

**LAPORAN PRAKTIKUM**  
**MODUL 2**  
**Pengenalan Bahasa C++**



**Nama :**

Candra Dinata (2311104061)

**Kelas :**

S1SE 07 02

**Dosen :**

Wahyu Andi Saputra

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024**

## **I. TUJUAN**

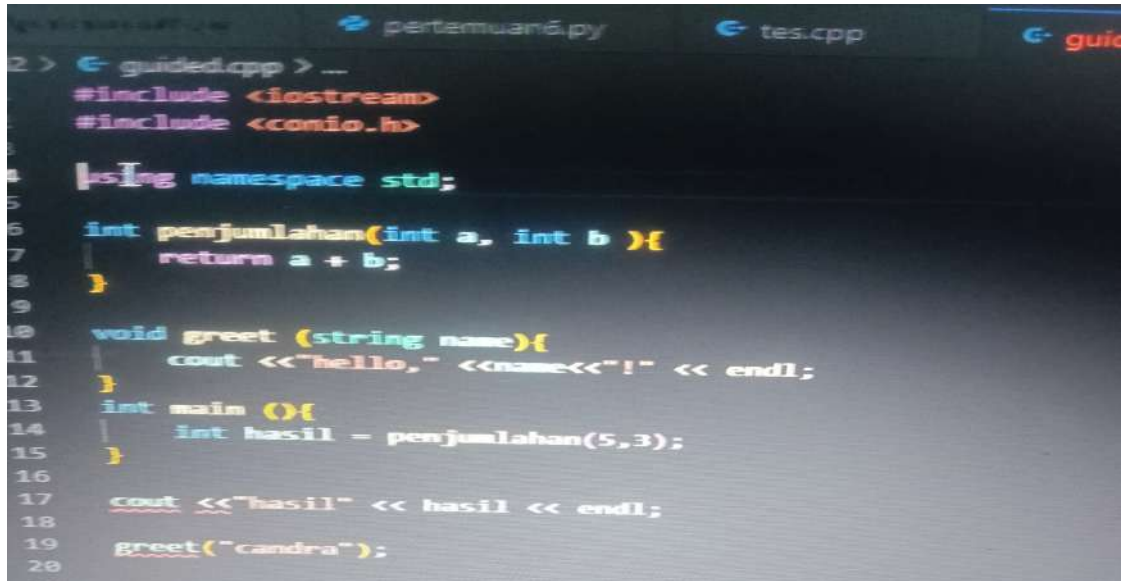
Tujuan dari pengenalan C++ dalam laporan praktikum adalah untuk membantu mahasiswa memahami dasar-dasar pemrograman, seperti sintaks, tipe data, operator, dan struktur kontrol, serta menguasai penggunaan fungsi untuk modularitas program. Selain itu, mahasiswa juga diperkenalkan dengan konsep Object-Oriented Programming (OOP) seperti kelas, objek, pewarisan, dan enkapsulasi. Melalui latihan ini, mahasiswa diharapkan mampu menerapkan konsep pemrograman untuk menyelesaikan masalah dan memahami cara kerja kompilator C++ dalam mengubah kode sumber menjadi program yang dapat dijalankan.

## **II. LANDASAN TEORI**

C++ adalah bahasa pemrograman yang dikembangkan sebagai pengembangan dari bahasa C dengan menambahkan fitur-fitur pemrograman berorientasi objek (Object-Oriented Programming, OOP). Dalam C++, konsep OOP seperti enkapsulasi, pewarisan, dan polimorfisme menjadi dasar untuk membuat program yang lebih modular, terstruktur, dan dapat digunakan kembali. Bahasa ini mendukung penggunaan fungsi, manipulasi memori secara langsung melalui pointer, serta memiliki kemampuan untuk menangani berbagai tipe data dasar dan struktural. C++ juga mendukung konsep pemrograman prosedural dan modular, yang memudahkan pengembangan program yang kompleks. Kompilasi program C++ dilakukan melalui kompilator yang menerjemahkan kode sumber menjadi file eksekusi yang bisa dijalankan oleh komputer.

### III. GUIDED

1.



```
2 > E guided.cpp > ...
#include <iostream>
#include <conio.h>

using namespace std;

int penjumlahan(int a, int b ){
    return a + b;
}

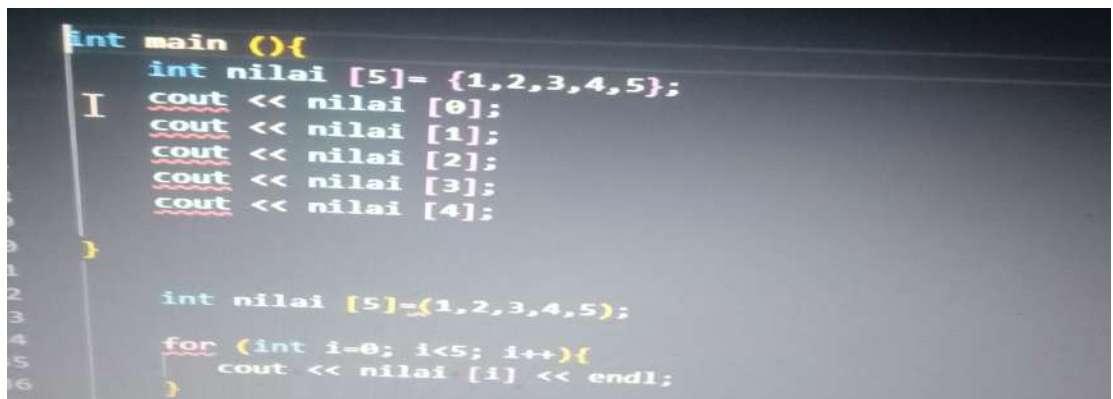
void greet (string name){
    cout <<"hello," <<name<<"!" << endl;
}

int main (){
    int hasil = penjumlahan(5,3);

    cout <<"hasil" << hasil << endl;
    greet("candra");
}
```

Kode ini menggunakan namespace `std` untuk menghindari penulisan `std::` sebelum elemen standar library seperti `cout`. Fungsi `penjumlahan(int a, int b)` menerima dua parameter dan mengembalikan hasil penjumlahan keduanya, sedangkan fungsi `greet(string name)` menampilkan pesan "hello, [name]!" menggunakan `cout`. Dalam fungsi utama `main()`, hasil dari penjumlahan 5 dan 3 disimpan dalam variabel `hasil`, dan fungsi `greet("candra")` dipanggil untuk mencetak "hello, candra!".

2.



```
int main (){
    int nilai [5]= {1,2,3,4,5};

    cout << nilai [0];
    cout << nilai [1];
    cout << nilai [2];
    cout << nilai [3];
    cout << nilai [4];

}

int nilai [5]={1,2,3,4,5};

for (int i=0; i<5; i++){
    cout << nilai [i] << endl;
}
```

Kode ini mendefinisikan sebuah array nilai dengan 5 elemen (1, 2, 3, 4, 5). Pada bagian pertama, elemen-elemen array ditampilkan secara manual dengan menggunakan lima kali pemanggilan cout untuk setiap indeks array (nilai[0] hingga nilai[4]).

3.

```
8
39      array 2 dimensi
40      int nilai[3][4] = {
41          {1,2,3,4},
42          {5,6,7,8},
43          {9,10,11,12}
44      };
45
46      for (int i=0; i<3; i++){
47          for (int j=0; j<4; j++){
48              cout << nilai [i][j] << " ";
49          }
50      }
51      cout endl;
52 }
```

Kode ini mendefinisikan sebuah array 2 dimensi `nilai` berukuran 3x4 yang menyimpan elemen-elemen dari 1 hingga 12 dalam bentuk matriks. Setiap baris dalam array terdiri dari 4 elemen, yaitu {1, 2, 3, 4}, {5, 6, 7, 8}, dan {9, 10, 11, 12}. Kode ini menggunakan dua perulangan bersarang (`for`), di mana loop luar (dengan variabel `i`) mengiterasi baris, dan loop dalam (dengan variabel `j`) mengiterasi kolom dari setiap baris. Di dalam loop dalam, setiap elemen array `nilai[i][j]` dicetak diikuti dengan spasi, dan setelah setiap baris selesai, `cout << endl;` digunakan untuk mencetak baris baru, sehingga menghasilkan tampilan array dalam bentuk matriks 3x4.

4.

```
55      pointer
56      int x,y;
57      int px;
58      x=87;
59      px = &x;
60      y = *px;
61
62      cout << "alamat x=" << &x << endl;
63      cout << "isi px= " << px << endl;
64      cout << "isi X=" << x << endl;
65      cout << "nilai yang ditunjuk px=" << *px << endl;
66      cout << "nilai y=" << y << endl;
```

Variabel x diinisialisasi dengan nilai 87, dan px adalah pointer yang menyimpan alamat memori dari variabel x dengan menggunakan operator &. Variabel y kemudian mendapatkan nilai yang ditunjuk oleh px, yaitu nilai dari x, menggunakan dereferensi pointer (\*px). Beberapa perintah cout digunakan untuk menampilkan hasil: alamat memori dari x, isi dari pointer px (alamat x), nilai x, nilai yang ditunjuk oleh px (nilai x), dan nilai y, yang sama dengan nilai x. Program ini menunjukkan cara kerja pointer, bagaimana pointer menyimpan alamat memori dan mengakses nilai dari alamat tersebut.

## IV. UNGUIDED

1.

```
1  #include <iostream>
2  #include <vector>
3
4  int main() {
5      const int size = 10; // Ukuran array
6      std::vector<int> data(size); // Membuat vector untuk menyimpan data
7
8      // Input data dari pengguna
9      std::cout << "Masukkan " << size << " angka (pisahkan dengan spasi): ";
10     for (int i = 0; i < size; i++) {
11         std::cin >> data[i];
12     }
13
14     std::cout << "Data array: ";
15     for (int num : data) {
16         std::cout << num << " ";
17     }
18     std::cout << std::endl;
19
20     // Menyimpan angka genap dan ganjil
21     std::vector<int> genap;
22     std::vector<int> ganjil;
23
24     // Memisahkan angka genap dan ganjil
25     for (int num : data) {
26         if (num % 2 == 0) {
27             genap.push_back(num);
28         } else {
29             ganjil.push_back(num);
30         }
31     }
```

```
PS C:\Users\Candra Dinata\pertemuan1> cd 'c:\Users\Candra Dinata\pertemuan1\mod2\'
PS C:\Users\Candra Dinata\pertemuan1\mod2\output> & .\'unguided.exe'
Masukkan 10 angka (pisahkan dengan spasi): 1 2 3 4 5 6 7 8 9 10
Data array: 1 2 3 4 5 6 7 8 9 10
Nomor genap: 2 4 6 8 10
Nomor ganjil: 1 3 5 7 9
PS C:\Users\Candra Dinata\pertemuan1\mod2\output>
```

Kode ini menggunakan `std::vector` untuk menyimpan data dan memisahkan angka genap dan ganjil. Pertama, sebuah vektor `data` dengan ukuran 10 dibuat untuk menyimpan input dari pengguna, yang dimasukkan melalui `std::cin`. Setelah semua angka dimasukkan, program menampilkan kembali isi dari vektor tersebut. Selanjutnya, dua vektor terpisah (`genap` dan `ganjil`) dibuat untuk menyimpan angka genap dan ganjil. Program kemudian melakukan iterasi pada vektor `data`, memeriksa setiap angka dengan operasi modulus (`%`), dan memisahkannya ke dalam vektor genap atau ganjil. Terakhir, program menampilkan semua angka genap dan ganjil yang disimpan.

2.

```

// Input ukuran array dari pengguna
std::cout << "Masukkan ukuran dimensi pertama (x): ";
std::cin >> x;
std::cout << "Masukkan ukuran dimensi kedua (y): ";
std::cin >> y;
std::cout << "Masukkan ukuran dimensi ketiga (z): ";
std::cin >> z;

// Membuat array tiga dimensi statis
int arr[x][y][z];

// Input elemen array
std::cout << "Masukkan elemen array tiga dimensi:" << std::endl;
for (int i = 0; i < x; i++) {
    for (int j = 0; j < y; j++) {
        for (int k = 0; k < z; k++) {
            std::cout << "Elemen arr[" << i << "][" << j << "][" << k << "]: ";
            std::cin >> arr[i][j][k];
        }
    }
}

// Menampilkan elemen array
std::cout << "Elemen array tiga dimensi:" << std::endl;
for (int i = 0; i < x; i++) {
    for (int j = 0; j < y; j++) {
        for (int k = 0; k < z; k++) {
            std::cout << "arr[" << i << "][" << j << "][" << k << "] = " << arr[i][j][k] << std::endl;
        }
    }
}

```

```

Masukkan ukuran dimensi pertama (x): 6
Masukkan ukuran dimensi kedua (y): 8
Masukkan ukuran dimensi ketiga (z): 4
Masukkan elemen array tiga dimensi:
Elemen arr[0][0][0]: 7
Elemen arr[0][0][1]: 5
Elemen arr[0][0][2]: 4
Elemen arr[0][0][3]: 7
Elemen arr[0][1][0]: 7
Elemen arr[0][1][1]: 7
Elemen arr[0][1][2]: 7

```

Kode ini meminta input dari pengguna untuk tiga ukuran dimensi (x, y, z) untuk membuat sebuah array tiga dimensi statis. Array ini kemudian diisi dengan nilai-nilai yang dimasukkan pengguna melalui konsol. Setelah array diisi, program akan menampilkan kembali elemen-elemen array tersebut dengan menunjukkan indeksnya masing-masing. Program ini mendemonstrasikan cara kerja array tiga dimensi di C++, tetapi terdapat kesalahan karena ukuran array seharusnya ditentukan secara dinamis saat runtime, sementara kode ini mencoba membuat array statis dengan ukuran yang ditentukan oleh input pengguna, yang tidak diperbolehkan di C++.

3.

```

#include <iostream>
#include <vector>

int main() {
    int n;

    // Input ukuran array
    std::cout << "Masukkan jumlah elemen array: ";
    std::cin >> n;

    std::vector<int> arr(n); // Membuat vector untuk menyimpan data

    // Input elemen array
    std::cout << "Masukkan " << n << " elemen array:" << std::endl;
    for (int i = 0; i < n; i++) {
        std::cout << "Elemen [" << i << "]: ";
        std::cin >> arr[i];
    }

    // Mencari nilai maksimum, minimum, dan rata-rata
    int max = arr[0];
    int min = arr[0];
    double sum = 0;

    for (int i = 0; i < n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
        if (arr[i] < min) {
            min = arr[i];
        }
        sum += arr[i];
    }
}

```



```
PS C:\Users\Candra Dinata\pertemuan  
Masukkan jumlah elemen array: 3  
Masukkan 3 elemen array:  
Elemen [1]: 5  
Elemen [2]: 7  
Elemen [3]: 9  
Nilai Maksimum: 9  
Nilai Minimum: 5  
Nilai Rata-Rata: 7  
PS C:\Users\Candra Dinata\pertemuan  
Compile  Debug
```

Program ini meminta pengguna untuk memasukkan jumlah elemen dari sebuah array dan kemudian menggunakan `std::vector` untuk menyimpan elemen-elemen tersebut. Setelah pengguna menginput nilai-nilai elemen array, program menghitung nilai maksimum, minimum, dan rata-rata dari elemen-elemen tersebut. Dalam proses ini, program menginisialisasi `max` dan `min` dengan elemen pertama array, kemudian melakukan iterasi untuk memperbarui nilai maksimum dan minimum serta menjumlahkan semua elemen. Akhirnya, program menampilkan hasil perhitungan nilai maksimum, minimum, dan rata-rata kepada pengguna. Kode ini menunjukkan penggunaan `std::vector` yang lebih fleksibel dibandingkan array statis, serta operasi dasar dengan struktur data.

## V. KESIMPULAN

Dalam praktikum ini, kita telah mempelajari dasar-dasar pemrograman C++, termasuk cara mengelola data menggunakan array. Melalui contoh kode, kita melihat bagaimana pengguna dapat berinteraksi dengan program melalui input, serta bagaimana data yang dimasukkan dapat diproses untuk menghasilkan informasi yang berguna, seperti nilai maksimum, minimum, dan rata-rata. Kami juga memahami pentingnya memilih struktur data yang tepat untuk situasi tertentu, di mana penggunaan array dinamis menawarkan fleksibilitas dalam pengelolaan ukuran array.