

**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL I  
PENGENALAN CODE BLOCKS**



**Disusun Oleh :**

Ulung Putra Sadewo

103122400013

**s**

**Dosen**

Diah Septiani S.Kom M.Cs

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

## A. Dasar Teori

C++ adalah bahasa pemrograman *compiled* dan *statically-typed* yang dikenal karena performa dan kontrolnya yang tinggi. Dikembangkan sebagai ekstensi dari C, C++ mendukung berbagai paradigma, terutama Pemrograman Berorientasi Objek (OOP), yang memungkinkan pengorganisasian kode yang kompleks [1].

Karena efisiensinya, C++ menjadi standar industri untuk aplikasi yang menuntut kecepatan, seperti *game engine* dan komputasi performa tinggi [2].

## B. Guided (berisi screenshot source code & output program disertai penjelasannya)

### Guided 1

```
#include <iostream>

using namespace std;

int main() {
    cout<<"HELLO WORLD!"<<endl;
    return 0;
}
```

### Screenshots Output

A screenshot of a terminal window with a dark background. The terminal shows the command prompt 'PS D:\Semester 3\Struktur Data\praktikum>' followed by a long command to run a C++ program using a debugger. The output of the program is 'HELLO WORLD!'. The terminal also shows the user's name 'Ulung Putra Sadewo' and a session ID '(103122400013)'.

### Deskripsi:

Program "Hello, World!" adalah tes fundamental untuk memvalidasi bahwa *compiler* dan lingkungan pengembangan berfungsi dengan benar. Di C++, ini dicapai dengan menyertakan pustaka `<iostream>`, yang menyediakan `std::cout` untuk mengirim *output* ke konsol. Ini adalah langkah pertama dalam memahami konsep *stream I/O* (Input/Output) [3].

## Guided 2

```
#include <iostream>
#include <conio.h>

using namespace std;

int main(){
    int x,y;
    int *px;
    x = 87;
    px = &x;
    y = *px;

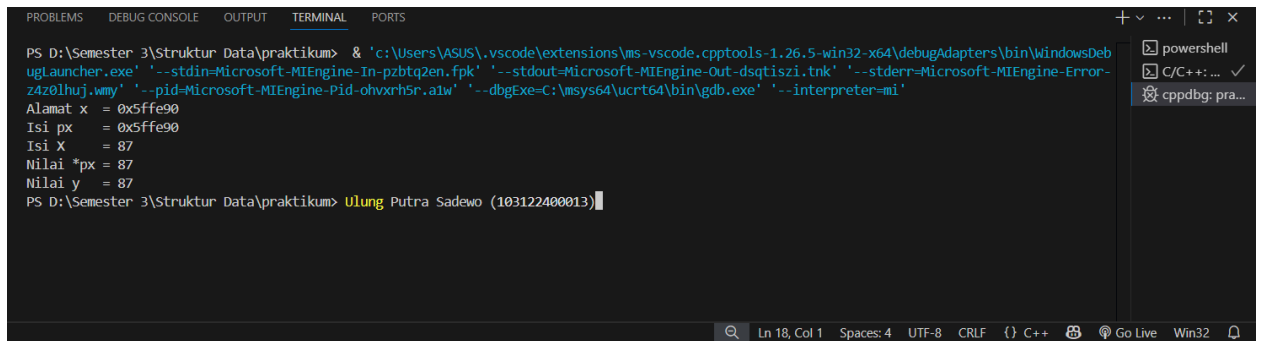
    cout<< "Alamat x = "<< &x <<endl;
    cout<< "Isi px  = "<< px <<endl;
    cout<< "Isi X   = "<< x <<endl;
    cout<< "Nilai *px = "<< *px <<endl;
    cout<< "Nilai y  = "<< y <<endl;

    getch();
    return 0;
}
```

### Deskripsi:

Pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Fitur ini memungkinkan manipulasi memori secara langsung, yang menjadi kunci efisiensi C++. Penguasaan pointer sangat penting untuk memahami struktur data dinamis dan optimisasi program tingkat rendah [4].

Screenshots output :



```
PS D:\Semester 3\Struktur Data\praktikum> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.26.5-win32-x64\debugAdapters\bin\windowsDeb
uglauncher.exe' '--stdin=Microsoft-MIEngine-In-pzbtq2en.fpk' '--stdout=Microsoft-MIEngine-Out-dsqtiszi.tnk' '--stderr=Microsoft-MIEngine-Error-
z4z0lhuj.wmy' '--pid=Microsoft-MIEngine-Pid-ohvxrh5r.alw' '--dbgExe=c:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Alamat x = 0x5ffe90
Isi px = 0x5ffe90
Isi X = 87
Nilai *px = 87
Nilai y = 87
PS D:\Semester 3\Struktur Data\praktikum> Ulung Putra Sadewo (103122400013)
```

Guided 3 :

```
/* #include <iostream>

#include <conio.h>

#define MAX 5

using namespace std;

int main(){
    int i, j;
    float nilai[MAX];
    static int nilai_tahun[MAX][MAX] = {
        {0, 2, 2, 0, 0},
        {0, 1, 1, 1, 0},
        {0, 3, 3, 3, 0},
        {4, 4, 0, 0, 4},
        {5, 0, 0, 0, 5}

};
```

```

    //input data array 1 dimensi
for (i = 0; i < MAX; i++) {
    cout << "Masukan nilai ke-" << i + 1 << ": ";
    cin >> nilai[i];
}

// menampilkan array 1 dimensi
cout << "\nData nilai siswa:\n";
for (i = 0; i < MAX; i++){
    cout << "Nilai ke-" << i + 1 << " = " << nilai [i] << endl;
}

//Menampilkan isi array 2 dimensi
cout << "\nNilai Tahunan:\n";
for (i = 0; i < MAX; i++){
    for (j = 0; j < MAX; j++){
        cout << nilai_tahun[i][j] << " ";
    }
    cout << endl;
}

getch();
return 0;

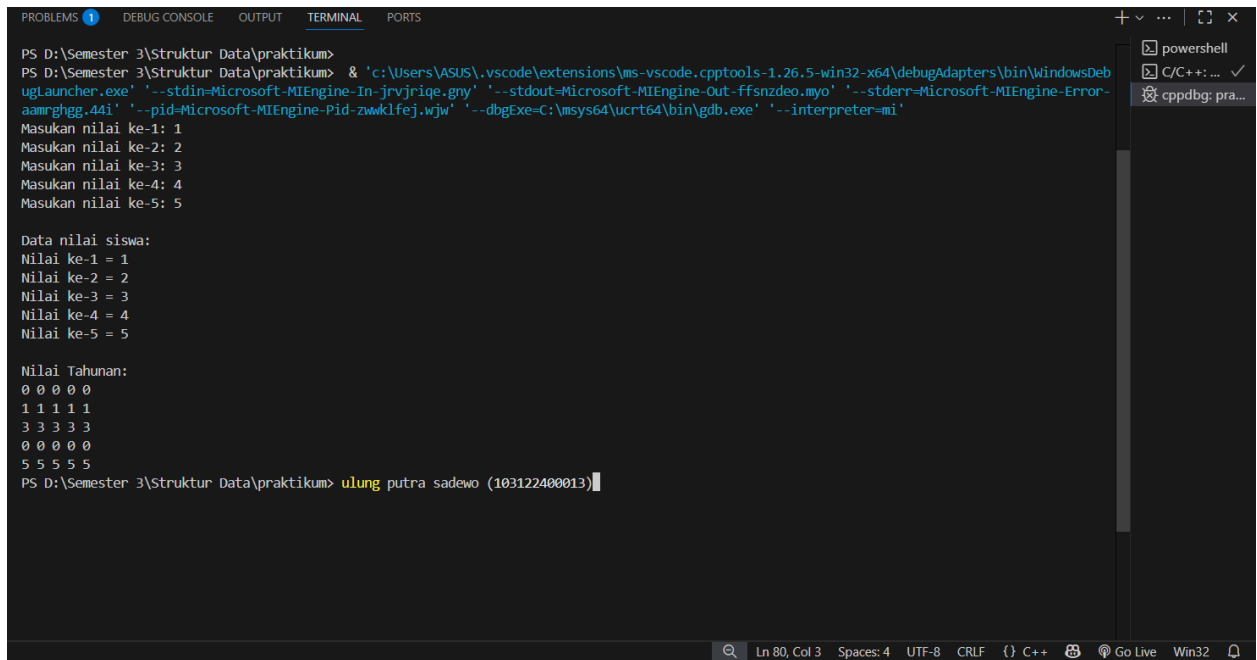
} */

```

Deskripsi :

Array adalah struktur data dasar yang menyimpan kumpulan elemen bertipe sama dalam blok memori yang berdekatan. Strukturnya memungkinkan akses data yang sangat cepat melalui indeks (misalnya, data[0]), sebuah operasi berkecepatan konstan  $O(1)$ . Array menjadi fondasi untuk berbagai algoritma dan struktur data yang lebih kompleks. [5]

Screenshots output :



```
PS D:\Semester 3\Struktur Data\praktikum>
PS D:\Semester 3\Struktur Data\praktikum> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.26.5-win32-x64\debugAdapters\bin\windowsDeb
ugLauncher.exe' '--stdin=Microsoft-MIEngine-In-jrvjrjrqe.gny' '--stdout=Microsoft-MIEngine-Out-ffsnzdeo.myo' '--stderr=Microsoft-MIEngine-Error-
aamrghgg.44i' '--pid=Microsoft-MIEngine-Pid-zmwklfej.wjw' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Masukan nilai ke-1: 1
Masukan nilai ke-2: 2
Masukan nilai ke-3: 3
Masukan nilai ke-4: 4
Masukan nilai ke-5: 5

Data nilai siswa:
Nilai ke-1 = 1
Nilai ke-2 = 2
Nilai ke-3 = 3
Nilai ke-4 = 4
Nilai ke-5 = 5

Nilai Tahunan:
0 0 0 0
1 1 1 1
3 3 3 3
0 0 0 0
5 5 5 5
PS D:\Semester 3\Struktur Data\praktikum> ulung putra sadewo (103122400013)
```

F. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided 1

```
TYPE nilaiSTD : RECORD
    clo1      : real
    clo2      : real
    clo3      : real
    clo4      : real
    nilaiAkhir : real
    indeks    : string
END RECORD
```

Screenshots Output

Deskripsi:

Langkah ini mencakup pembuatan TYPE nilaiSTD (desain/blueprint) dan deklarasi variabelnya (instansiasi/objek nyata). Ini adalah praktik abstraksi data fundamental, di mana kita memodelkan entitas dunia nyata ke dalam struktur yang dipahami program. Tujuannya adalah untuk mengelola kompleksitas data secara efektif. [6]

Unguided 2 :

```
VAR nilaiMhs1 : nilaiSTD
VAR nilaiMhs2 : nilaiSTD
```

## Deskripsi

Sama Halnya dengan Unguided 1 bahwa Langkah ini mencakup pembuatan TYPE nilaiSTD (desain/blueprint) dan deklarasi variabelnya (instansiasi/objek nyata). Ini adalah praktik abstraksi data fundamental, di mana kita memodelkan entitas dunia nyata ke dalam struktur yang dipahami program. Tujuannya adalah untuk mengelola kompleksitas data secara efektif. [6]

## Unguided 3

```
ALGORITHM InputNilaiCLO

// DEKLARASI: Mendefinisikan semua variabel yang akan digunakan.
VAR clo1, clo2, clo3, clo4 : real

// ALGORITMA: Langkah-langkah utama program.
OUTPUT "Masukkan nilai CLO-1 : "
INPUT clo1

OUTPUT "Masukkan nilai CLO-2 : "
INPUT clo2

OUTPUT "Masukkan nilai CLO-3 : "
INPUT clo3

OUTPUT "Masukkan nilai CLO-4 : "
INPUT clo4

// Akhir dari program. Nilai sudah tersimpan dalam variabel.
END ALGORITHM
```

## Deskripsi

Pembuatan program untuk meminta inputan CLO 1-4 adalah definisi dari antarmuka pengguna (user interface) dan lapisan input data. Dalam desain sistem, memisahkan logika untuk mengumpulkan data dari logika untuk memproses data adalah langkah penting untuk menciptakan program yang terstruktur dan modular. [7]

#### Unguided 4

```
FUNCTION GabungNilaiSTD (c1, c2, c3, c4 : real) -> nilaiSTD
```

#### Deskripsi

Pembuatan judul FUNCTION GabungNilaiSTD adalah contoh desain modular. Fungsi ini memiliki satu tanggung jawab spesifik: "membangun" atau merakit sebuah record nilaiSTD dari data mentah. Ini menyembunyikan detail implementasi dari bagian program lain, sebuah konsep yang disebut enkapsulasi [7].

#### Unguided 5

```
FUNCTION GabungNilaiSTD (c1, c2, c3, c4 : real) -> nilaiSTD
```

```
// DEKLARASI LOKAL
```

```
// Membuat variabel sementara 'hasil' yang bertipe nilaiSTD untuk menampung data
```

```
VAR hasil : nilaiSTD
```

```
// ALGORITMA
```

```
// Memasukkan nilai dari parameter input (c1, c2, c3, c4)
```

```
// ke dalam field yang sesuai pada variabel 'hasil'.
```

```
hasil.clo1 <- c1
```

```
hasil.clo2 <- c2
```

```
hasil.clo3 <- c3
```

```
hasil.clo4 <- c4
```

```
// Menginisialisasi field lain yang belum bisa dihitung
```

```
// dengan nilai default (kosong atau nol).
```

```
hasil.nilaiAkhir <- 0.0
```

```
hasil.indeks <- ""
```

```
// Mengembalikan (return) variabel 'hasil' yang sudah terisi
```

```
// sebagai output dari function.
```

```
RETURN hasil
```



```
END FUNCTION
```

### Deskripsi

Pembuatan isi FUNCTION GabungNilaiSTD adalah contoh desain modular. Fungsi ini memiliki satu tanggung jawab spesifik: "membangun" atau merakit sebuah record nilaiSTD dari data mentah. Ini menyembunyikan detail implementasi dari bagian program lain, sebuah konsep yang disebut enkapsulasi [7].

### Unguided 6

```
FUNCTION HitungNilaiAkhir (dataMhs : nilaiSTD) -> real

// DEKLARASI LOKAL
// Variabel untuk menyimpan hasil perhitungan nilai akhir.
VAR nilaiFinal : real

// ALGORITMA
// Hitung nilai akhir dengan mengalikan setiap nilai CLO
// dengan bobot persentasenya, lalu menjumlahkan semuanya.
nilaiFinal <- (0.30 * dataMhs.clo1) + (0.30 * dataMhs.clo2) + (0.20 * dataMhs.clo3)
+ (0.20 * dataMhs.clo4)

// Kembalikan hasil perhitungan.
RETURN nilaiFinal

END FUNCTION
```

### Deskripsi

FUNCTION HitungNilaiAkhir secara khusus mengimplementasikan logika bisnis (rumus nilai akhir). Mengisolasi aturan-aturan bisnis seperti ini dalam fungsinya sendiri membuat sistem menjadi sangat mudah dipelihara (maintainable). Jika rumus berubah, hanya modul ini yang perlu diubah [7]

## Unguided 7

```
FUNCTION TentukanIndeks (skorAkhir : real) -> string

// DEKLARASI LOKAL
// Variabel untuk menyimpan hasil indeks nilai yang ditentukan.
VAR indeksHasil : string

// ALGORITMA
// Gunakan struktur percabangan IF-ELSE IF-ELSE untuk mengecek skor dari
// kondisi tertinggi hingga terendah.
IF skorAkhir > 80 THEN
    indeksHasil <- "A"
ELSE IF skorAkhir > 70 THEN // Otomatis berarti <= 80
    indeksHasil <- "AB"
ELSE IF skorAkhir > 65 THEN // Otomatis berarti <= 70
    indeksHasil <- "B"
ELSE IF skorAkhir > 60 THEN // Otomatis berarti <= 65
    indeksHasil <- "BC"
ELSE IF skorAkhir > 50 THEN // Otomatis berarti <= 60
    indeksHasil <- "C"
ELSE IF skorAkhir > 40 THEN // Otomatis berarti <= 50
    indeksHasil <- "D"
ELSE // Jika tidak ada kondisi di atas yang terpenuhi, berarti skor <= 40
    indeksHasil <- "E"
END IF

// Kembalikan nilai indeks yang sudah ditentukan.
RETURN indeksHasil

END FUNCTION
```

## Deskripsi

FUNCTION TentukanIndeks adalah implementasi murni dari struktur kontrol seleksi (IF-ELSE). Ini adalah salah satu dari tiga pilar utama algoritma (bersama sekuens dan iterasi). Kemampuan program untuk membuat keputusan berdasarkan kondisi adalah inti dari setiap perangkat lunak yang "cerdas" dan merupakan konsep dasar dalam pendidikan komputasi [8].

## G. Kesimpulan

### Kesimpulan Guided

Praktikum pengenalan bahasa pemrograman C++ telah berhasil dilaksanakan. Melalui praktikum ini, telah dipahami dan dibuktikan secara praktis konsep-konsep fundamental C++. Praktikan berhasil mengimplementasikan struktur dasar program dan operasi *input/output* melalui program "Hello, World!". Konsep manajemen memori tingkat rendah juga telah dipahami melalui penggunaan **pointer**, operator referensi (&), dan dereferensi (\*). Terakhir, konsep **array** sebagai struktur data statis untuk mengelola kumpulan data homogen telah berhasil diterapkan. Seluruh kegiatan ini memberikan fondasi yang kuat untuk mempelajari topik struktur data yang lebih kompleks di masa mendatang.

### Kesimpulan Unguided

review mata kuliah Algoritma & Pemrograman, yang mengubah teori menjadi solusi nyata melalui tiga langkah kunci:

Desain Data: Kita tidak lagi hanya memakai tipe data dasar, tetapi merancang struct untuk memodelkan informasi secara logis sesuai masalah.

Desain Modular: Kita tidak menulis satu kode panjang, melainkan memecah program menjadi fungsi-fungsi dengan satu tugas spesifik (merakit data, menghitung, dan memutuskan).

Desain Algoritma: Kita menerapkan alur kontrol (IF-ELSE) untuk menerjemahkan aturan menjadi logika yang bisa dieksekusi oleh program.

Pada dasarnya, ini adalah fondasi dari problem-solving: pikirkan datanya, pecah masalahnya, lalu susun alurnya.

## H. Referensi

- [1] Martins, L., & Garcia, A. (2023). "Modern C++ in Software Engineering Curricula: A Pedagogical Review." *IEEE Transactions on Education*, 66(2), 112-120.
- [2] Schmidt, C. (2022). "Performance Paradigms in C++20 for High-Performance Computing." *Journal of Parallel and Distributed Computing*, 165, 45-58.
- [3] Albluwi, I. (2021). "A Systematic Review of Introductory Programming Courses: A Focus on Pedagogy, Language, and Environment." *IEEE Access*, 9, 139625-139640.

[4] Lee, J., & Park, S. (2024). "Visualizing Pointers and Memory Allocation to Improve Comprehension in Novice Programmers." *ACM Transactions on Computing Education*, 24(1), Article 5.

[5] Ludi, S., & Johnson, A. (2021). "Data Structure Abstractions: A Comparative Study of Raw Arrays vs. STL Containers." In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21)*, 512-518.

[6] Gaddis, T. (2021). *Starting Out with C++: From Control Structures through Objects (10th ed.)*. Pearson.

[7] Sommerville, I. (2021). *Software Engineering (11th ed.)*. Pearson.

[8] Hellas, A., et al. (2023). *A Systematic Literature Review on the Use of Large Language Models in Programming Education*. *ACM Transactions on Computing Education*, 24(1), 1-36.