



Laporan THR (Tugas Hari Raya)

Mata Kuliah : Pemrograman Berorientasi Objek
Bahasan : Sistem Manajemen Karyawan di Perusahaan
Halaman : 1/12

NIM	242410103041
Nama	AHMAD ZAFARELL ZOUVAN DHANI
Kelas	C
Program Studi	INFORMATIKA
Asisten	ADRIAN FATHAN IMAMA MUHAMMAD TAUFIQ ROHMAN

LANGKAH KERJA

Sebuah perusahaan ingin membuat sistem manajemen karyawan yang mencakup karyawan tetap, karyawan kontrak, dan karyawan magang.

Persyaratan Implementasi

1. Buat kelas Karyawan yang memiliki atribut private: - Nama (string) - ID (string) - Gaji Pokok (double) Gunakan getter dan setter untuk mengakses atribut ini secara aman.

2. Karyawan adalah kelas parent. Karyawan Tetap, Karyawan Kontrak, dan Karyawan Magang adalah kelas turunan. Setiap jenis karyawan memiliki metode Hitung Gaji(), tetapi cara perhitungannya berbeda:

- Karyawan Tetap: Gaji akhir = Gaji Pokok + Bonus Tetap (500000)

- Karyawan Kontrak: Gaji akhir = Gaji Pokok - Potongan Kontrak (200000)

- Karyawan Magang: Gaji akhir = Gaji Pokok (tidak ada bonus atau potongan)

3. Gunakan override untuk mengganti metode Hitung Gaji() sesuai jenis karyawan.
4. Minta input dari pengguna untuk memasukkan jenis karyawan, nama, ID, dan gaji pokok. Buat objek sesuai jenis karyawan yang dipilih.
5. Hitung dan tampilkan gaji akhir karyawan tersebut

1. Deskripsi Program

Dalam implementasi program ini, saya memecah setiap class menjadi file .cs terpisah. Hal ini saya lakukan agar dapat memanfaatkan fitur class diagram dari Visual Studio, sehingga hubungan antar class akan lebih mudah divisualisasikan secara hierarkis dan terstruktur.

Program ini dibuat berdasarkan perintah soal, yaitu membuat sistem manajemen karyawan yang mencakup tiga jenis karyawan:

- Karyawan Tetap
- Karyawan Kontrak
- Karyawan Magang

Untuk membangun sistem ini, saya menerapkan konsep Object-Oriented Programming (OOP) seperti inheritance dan polymorphism yang telah saya pelajari melalui pertemuan teori dan praktikum. Konsep tersebut saya implementasikan sebagai kekuatan utama dalam menyusun arsitektur program ini.

Masing-masing jenis karyawan diwakili oleh class tersendiri yang merupakan turunan dari superclass Karyawan. Setiap karyawan memiliki atribut:

- Nama
- ID
- Gaji Pokok

Selain itu, masing-masing class turunan meng-override method Hitung_Gaji() dengan rumus yang sesuai. User akan diminta memasukkan jenis karyawan, lalu program akan meminta input data dan menghitung gaji akhir sesuai jenisnya.

2. Struktur Class

Class Karyawan merupakan **superclass** (induk) dalam hierarki sistem manajemen karyawan. Class ini berfungsi sebagai dasar yang akan diwarisi oleh class turunan seperti Karyawan_Tetap, Karyawan_Kontrak, dan Karyawan_Magang.

1. Atribut (Private Fields)

```
class Karyawan
{
    private string Nama;
    private string ID;
    private double Gaji_Pokok;
```

....

Class ini memiliki tiga atribut yang disimpan secara **private** untuk menjaga enkapsulasi:

- Nama (string) – Menyimpan nama karyawan.
- ID (string) – Menyimpan ID karyawan.
- Gaji_Pokok (double) – Menyimpan gaji pokok dari karyawan.

2. Properti (Getter & Setter)

```
public string nama
```

```
{
    get { return Nama; }
    set { Nama = value; }
}
```

```
public string id
```

```
{
    get { return ID; }
    set { ID = value; }
}
```

```
public double gaji_pokok
```

```
{
    get { return Gaji_Pokok; }
    set { Gaji_Pokok = value; }
}
```

...

Untuk mengakses atribut-atribut private di atas secara aman, disediakan tiga properti public:

- nama → mengakses dan mengubah Nama
- id → mengakses dan mengubah ID
- gaji_pokok → mengakses dan mengubah Gaji_Pokok

Properti ini memastikan bahwa akses ke data internal tetap terkendali dan mengikuti prinsip OOP.

3. Method: Hitung_Gaji()

```
public virtual void Hitung_Gaji()
{
    Console.WriteLine($"Nama: {nama}");
    Console.WriteLine($"ID: {id}");
    Console.WriteLine($"Gaji Akhir: {gaji_pokok}");
}
```

...

Method Hitung_Gaji() dideklarasikan sebagai virtual, yang artinya dapat dioverride oleh class-class turunan. Fungsi ini menampilkan data umum dari karyawan, yaitu:

- Nama
- ID
- Gaji Pokok (yang dianggap sebagai gaji akhir untuk karyawan magang)

4. Peran dalam Pewarisan

Sebagai superclass, class Karyawan menjadi dasar dari semua jenis karyawan. Semua class turunan akan mewarisi atribut dan method dari class ini, dan bisa menyesuaikan perilakunya dengan **override method** Hitung_Gaji() sesuai dengan kebijakan perhitungan masing-masing jenis karyawan.

Struktur Class Turunan. Ketiga class berikut merupakan turunan dari class Karyawan dan masing-masing mengoverride method Hitung_Gaji() sesuai kebijakan perhitungan gaji masing-masing jenis karyawan.

Class Karyawan_Tetap

Class ini mewakili karyawan tetap yang menerima bonus tetap sebesar 500.000 setiap kali perhitungan gaji.

```
class Karyawan_Tetap : Karyawan
```

```

{
    public override void Hitung_Gaji()
    {
        double bonus = 500_000;
        Console.WriteLine($"Nama: {nama}");
        Console.WriteLine($"ID: {id}");
        Console.WriteLine($"Gaji Akhir: {gaji_pokok + bonus}");
    }
}

```

Ciri khas:

- Menggunakan konsep method overriding untuk menambahkan bonus ke gaji pokok.
- Menampilkan nama, ID, dan gaji akhir.

Class Karyawan_Kontrak

Class ini mewakili karyawan kontrak yang dikenai potongan sebesar 200.000 dari gaji pokok.

class Karyawan_Kontrak : Karyawan

```

{
    public override void Hitung_Gaji()
    {
        double potongan = 200_000;
        Console.WriteLine($"Nama: {nama}");
        Console.WriteLine($"ID: {id}");
        Console.WriteLine($"Gaji Akhir: {gaji_pokok - potongan}");
    }
}

```

Ciri khas:

- Memotong gaji pokok sesuai nilai tetap (potongan kontrak).
- Sama seperti class tetap, menampilkan nama, ID, dan hasil akhir gaji.

Class Karyawan_Magang

Class ini mewakili karyawan magang yang tidak mendapat bonus atau potongan. Gaji akhirnya hanya sesuai gaji pokok.

```
class Karyawan_Magang : Karyawan
```

```
{  
    public override void Hitung_Gaji()  
    {  
        base.Hitung_Gaji();  
    }  
}
```

Ciri khas:

- Tidak mengubah perilaku Hitung_Gaji() dari parent, hanya memanggil method bawaan dari superclass (base.Hitung_Gaji()).
- Ini menegaskan bahwa gaji magang = gaji pokok, tanpa tambahan atau pengurangan.

Penjelasan Kode Program Program.cs

Class Program ini berisi Main(), yaitu titik masuk utama program. Di dalamnya terdapat alur logika utama dari sistem manajemen karyawan yang saya buat. Berikut adalah penjelasan bagian-bagian penting dari kode:

Blok try-catch

```
try {  
    // kode utama program  
}  
catch (FormatException) {  
    // jika input tidak valid (misal: huruf dimasukkan ke input angka)  
}  
catch (Exception ex) {  
    // menangani error lain yang tidak terduga  
}
```

Kenapa menggunakan try-catch?

Awalnya, saya menggunakan try-catch karena **penasaran dan ingin mencoba**, terinspirasi dari pengalaman pribadi saat mengerjakan proyek menggunakan Python, di mana try/except menjadi hal yang sangat membantu.

Ternyata, ini adalah **kebiasaan baik** karena:

- Mencegah **program crash** saat user salah memasukkan input (misalnya mengetik huruf ketika diminta angka).
- Menangani **error tak terduga lainnya** tanpa menghentikan eksekusi program secara brutal.
- Memberikan feedback yang **lebih ramah** kepada pengguna.

Contoh error yang bisa ditangkap:

- User mengetik abc saat diminta angka untuk pilihan atau gaji → `FormatException`.
- Error umum lainnya ditangani oleh `Exception` ex.

Langkah-langkah Program:

1. Menampilkan pilihan jenis karyawan

```
Console.WriteLine("Pilih jenis karyawan:");
```

```
Console.WriteLine("1. Karyawan Tetap\n2. Karyawan Kontrak\n3. Karyawan Magang");
```

```
int pilihan = Convert.ToInt32(Console.ReadLine());
```

➡ Program menampilkan tiga pilihan jenis karyawan, dan input dari user akan disimpan dalam variabel `pilihan`.

2. Mengambil data karyawan

```
Console.WriteLine("Nama Karyawan:");
```

```
string nama = Console.ReadLine();
```

```
Console.WriteLine("ID Karyawan:");
```

```
string id = Console.ReadLine();
```

```
Console.WriteLine("Gaji Pokok:");
```

```
double gaji = Convert.ToDouble(Console.ReadLine());
```

➡ Di sini user diminta untuk menginput **nama**, **ID**, dan **gaji pokok** karyawan. Nilai-nilai ini akan digunakan untuk mengisi data objek nanti.

3. Menentukan objek karyawan berdasarkan pilihan

```
Karyawan karyawan;
```

```
switch (pilihan) {
```

```
    case 1:
```

```
        karyawan = new Karyawan_Tetap();
```

```
        break;
```

```
    case 2:
```

```
        karyawan = new Karyawan_Kontrak();
```

```
        break;
```

```
    case 3:
```

```
        karyawan = new Karyawan_Magang();
```

```
        break;
```

```
    default:
```

```
        Console.WriteLine("Pilihan tidak valid.");
```

```
        return;
```

```
}
```

➡ Inilah inti dari **polymorphism** dalam program ini:

- Saya menggunakan variabel bertipe **Karyawan** (superclass).
- Lalu saya menginisialisasinya dengan objek **subclass** tergantung pilihan user:
 - Karyawan_Tetap
 - Karyawan_Kontrak
 - Karyawan_Magang

Hal ini memungkinkan pemanggilan method `Hitung_Gaji()` nanti akan **berperilaku berbeda** tergantung tipe objek yang sebenarnya.

4. Mengisi data dan menampilkan hasil

```
karyawan.nama = nama;
```

```
karyawan.id = id;
```

```
karyawan.gaji_pokok = gaji;
```

```
Console.WriteLine("\n--- Detail Gaji ---");
```

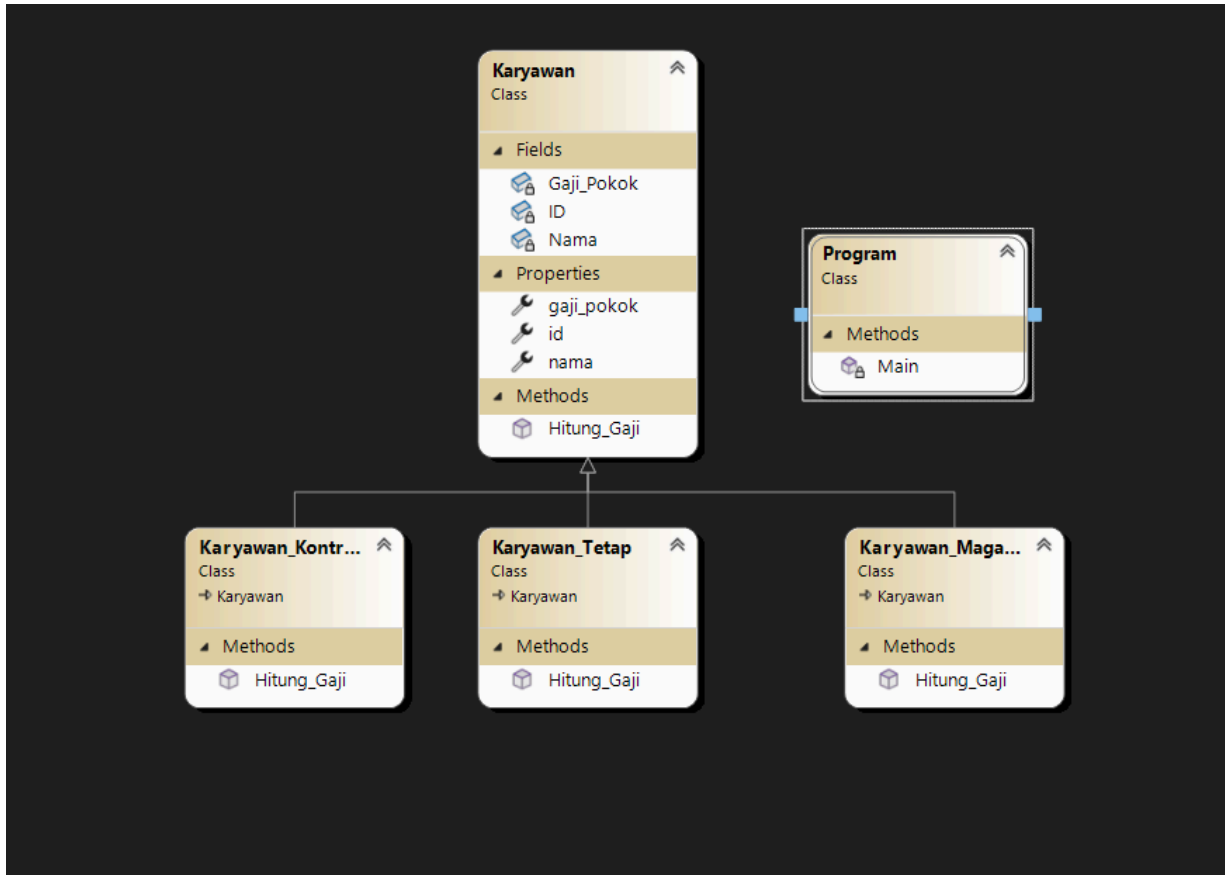
```
karyawan.Hitung_Gaji();
```

➡ Setelah objek dibuat, data yang diinput user dimasukkan ke **property** masing-masing.

Kemudian method `Hitung_Gaji()` dipanggil. Karena method ini di-*override* pada tiap subclass, hasilnya akan **berbeda-beda**:

- Karyawan_Tetap → Gaji pokok + bonus.
- Karyawan_Kontrak → Gaji pokok - potongan.
- Karyawan_Magang → Hanya menampilkan gaji pokok.

3. Class Diagram



Penjelasan Class Diagram


1. Class Karyawan (Superclass)

Sebagai superclass atau induk dari semua jenis karyawan.

Fields/variabel (Private):

- Nama
- ID
- Gaji_Pokok
(Disimbolkan dengan 🔒 karena bersifat private)

Properties (Public):

- nama
- id
- gaji_pokok
(Disimbolkan dengan  karena berupa property setter/getter)

Methods:

- Hitung_Gaji()

Method ini akan di-*override* oleh class turunan untuk menghitung gaji sesuai jenis karyawan.

2. Class Karyawan_Tetap, Karyawan_Kontrak, dan Karyawan_Magang (Subclass)

Ketiganya **mewarisi class Karyawan**, yang ditunjukkan dengan panah **berarah ke atas** (panah pewarisan/inheritance).

Isi Masing-Masing:

- Hanya terdapat 1 method Hitung_Gaji() di setiap class.
- Method tersebut adalah hasil **override dari method milik Karyawan**.

Catatan:

- Ini menunjukkan penggunaan **polymorphism** – method yang sama (Hitung_Gaji) namun perilakunya berbeda tergantung jenis objek.

3. Class Program

Ini adalah class tempat method Main() berada.

- Menjadi titik masuk (entry point) dari aplikasi.
- Tidak punya relasi inheritance karena tidak mewarisi dari class mana pun.
- Hanya memiliki method Main.

Relasi antar-class

Class	Hubungan	Keterangan
Karyawan	Superclass	Induk dari tiga jenis karyawan.
Karyawan_Tetap	Inherits Karyawan	Override Hitung_Gaji() dan tambahkan bonus.
Karyawan_Kontrak	Inherits Karyawan	Override Hitung_Gaji() dan tambahkan potongan.
Karyawan_Magang	Inherits Karyawan	Override Hitung_Gaji() tapi hanya memanggil method asli.
Program	Berdiri sendiri	Mengatur alur program dan menggunakan polymorphism untuk instansiasi objek.