

Modul 4

1. Capaian Pembelajaran Matakuliah (CPMK)

CPMK143:

Mampu menggunakan bahasa pemrograman, pustaka, framework, IDE, version control, debugging tools, dan tools pendukung lainnya secara tepat dalam proses pengembangan perangkat lunak.

Indikator Ketercapaian:

Mahasiswa memahami constructor dan package.

2. Dasar Teori

CONSTRUCTOR, PACKAGE/NAMESPACE, DAN OVERLOADING

1. Constructor (Konstruktor)

Constructor adalah metode khusus di dalam *class* yang dipanggil secara otomatis saat sebuah objek diinstansiasikan. Karakteristik utama dari *constructor*:

- Memiliki nama yang **sama persis** dengan nama *class*.
- **Tidak memiliki nilai kembalian** (*return value*), bahkan tidak menggunakan *void*.
- Digunakan untuk mengalokasikan memori awal dan menginisialisasi atribut objek.

2. Method & Constructor Overloading

Overloading adalah mekanisme dalam OOP di mana dua atau lebih fungsi/metode di dalam kelas yang sama memiliki **nama yang identik**, namun **jumlah atau tipe parameternya berbeda**.

- **Constructor Overloading:** Menyediakan beberapa pilihan opsi saat membuat objek (misalnya: membuat objek tanpa parameter awal vs membuat objek dengan parameter awal lengkap).
- **Method Overloading:** Mengizinkan satu nama *method* untuk memproses jenis tipe data atau jumlah masukan yang berbeda.

3. Namespace (Konsep Package pada C++)

Jika di bahasa seperti Java kita mengenal *Package*, maka di C++ konsep pengelompokan kode modular tersebut menggunakan **namespace**.

Namespace digunakan untuk mencegah terjadinya **Name Collision** (bentrokan nama *class* atau fungsi) ketika proyek bertambah besar atau saat menggunakan perpustakaan (*library*) pihak ketiga.

3. Alat dan Bahan

Hardware :

1. Laptop/PC
2. LCD/proyektor
3. Jaringan internet

Software :

1. Sistem operasi Windows/Linux/macOS
2. Java Development Kit (JDK)
3. IDE pemrograman: NetBeans / IntelliJ IDEA / Visual Studio Code
4. LMS untuk materi dan aktivitas pembelajaran

4. Langkah Kerja / Prosedur Praktikum

Membuat Berkas Program Module 4:

Persiapan Workspace.

1. Buka aplikasi **Visual Studio Code**.
2. Buka folder kerja praktikum Anda.
3. Buat berkas baru bernama modul4_advanced_oop.cpp.

Menuliskan Namespace, Class, Constructor Overloading, dan Method Overloading:

Langkah 1.

Ketikkan kode berikut di dalam berkas modul4_advanced_oop.cpp:

```
#include <iostream>
#include <string>

// 1. PENERAPAN NAMESPACE (Konsep Package di C++)
namespace ModulKomputasi {

    class KalkulatorMatematika {
    private:
        std::string namaSistem;

    public:
        // 2. CONSTRUCTOR OVERLOADING
        // Constructor 1: Tanpa Parameter (Default Constructor)
        KalkulatorMatematika() {
            namaSistem = "Kalkulator Standar";
        }
    }
```

```

// Constructor 2: Dengan Parameter
KalkulatorMatematika(std::string label) {
    namaSistem = label;
}

// Getter untuk atribut namaSistem
std::string getNamaSistem() {
    return namaSistem;
}

// 3. METHOD OVERLOADING
// Overload 1: Menghitung penjumlahan 2 bilangan bulat (int)
int tambah(int a, int b) {
    return a + b;
}

// Overload 2: Menghitung penjumlahan 3 bilangan bulat (int)
int tambah(int a, int b, int c) {
    return a + b + c;
}

// Overload 3: Menghitung penjumlahan 2 bilangan desimal (double)
double tambah(double a, double b) {
    return a + b;
}
};

} // Akhir dari Namespace

```

Lanjutkan dengan menambahkan fungsi main() di bawah deklarasi namespace pada berkas yang sama:

```

int main() {
    // Menggunakan scope resolution operator (::) untuk mengakses class dalam
    namespace

    // Objek 1 menggunakan Constructor 1 (Tanpa Parameter)
    ModulKomputasi::KalkulatorMatematika k1;

    // Objek 2 menggunakan Constructor 2 (Dengan Parameter)
    ModulKomputasi::KalkulatorMatematika k2("Kalkulator Ilmiah Advanced");

    std::cout << "--- PENGUJIAN CONSTRUCTOR ---" << std::endl;
}

```

```

std::cout << "Sistem 1: " << k1.getNamaSistem() << std::endl;
std::cout << "Sistem 2: " << k2.getNamaSistem() << std::endl;

std::cout << "\n--- PENGUJIAN METHOD OVERLOADING ---" << std::endl;
// Memanggil Overload 1 (2 int)
std::cout << "2 + 3    = " << k1.tambah(2, 3) << std::endl;

// Memanggil Overload 2 (3 int)
std::cout << "2 + 3 + 4 = " << k1.tambah(2, 3, 4) << std::endl;

// Memanggil Overload 3 (2 double)
std::cout << "2.5 + 4.3 = " << k1.tambah(2.5, 4.3) << std::endl;

return 0;
}

```

Buka Terminal di VS Code.

Jalankan perintah kompilasi. Amati dan analisis program.

5. Lembar Kerja

Latihan 4

Buat sebuah program dengan contoh **Constructor** dan lakukan coding program seperti dalam modul 4.

HASIL PERCOBAAN:

Screenshot Hasil LATIHAN 4 dan Berikan Analisis.
(UPLOAD ke LMS → TUGAS PRAKTIKUM 4)

Tugas Mandiri 4

(Kerjakan Tugas Mandiri 4 Upload ke LMS → **TUGAS TEORI 4**)

1. Apa fungsi utama dari *constructor* dalam sebuah *class*? Jelaskan perbedaan perilaku antara program yang menyediakan *constructor* buatan sendiri vs program yang tidak mendefinisikan *constructor* sama sekali?
2. Bagaimana kompilator C++ dapat membedakan metode mana yang harus dieksekusi ketika ada dua atau lebih metode yang memiliki nama sama (*method overloading*)? Apakah sekadar mengubah tipe data kembalian (*return type*) saja diperbolehkan dalam *overloading*? Jelaskan alasannya
3. Mengapa *namespace* sangat krusial dalam pengembangan proyek perangkat lunak berskala besar yang dikerjakan oleh tim? Apa risiko yang terjadi jika kita tidak menerapkan *namespace* pada proyek tersebut?



Program Studi
SISTEM KOMPUTER