

Modul 5

1. Capaian Pembelajaran Matakuliah (CPMK)

CPMK142:

Mampu menerapkan teknik pemrograman terstruktur, modular, dan berorientasi objek untuk menyelesaikan permasalahan komputasi sesuai kebutuhan aplikasi sistem komputer.

Indikator Ketercapaian:

Mahasiswa mampu menerapkan inheritance.

2. Dasar Teori

Apa itu Inheritance (Pewarisan)?

Inheritance adalah salah satu pilar utama OOP yang memungkinkan sebuah kelas baru (*derived class / child class*) mengadopsi atribut dan metode dari kelas yang sudah ada (*base class / parent class*).

Dengan pewarisan, kita tidak perlu menulis ulang kode yang sama untuk kelas-kelas yang memiliki karakteristik serupa, sehingga menciptakan struktur kode yang modular dan mudah dipelihara (*reusable code*).

Hubungan Parent Class dan Child Class

- **Base Class (Parent Class):** Kelas induk yang menyediakan atribut dan metode umum.
- **Derived Class (Child Class):** Kelas anak yang mewarisi fitur dari kelas induk, serta dapat memiliki atribut/metode spesifik tambahan sendiri.

Access Modifier dalam Inheritance

Tingkat aksesibilitas anggota kelas induk saat diturunkan ke kelas anak ditentukan oleh *specifier* pewarisan: Contoh:

```
class ParentClass {  
    protected:  
        // Anggota protected dapat diakses langsung oleh ChildClass,  
        // tetapi tetap tersembunyi dari luar kelas (main function).  
};  
  
class ChildClass : public ParentClass {  
    // Pewarisan bersifat public  
};
```

- **public Inheritance:** Anggota public induk menjadi public di anak; anggota protected induk tetap protected di anak.
- **protected Access Modifier:** Digunakan khusus agar atribut/metode tetap terenkapsulasi dari pihak luar, namun tetap dapat diakses secara langsung oleh kelas turunan (*child class*)

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 5:

Persiapan Workspace.

1. Buka aplikasi **Visual Studio Code**.
2. Buka folder kerja praktikum Anda.
3. Buat berkas baru bernama modul5_inheritance.cpp.

Menuliskan Induk dan Anak Class

Ketikkan kode berikut di dalam berkas modul5_inheritance.cpp:

Langkah 1.

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

// 1. BASE CLASS (Parent Class)
class Kendaraan {
protected:
    // Anggota protected dapat diakses langsung oleh child class
    std::string merk;
    int tahunProduksi;

public:
    // Constructor Induk
```

```

Kendaraan(std::string m, int t) {
    merk = m;
    tahunProduksi = t;
}

// Method Umum yang dapat digunakan kembali (Reusable Code)
void tampilkanInfoDasar() {
    std::cout << "Merk      : " << merk << std::endl;
    std::cout << "Tahun Produksi : " << tahunProduksi << std::endl;
}
};

// 2. DERIVED CLASS (Child Class 1)
class Mobil : public Kendaraan {
private:
    int jumlahPintu;

public:
    // Constructor Anak (Memanggil Constructor Induk)
    Mobil(std::string m, int t, int pintu) : Kendaraan(m, t) {
        jumlahPintu = pintu;
    }

    // Method Spesifik Kelas Mobil
    void tampilkanInfoMobil() {
        tampilkanInfoDasar(); // Memanggil method milik kelas induk
        std::cout << "Jumlah Pintu  : " << jumlahPintu << std::endl;
        std::cout << "Jenis Kendaraan: Mobil Roda 4" << std::endl;
    }
};

// 3. DERIVED CLASS (Child Class 2)
class SepedaMotor : public Kendaraan {
private:
    std::string tipeMesin; // Contoh: 2-tak, 4-tak, Listrik

public:
    // Constructor Anak (Memanggil Constructor Induk)
    SepedaMotor(std::string m, int t, std::string tipe) : Kendaraan(m, t) {
        tipeMesin = tipe;
    }
};

```

```

    }

    // Method Spesifik Kelas SepedaMotor
    void tampilkanInfoMotor() {
        tampilkanInfoDasar(); // Memanggil method milik kelas induk
        std::cout << "Tipe Mesin    : " << tipeMesin << std::endl;
        std::cout << "Jenis Kendaraan: Sepeda Motor Roda 2" << std::endl;
    }
};

```

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

```

int main() {
    std::cout << "=== INFORMASI MOBIL ===" << std::endl;
    Mobil mobilSaya("Toyota Avanza", 2022, 5);
    mobilSaya.tampilkanInfoMobil();

    std::cout << "\n=== INFORMASI SEPEDA MOTOR ===" << std::endl;
    SepedaMotor motorSaya("Honda Vario", 2021, "4-Tak");
    motorSaya.tampilkanInfoMotor();

    return 0;
}

```

Buka Terminal di VS Code.

Jalankan perintah kompilasi. Amati dan analisis program.

5. Lembar Kerja

Latihan 5

Buatlah program C++ lengkap dengan spesifikasi berikut:

- Buat **Base Class** bernama Karyawan dengan atribut protected: nama (string) dan idKaryawan (string), serta *method* tampilkanIdentitas().
- Buat **Derived Class 1** bernama Dosen yang mewarisi Karyawan, dengan atribut spesifik privat: nidn (string) dan *method* tampilkanDetailDosen().
- Buat **Derived Class 2** bernama Tendik (Tenaga Kependidikan) yang mewarisi Karyawan, dengan atribut spesifik privat: unitKerja (string) dan *method* tampilkanDetailTendik().

- Di dalam fungsi `main()`, buat masing-masing satu objek dari kelas `Dosen` dan `Tendik`, isi datanya melalui *constructor*, lalu tampilkan detail informasi dari kedua objek tersebut!

HASIL PERCOBAAN:

**Screenshot Hasil LATIHAN 5 dan Berikan Analisis.
(UPLOAD ke LMS → TUGAS PRAKTIKUM 5)**

Tugas Mandiri 5

(Kerjakan Tugas Mandiri 5 Upload ke LMS → **TUGAS TEORI 5**)

1. Bagaimana mekanisme *inheritance* membantu programmer untuk menerapkan prinsip DRY (*Don't Repeat Yourself*) dalam pembuatan perangkat lunak?
2. Mengapa pada contoh kode praktikum di atas atribut merk dan tahunProduksi pada kelas `Kendaraan` diatur sebagai `protected` dan bukan `private` atau `public`? Apa yang terjadi jika atribut tersebut diubah menjadi `private`?
3. Jelaskan bagaimana proses pemanggilan *constructor* terjadi pada saat objek dari kelas anak (`Mobil` atau `SepedaMotor`) diinstansiasikan! Mengapa sintaks : `Kendaraan(m, t)` diperlukan pada *constructor* kelas anak



Program Studi
SISTEM KOMPUTER