

Posttest Pemrograman Berorientasi Objek Praktik
Pertemuan ke-6

Nama	:	Muhamad Zaki As Shidiqi
NPM	:	5240411230
Kelas	:	VI

Petunjuk Pengerjaan : Jawablah secara singkat dan jelas. Waktu pengerjaan maksimal 25 menit.

Soal No. 1

Jelaskan bagaimana hubungan antara Class Diagram UML dan kode Python!

Jawab:

Use Case Diagram digunakan untuk menjelaskan apa yang dilakukan oleh sistem, sedangkan Class Diagram menggambarkan struktur sistem yang terdiri dari kelas, atribut, dan metode. Pada pertemuan hari ini, fokusnya adalah menerjemahkan Class Diagram menjadi kode Python, di mana UML berperan dalam perancangan sistem, sedangkan kode Python menjadi implementasi nyata dari desain tersebut.

Soal No. 2

Berikan contoh sederhana kelas Python yang memiliki atribut public, protected, dan private sekaligus!

Jawab:

```
class Karyawan:
    def __init__(self, nama, gaji, id_karyawan):
        self.nama = nama
        self._gaji = gaji
        self.__id_karyawan = id_karyawan

    def tampilkan_data(self):
        print(f>Nama: {self.nama}")
        print(f>Gaji: {self._gaji}")
        print(f>ID Karyawan: {self.__id_karyawan}")

k1 = Karyawan("Andi", 5000000, "KRY001")
print(k1.nama)
print(k1._gaji)
print(k1._Karyawan__id_karyawan)
k1.tampilkan_data()
```

Soal No. 3

Mengapa kita perlu menggunakan getter dan setter dalam pemrograman berorientasi objek?

Jawab:

Untuk mencapai enkapsulasi data, yang melindungi data dari akses dan modifikasi yang tidak disengaja

Soal No. 4

Apa yang akan terjadi jika kita mencoba mengakses atribut **__password** secara langsung dari luar kelas? Bagaimana cara yang benar untuk mengaksesnya?

Jawab:

Akan terjadi error, cara mengaksesnya dapat menggunakan getter untuk mengamankan data.

Soal No. 5

Buatlah contoh sederhana (tidak perlu lebih dari 10 baris) kelas Python yang menggambarkan diri Anda sendiri sebagai mahasiswa, dengan minimal:

- 1 atribut public,
- 1 atribut protected,
- 1 atribut private, dan
- 1 metode untuk menampilkan deskripsi diri Anda menggunakan ketiga atribut tersebut.

Lalu buatlah objek dengan data yang sesuai dengan data diri Anda sendiri (bukan data *dummy*)

Jawab:

```
class Mahasiswa:
    def __init__(self, nama, prodi, npm):
        self.nama = nama
        self._prodi = prodi
        self.__npm = npm

    def deskripsi_diri(self):
        print(f"Halo, saya {self.nama} dari program studi {self._prodi} dengan NPM {self.__npm}.")

mhs = Mahasiswa("Muhammad Zaki As Shidiqi", "Informatika", "5240411230")
mhs.deskripsi_diri()
```