NAMA : Ninda Fadillah Heryanuryanti

NIM : 202412029

MATA KULIAH : Pemograman Berorientasi Objek

a. Source Code

menajemen karyawan.py | mk.py | ManajemenKaryawan.py

PBO > ManajemenKaryawan.py > ...

```python
# Parent class: Karyawan
class Karyawan:
    # Membuat class induk bernama Karyawan
    def __init__(self, nama, id_karyawan, gaji_pokok):
        # Fungsi __init__ adalah konstruktor yang otomatis dijalankan saat objek dibuat
        # Inisialisasi atribut dasar karyawan: nama, id_karyawan, dan gaji_pokok
        self.nama = nama
        self.id_karyawan = id_karyawan
        self.gaji_pokok = gaji_pokok

    def hitung_gaji(self):
        # Method dasar untuk menghitung gaji
        # Di class induk, gaji hanya berdasarkan gaji pokok
        return self.gaji_pokok

    def info(self):
        # Method untuk menampilkan informasi lengkap karyawan
        # f-string digunakan agar bisa menampilkan variabel langsung di dalam teks
        return f"Karyawan : {self.nama}, ID: {self.id_karyawan}, Gaji: {self.hitung_gaji():,.1f}"


# Child class: Manager (turunan dari Karyawan)
class Manager(Karyawan):
    # Membuat class Manager yang mewarisi class Karyawan
    def __init__(self, nama, id_karyawan, gaji_pokok, tunjangan):
        # super() digunakan untuk memanggil konstruktor dari class induk (Karyawan)
        super().__init__(nama, id_karyawan, gaji_pokok)
        # Menambahkan atribut baru khusus untuk Manager, yaitu tunjangan
        self.tunjangan = tunjangan

    def hitung_gaji(self):
        # Override method dari class Karyawan
        # Pada Manager, gaji adalah gaji pokok ditambah tunjangan
        return self.gaji_pokok + self.tunjangan

    def info(self):
        # Override method info untuk menampilkan informasi Manager
        return f"Manager : {self.nama}, ID: {self.id_karyawan}, Gaji: {self.hitung_gaji():,.1f}"


# Child class: Programmer (turunan dari Karyawan)
class Programmer(Karyawan):
    # Membuat class Programmer yang juga mewarisi class Karyawan
    def __init__(self, nama, id_karyawan, gaji_pokok, bonus):
        # super() memanggil konstruktor dari class Karyawan
        super().__init__(nama, id_karyawan, gaji_pokok)
        # Menambahkan atribut baru khusus Programmer, yaitu bonus
        self.bonus = bonus

    def hitung_gaji(self):
        # Override method dari class Karyawan
        # Pada Programmer, gaji adalah gaji pokok ditambah bonus
        return self.gaji_pokok + self.bonus

    def info(self):
        # Override method info untuk menampilkan informasi Programmer
        return f"Programmer : {self.nama}, ID: {self.id_karyawan}, Gaji: {self.hitung_gaji():,.1f}"
```

```
58
59
60  # ==== Bagian utama program ====
61  if __name__ == "__main__":
62      # Baris ini memastikan kode di bawah hanya dijalankan jika file ini dijalankan langsung
63      # bukan saat diimpor sebagai modul di file lain
64
65      # Membuat objek dari class Manager dan Programmer
66      manager1 = Manager("Ninda", "M002", 12_000_000, 3_000_000)   # Membuat objek Manager dengan nama, ID, gaji, dan t
67      programmer1 = Programmer("Fadillah", "P001", 10_000_000, 2_000_000)  # Membuat objek Programmer dengan nama, ID,
68
69      # Polymorphism: menyimpan objek Manager dan Programmer dalam satu list
70      daftar_karyawan = [manager1, programmer1]
71
72      # Menggunakan perulangan untuk menampilkan informasi semua karyawan
73      # Meskipun kedua objek berbeda class, mereka bisa menggunakan method info() yang sama
74      for karyawan in daftar_karyawan:
75          print(karyawan.info())  # Menampilkan hasil info sesuai class masing-masing
76
```

b. Output

```
PS C:\Users\hero3\Documents\python ninda> & C:/Users/hero
on ninda/PBO/ManajemenKaryawan.py"
Manager : Ninda, ID: M002, Gaji: 15,000,000.0
Programmer : Fadillah, ID: P001, Gaji: 12,000,000.0
PS C:\Users\hero3\Documents\python ninda>
```

c. Penjelasan

Program di atas merupakan contoh penerapan konsep pewarisan (inheritance) dan polimorfisme (polymorphism) dalam pemrograman berorientasi objek menggunakan bahasa Python. Program ini diawali dengan pembuatan kelas induk bernama Karyawan, yang berfungsi sebagai dasar untuk kelas turunan lainnya. Di dalam kelas ini terdapat konstruktor __init__ yang berfungsi untuk menginisialisasi atribut dasar seperti nama, id_karyawan, dan gaji_pokok. Selain itu, terdapat metode hitung_gaji() yang mengembalikan nilai gaji pokok, serta metode info() yang digunakan untuk menampilkan informasi lengkap tentang karyawan dalam format teks.

Selanjutnya, program memiliki dua kelas turunan yaitu Manager dan Programmer yang sama-sama mewarisi sifat dari kelas Karyawan. Pada kelas Manager, konstruktor __init__ memanggil konstruktor induknya menggunakan super().__init__ agar atribut dasar tetap terinisialisasi, kemudian menambahkan atribut baru yaitu tunjangan. Metode hitung_gaji() di-*override* agar menghitung gaji total dengan menambahkan tunjangan. Hal yang sama juga dilakukan pada kelas Programmer, tetapi atribut tambahannya adalah bonus. Keduanya juga memiliki metode info() yang di-*override* untuk menampilkan informasi sesuai jabatan masing-masing.

Pada bagian akhir program terdapat blok if __name__ == "__main__":, yang berfungsi untuk memastikan bahwa kode di dalamnya hanya dijalankan ketika file ini dieksekusi langsung, bukan saat diimpor ke file lain. Di dalam blok ini, dibuat dua objek yaitu manager1 dan programmer1 dengan data gaji dan tunjangan/bonus masing-masing. Kedua objek tersebut dimasukkan ke dalam list daftar_karyawan, lalu dilakukan perulangan for untuk menampilkan informasi masing-masing karyawan dengan memanggil metode info(). Konsep polimorfisme terlihat di sini karena meskipun objeknya berbeda (Manager dan Programmer), keduanya bisa diperlakukan sama saat metode info() dipanggil.