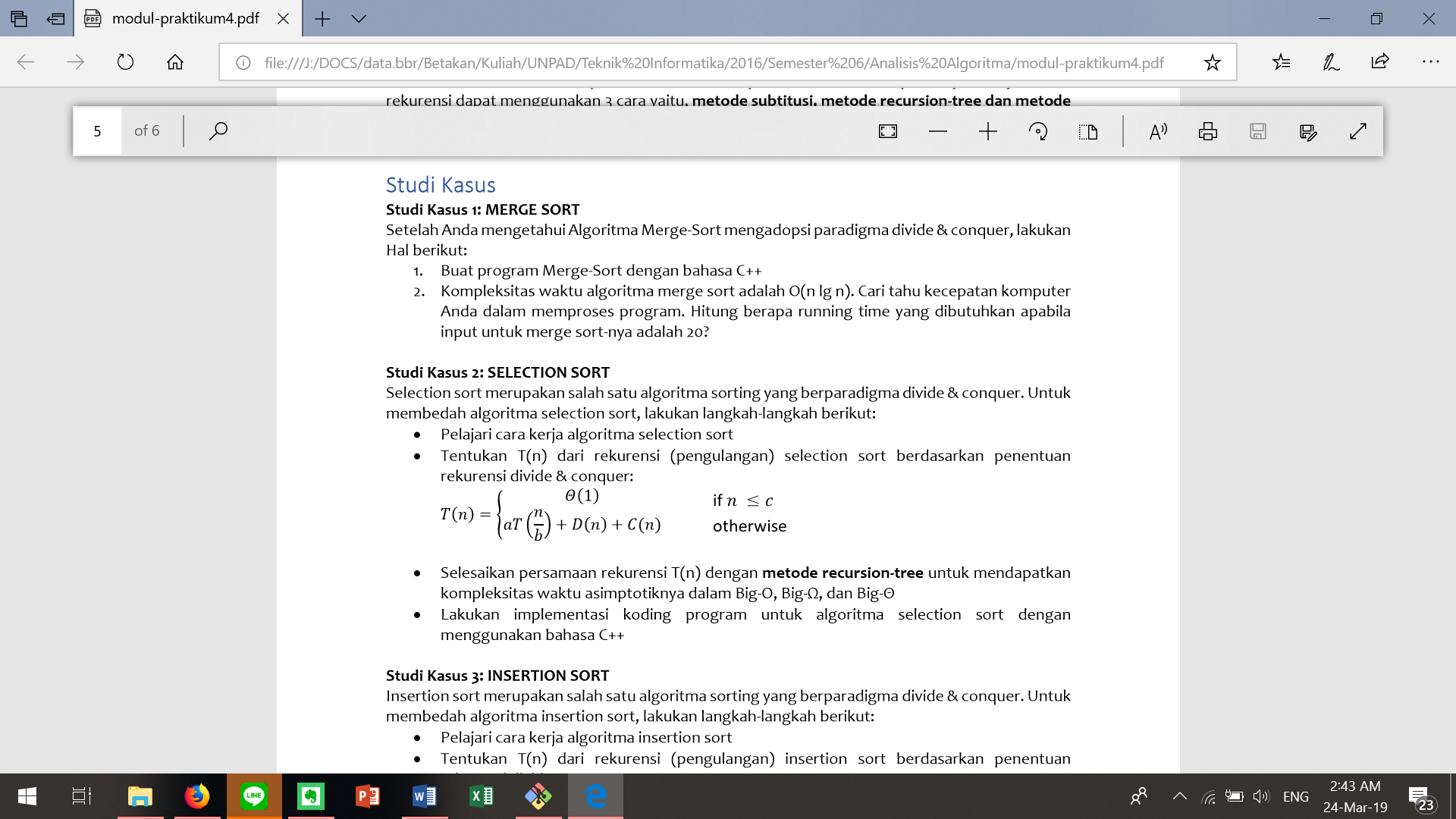
PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2019

**Nomor 1**



Jawab:

* Program: [cpp.sh/8dvp7](http://cpp.sh/8dvp7)

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void merge (int \*a, int low, int high, int mid){

int i, j, k, temp[high-low+1];

i = low;

k = 0;

j = mid + 1;

while (i <= mid && j <= high){

if (a[i] < a[j]){

temp[k] = a[i];

k++;

i++;

}

else {

temp[k] = a[j];

k++;

j++;

}

}

while (i <= mid){

temp[k] = a[i];

k++;

i++;

}

while (j <= high){

temp[k] = a[j];

k++;

j++;

}

for (i = low; i <= high; i++){

a[i] = temp[i-low];

}

}

void mergeSort(int \*a, int low, int high){

int mid;

if (low < high){

mid=(low+high)/2;

mergeSort(a, low, mid);

mergeSort(a, mid+1, high);

merge(a, low, high, mid);

}

}

int main(){

int n, i;

high\_resolution\_clock::time\_point t1 = high\_resolution\_ clock::now();

cout<<"Masukkan jumlah elemen data yang ingin diurutkan: ";

cin>>n;

int arr[n];

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

cout<<"Masukkan elemen ke-"<<i+1<<": ";

cin>>arr[i];

}

mergeSort(arr, 0, n-1);

cout<<"\nArray yang telah diurutkan: ";

for (i = 0; i < n; i++) cout<<" "<<arr[i];

high\_resolution\_clock::time\_point t2 = high\_resolution\_ clock::now();

auto duration = duration\_cast<microseconds>( t2 - t1 ). count();

cout<<endl<<duration<<" microseconds" <<endl;

}

* Output (input n = 20):

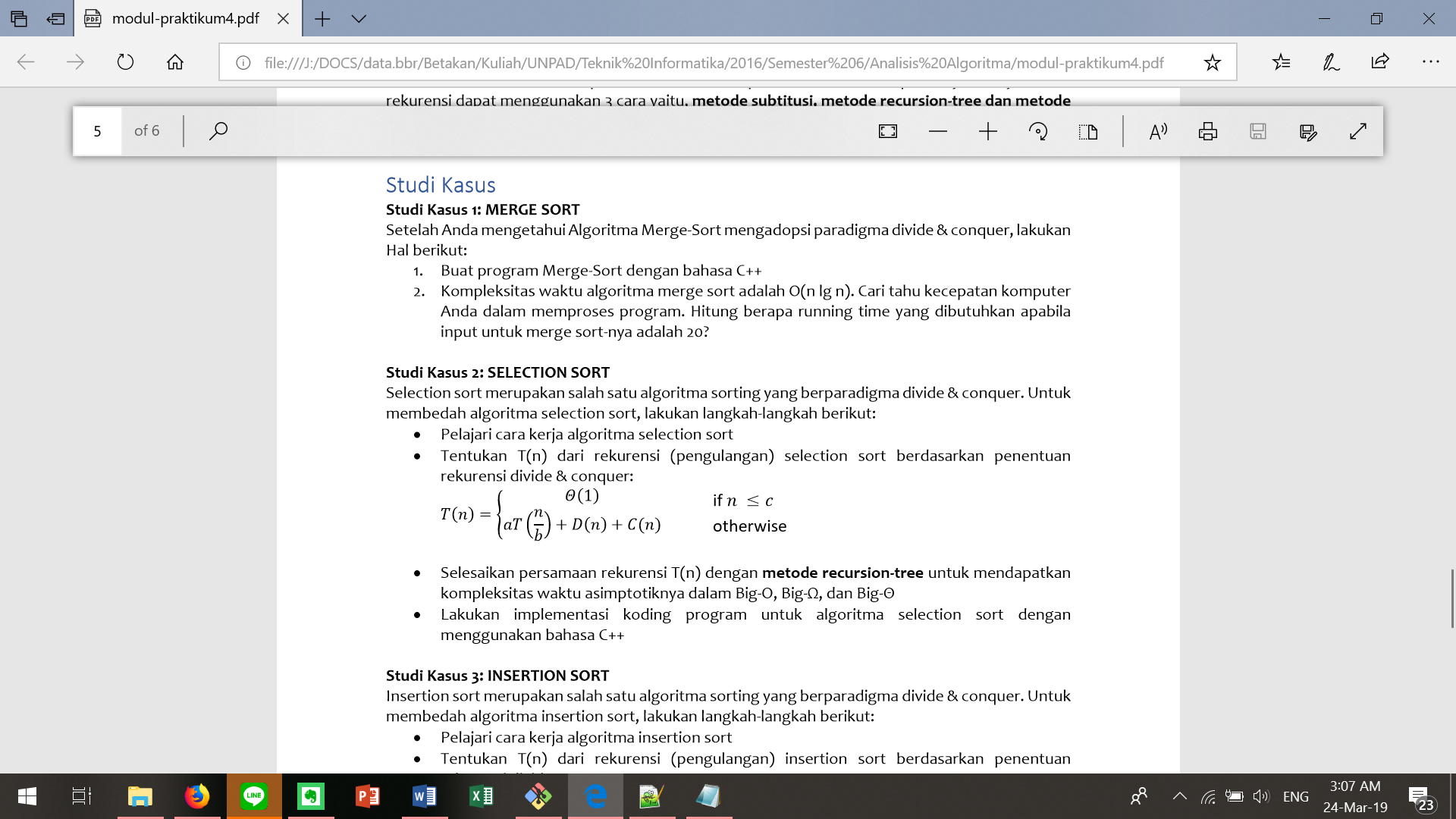


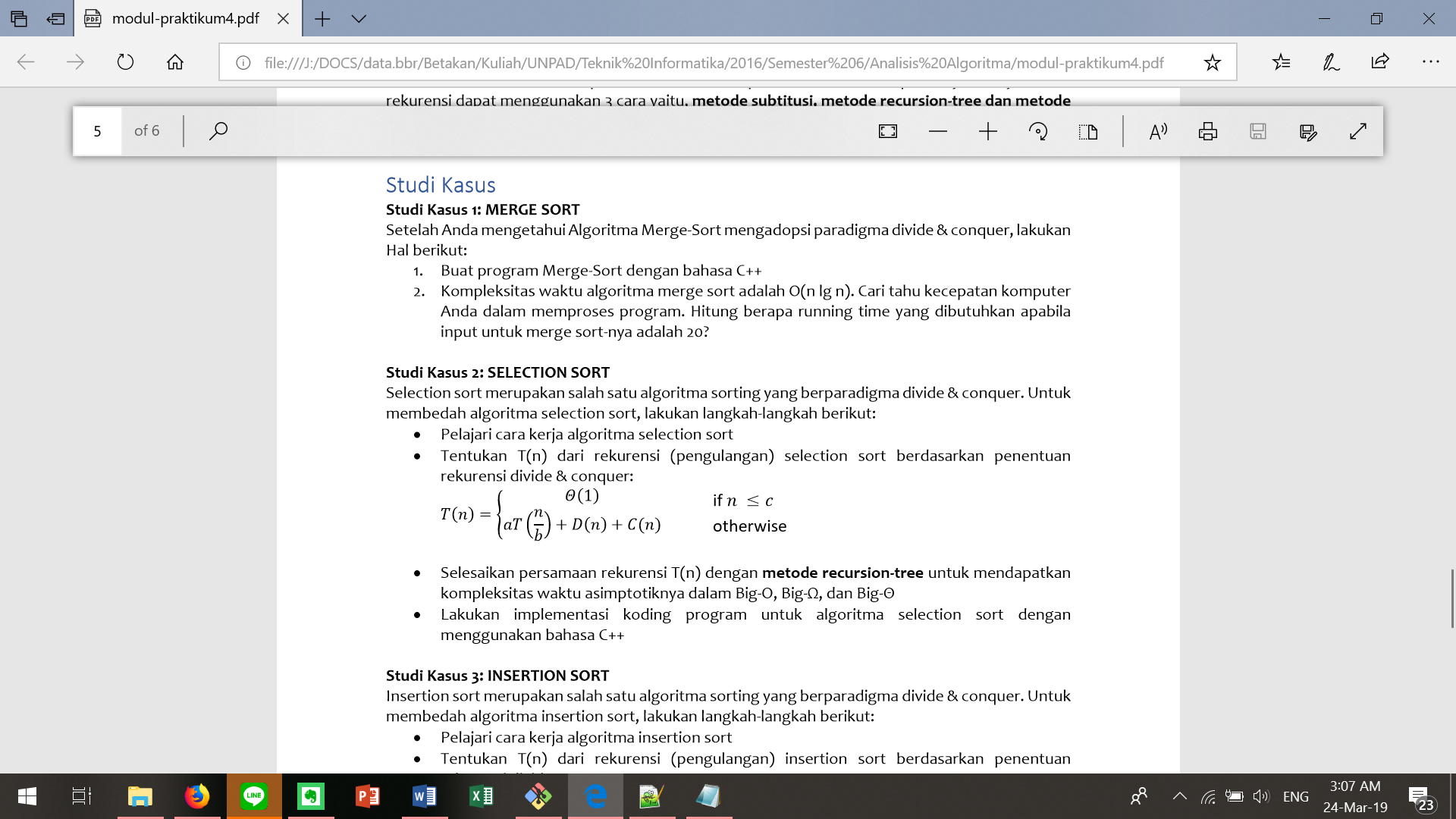
* Kompleksitas waktu:

Durasi waktu yang dibutuhkan untuk 20 input: 34654760 ms = 34.65476 s

Big-O = Big-Ω = Big-θ = n \* log n

**Nomor 2**





Jawab:

* Program: [cpp.sh/4t2kh](http://cpp.sh/4t2kh)

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void selectionSort (int arr[], int n){

int i, j;

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

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

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

arr[i] = arr[i]+arr[j];

arr[j] = arr[i]-arr[j];

arr[i] = arr[i]-arr[j];

}

}

}

}

int main(){

int n, i;

high\_resolution\_clock::time\_point t1 = high\_resolution\_ clock::now();

cout<<"Masukkan jumlah elemen data yang ingin diurutkan: ";

cin>>n;

int arr[n];

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

cout<<"Masukkan elemen ke-"<<i+1<<": ";

cin>>arr[i];

}

selectionSort(arr, n);

cout<<"\nArray yang telah diurutkan: ";

for (i = 0; i < n; i++) cout<<" "<<arr[i];

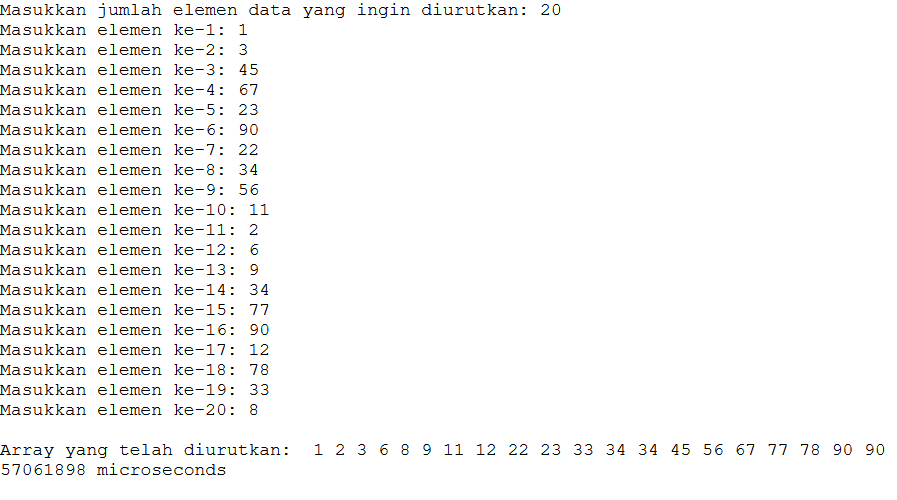
high\_resolution\_clock::time\_point t2 = high\_resolution\_ clock::now();

auto duration = duration\_cast<microseconds>( t2 - t1 ). count();

cout<<endl<<duration<<" microseconds" <<endl;

}

* Output (input n = 20):



* Kompleksitas waktu:

Durasi waktu yang dibutuhkan untuk 20 input: 57061898 ms = 57.061898 s

Big-O = Big-Ω = Big-θ = n2

**Nomor 3**







Jawab:

* Program: [cpp.sh/96tc4](http://cpp.sh/96tc4)

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

struct list {

int data;

list \*next;

};

list\* InsertinList(list \*head, int n){

list \*newnode = new list;

list \*temp = new list;

newnode->data = n;

newnode->next = NULL;

if(head == NULL){

head = newnode;

return head;

}

else {

temp = head;

if(newnode->data < head->data){

newnode->next = head;

head = newnode;

return head;

}

while(temp->next != NULL){

if(newnode->data < (temp->next)->data)

break;

temp=temp->next;

}

newnode->next = temp->next;

temp->next = newnode;

return head;

}

}

int main(){

int n, i, num;

list \*head = new list;

head = NULL;

high\_resolution\_clock::time\_point t1 = high\_resolution\_ clock::now();

cout<<"Masukkan jumlah elemen data yang ingin diurutkan: ";

cin>>n;

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

cout<<"Masukkan elemen ke-"<<i+1<<": ";

cin>>num;

head = InsertinList(head, num);

}

cout<<"\nArray yang telah diurutkan: ";

while(head != NULL){

cout<<" "<<head->data;

head = head->next;

}

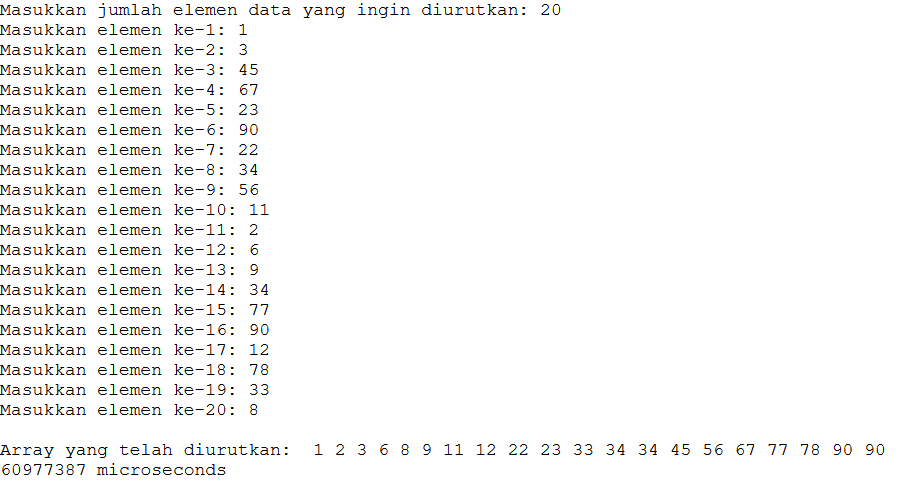
high\_resolution\_clock::time\_point t2 = high\_resolution\_ clock::now();

auto duration = duration\_cast<microseconds>( t2 - t1 ). count();

cout<<endl<<duration<<" microseconds" <<endl;

}

* Output (input n = 20):



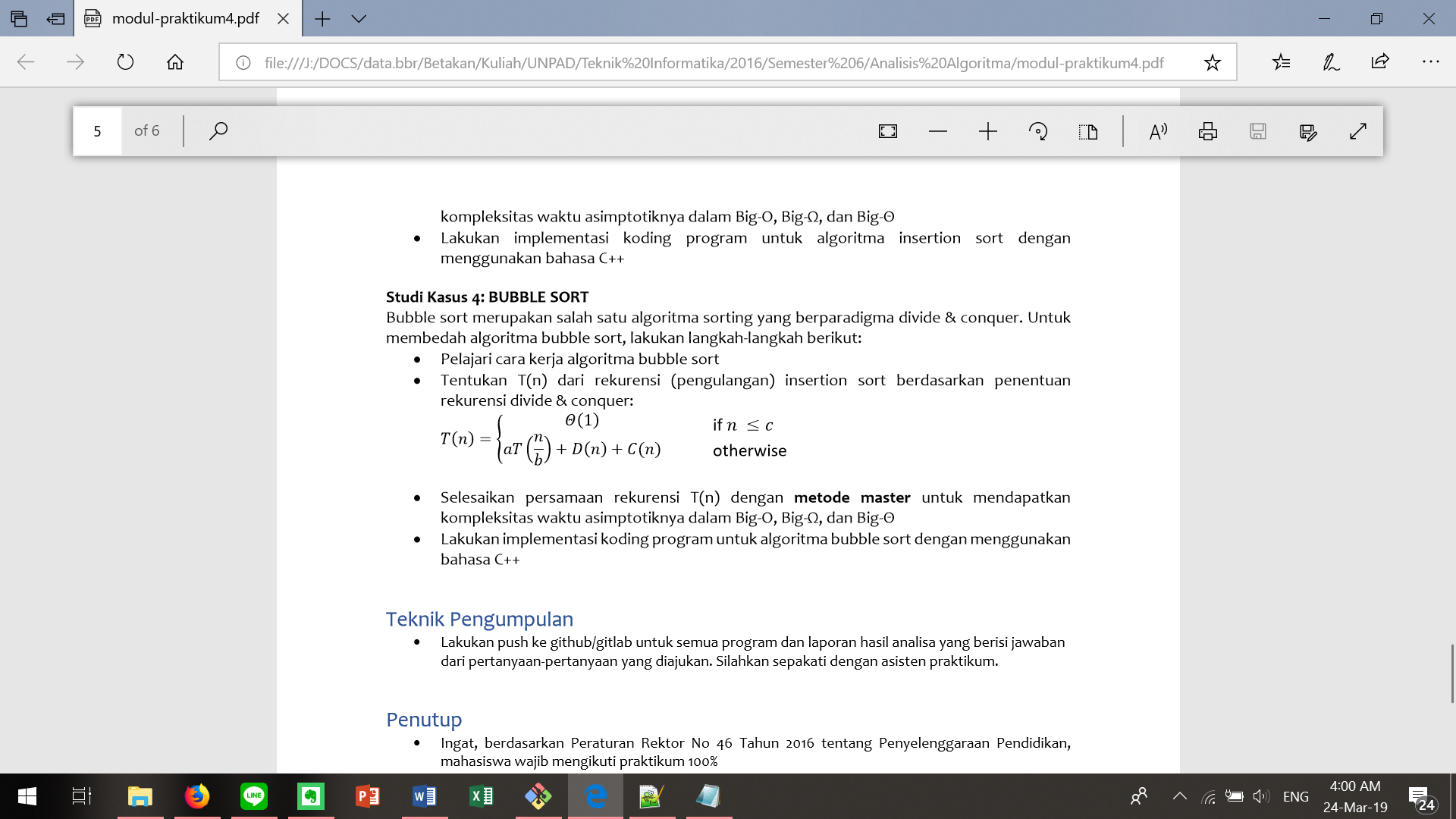
* Kompleksitas waktu:

Durasi waktu yang dibutuhkan untuk 20 input: 60977387 ms = 60.977387 s

Big-O = n

Big-Ω = Big-θ = n2

**Nomor 4**



Jawab:

* Program: [cpp.sh/3ay7x](http://cpp.sh/3ay7x)

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void bubbleSort (int arr[], int n){

int i, j;

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

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

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

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

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

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

}

}

}

}

int main(){

int n, i;

high\_resolution\_clock::time\_point t1 = high\_resolution\_ clock::now();

cout<<"Masukkan jumlah elemen data yang ingin diurutkan: ";

cin>>n;

int arr[n];

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

cout<<"Masukkan elemen ke-"<<i+1<<": ";

cin>>arr[i];

}

bubbleSort(arr, n);

cout<<"\nArray yang telah diurutkan: ";

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

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

}

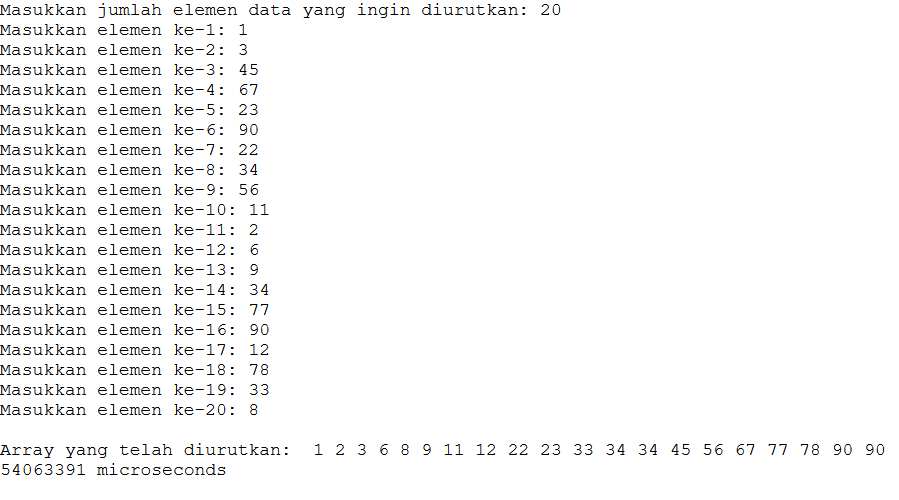
high\_resolution\_clock::time\_point t2 = high\_resolution\_ clock::now();

auto duration = duration\_cast<microseconds>( t2 - t1 ). count();

cout<<endl<<duration<<" microseconds" <<endl;

}

* Output (input n = 20):



* Kompleksitas waktu:

Durasi waktu yang dibutuhkan untuk 20 input: 54063391 ms = 54.063391 s

Big-O = n

Big-Ω = Big-θ = n2