# MAKALAH LAB DESAIN & ANALISIS ALGORITMA



**Nama: Jonathan**

**NIM: 232310014**

**Kelas: TI-23-PA 1**

**Prodi: Teknologi Informasi**

**Institut Bisnis Informatika Kesatuan Bogor**

**JL. Rangga Gading No. 01**

**Kecamatan Bogor Tengah**

**Kota Bogor**

**Kode Pos 16123**

# DAFTAR ISI

# KATA PENGANTAR

Segala puji dan syukur saya penjatkan kepada Tuhan Yang Maha Esa, karena atas penyertaan-Nya makalah Lab Desain dan Analisis Algortima dapat dirangkai dengan baik tanpa kendala. Rasa terimakasih juga penyusun sampaikan kepada pihak-pihak yang telah membantu untuk merealisasikan makalah Lab Desain dan Analisis Algortima. Makalah ini diperkenankan untuk memberi pengetahuan yang lebih lagi mengenai penggunana bahasa pemrograman C++ agar dapat digunakan dengan baik dan seefektif mungkin.

Makalah yang telah dirancang tentunya penyusun sadari masih jauh dari kata sempurna. Maka dari itu diharapkan saran dan kritik dari para pembaca agar modul dapat diperbaharui dan memiliki informasi yang lebih berkualitas dan lebih tepat kedepannya.

# BAB I SORTING

* 1. Sorting

Secara ringkas dan jelas, sorting merupakan algoritma C++ yang memiliki fungsi untuk menyusuk elemen-elemen array yang acak menjadi tersusun secara *ascending* (kecil->besar) atau *descending*  (besar -> kecil).

* 1. Bubble Sort

Bubble Sort adalah salah satu algoritma sorting paling sederhana, bubble sort mengurutkan dengan cara membandingkan elemen array dengan elemen sebelahnya yang lebih besar indeks nya, jika nilai elemen sebelahnya lebih kecil maka akan bertukar tempat. Proses tersebut akan di ulang terus menerus hingga data terurut dengan benar.

#include <iostream>

using namespace std;

* #include <iostream> digunakan untuk input dan output standar
* Using namespace std; digunakan untuk menghindari penulisan std:: agar lebih mudah dalam penulisan kode

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

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

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

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

int temp = arr[j];

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

arr[j + 1] = temp;

}

}

}

}

Jika nilai array pertama lebih kecil dari array di sebelahnya maka akan menukar tempat kedua array tersebut dengan menggunakan *int temp* sebagai wadah sementara untuk memindahkan array tersebut

int main() {

int n;

cout << "Masukkan jumlah elemen dalam array: ";

cin >> n;

int arr[n];

cout << "Masukkan elemen-elemen array:\n";

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

cout << "Elemen ke-" << i + 1 << ": ";

cin >> arr[i];

}

bubbleSort(arr, n);

cout << "Array setelah diurutkan menggunakan Bubble Sort:\n";

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

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

}

cout << endl;

return 0;

}

* Dalam int main terdapat kode dasar seperti masukan jumlah elemen array yang ingin digunakan dan juga mencetak kembali nilai array yang sudah dimasukan setelah mengalami proses bubble sort.
  1. Insertion Sort

Insertion Sort merupakan metode sorting dengan cara membandingkan nilai array dengan nilai disebelahnya, jika nilai disebelahnya lebih kecil maka nilai array kedua akan berpindah ke tempat nilai array pertama sehingga bukan bertukar tempat namun menggeser posisi array ke kanan satu kali.

#include <iostream>

using namespace std;

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

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

int key = arr[i];

int j = i - 1;

// Geser elemen-elemen yang lebih besar dari key ke kanan

while (j >= 0 && arr[j] > key) {

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

j--;

}

// Tempatkan key pada posisi yang sesuai

arr[j + 1] = key;

}

}

int main() {

int n;

cout << "Masukkan jumlah elemen dalam array: ";

cin >> n;

int arr[n];

cout << "Masukkan elemen-elemen array:\n";

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

cout << "Elemen ke-" << i + 1 << ": ";

cin >> arr[i];

}

insertionSort(arr, n);

cout << "Array setelah diurutkan menggunakan Insertion Sort:\n";

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

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

}

cout << endl;

return 0;

}

* 1. Selection Sort

# BAB II SORTING (LANJUTAN)

* 1. Quick Sort
  2. Merge Sort

# BAB III CLASS

* 1. Class
  2. Class Setter dan Getter

# BAB IV KONSTRUKTOR DAN DESTRUKTOR

* 1. Konstruktor
  2. Destruktor

# BAB V CLASS (OOP)

* 1. Class (OOP)
  2. Enkapsulasi
  3. Polimorfisme

# BAB VI SEARCHING

* 1. Searching
  2. Sequential Search
  3. Binary Search