Laporan Praktikum Algoritma & Struktur Data

Modul I –Array, Memory, Class, dan Object

Muhammad Yusron Nashrulloh/22103001005

Dosen : Achmad Arif Munaji, ST., M.Kom

Tanggal praktikum: 29 September 2023

myusron434@gmail.com

Teknik Komputer

Institut Teknologi dan Sains Nahdlatul Ulama Kalimantan

***Abstrak*—Praktikum ini bertujuan untuk memahami konsep dasar pemrograman berorientasi objek (OOP) dengan menggunakan bahasa pemrograman Python. Dalam praktikum ini, kita fokus pada pembuatan kelas dan objek sebagai cara untuk merepresentasikan entitas dalam program komputer.** **Hasi akhir dari praktikum ini adalah pemahaman yang lebih baik tentang bagaimana menggunakan OOP dalam Python untuk merepresentasikan objek-objek dan mengelola informasi tentang entitas tertentu. Ini merupakan dasar yang penting dalam pengembangan perangkat lunak yang melibatkan pemodelan dunia nyata ke dalam program komputer. Dengan demikian, praktikum ini membekali peserta dengan keterampilan dasar dalam penggunaan konsep OOP dalam Python.**

***Kata kunci—Array,Memory,Class, dan Object***

1. hasil dan pembahasan

Pada praktikum Algoritma & Struktur Data yang pertama, terdapat beberapa pmebahasan yaitu mengenai Array,Memory, Class dan Object. Berikut merupakan codingan dan penjelasan dari hasil praktikum yang telah dilakukan :

Tugas 1 :

   class Mobil:

def \_\_init\_\_(self, merek, model, tahun\_produksi, harga):

self.merek = merek

self.model = model

self.tahun\_produksi = tahun\_produksi

self.harga = harga

def informasi(self):

return f"Mobil {self.merek} {self.model} tahun {self.tahun\_produksi}, harga: {self.harga} USD"

Kelas Mobil memiliki atribut seperti merek, model, tahun produksi, dan harga. Ini memungkinkan Anda untuk membuat objek yang merepresentasikan mobil dengan informasi tertentu. Metode informasi digunakan untuk mengembalikan informasi lengkap tentang mobil berdasarkan atribut-atributnya.

class Mahasiswa:

    def \_\_init\_\_(self, nama, usia, jurusan):

        self.nama = nama

        self.usia = usia

        self.jurusan = jurusan

    def informasi(self):

        return f"Mahasiswa {self.nama}, usia {self.usia} tahun, jurusan: {self.jurusan}"

Kelas Mahasiswa memiliki atribut seperti nama, usia, dan jurusan. Ini memungkinkan Anda untuk membuat objek yang merepresentasikan mahasiswa dengan informasi tertentu. Metode informasi digunakan untuk mengembalikan informasi lengkap tentang mahasiswa berdasarkan atribut-atributnya.

class Buku:

    def \_\_init\_\_(self, judul, penulis, tahun\_terbit, harga):

        self.judul = judul

        self.penulis = penulis

        self.tahun\_terbit = tahun\_terbit

        self.harga = harga

    def informasi(self):

        return f"Buku '{self.judul}' oleh {self.penulis}, tahun terbit: {self.tahun\_terbit}, harga: {self.harga} USD"

Kelas Buku memiliki atribut seperti judul, penulis, tahun terbit, dan harga. Ini memungkinkan Anda untuk membuat objek yang merepresentasikan buku dengan informasi tertentu. Metode informasi digunakan untuk mengembalikan informasi lengkap tentang buku berdasarkan atribut-atributnya.

# Membuat objek Mobil

mobil1 = Mobil("Toyota", "Camry", 2020, 25000)

mobil2 = Mobil("Honda", "Civic", 2019, 22000)

# Menampilkan informasi mobil

print(mobil1.informasi())

print(mobil2.informasi())

# Membuat objek Mahasiswa

mahasiswa1 = Mahasiswa("John Doe", 20, "Teknik Informatika")

mahasiswa2 = Mahasiswa("Jane Smith", 22, "Manajemen")

# Menampilkan informasi mahasiswa

print(mahasiswa1.informasi())

print(mahasiswa2.informasi())

# Membuat objek Buku

buku1 = Buku("Harry Potter and the Sorcerer's Stone", "J.K. Rowling", 1997, 15)

buku2 = Buku("The Great Gatsby", "F. Scott Fitzgerald", 1925, 12)

Dalam bagian ini, objek-objek dari kelas Mobil, Mahasiswa, dan Buku dibuat dengan atribut-atribut yang berbeda. Setiap objek dipersonalisasi dengan informasi yang relevan.

# Menampilkan informasi mobil

print(mobil1.informasi())

print(mobil2.informasi())

# Menampilkan informasi mahasiswa

print(mahasiswa1.informasi())

print(mahasiswa2.informasi())

# Menampilkan informasi buku

print(buku1.informasi())

print(buku2.informasi())

Pada bagian ini, metode informasi dari setiap objek digunakan untuk mencetak informasi lengkap tentang objek tersebut. Hasil keluaran dari kode ini adalah tampilan informasi lengkap tentang mobil, mahasiswa, dan buku yang telah dibuat. Ini adalah contoh dasar penggunaan kelas dan objek dalam Python untuk mengelola dan menampilkan informasi tentang berbagai entitas.

Tugas 3 :

Definisi Kelas Mahasiswa:

class Mahasiswa:

    def \_\_init\_\_(self, nama, usia, jurusan):

        self.nama = nama

        self.usia = usia

        self.jurusan = jurusan

    def informasi(self):

        return f"Mahasiswa {self.nama}, usia {self.usia} tahun, jurusan: {self.jurusan}"

Kelas Mahasiswa memiliki tiga atribut: nama, usia, dan jurusan. Konstruktor \_*\_init*\_\_ digunakan untuk menginisialisasi atribut-atribut ini saat objek Mahasiswa dibuat. Metode informasi digunakan untuk mengembalikan informasi lengkap tentang mahasiswa berdasarkan atribut-atribut yang telah diatur.

Membuat Objek Mahasiswa :

mahasiswa1 = Mahasiswa("Yusron", 20, "Teknik Komputer")

mahasiswa2 = Mahasiswi("Indriani", 22, "Manajemen")

mahasiswa3 = Mahasiswa("Fajar", 20, "Psikologi")

Dalam bagian ini, tiga objek Mahasiswa dibuat dengan atribut-atribut yang berbeda. Setiap objek merepresentasikan seorang mahasiswa dengan nama, usia, dan jurusan yang berbeda.

Menyimpan Objek Dalam Array :

daftar\_mahasiswa = [mahasiswa1, mahasiswa2, mahasiswa3]

Array daftar\_mahasiswa digunakan untuk menyimpan objek-objek Mahasiswa yang telah dibuat sebelumnya. Ini memungkinkan Anda untuk dengan mudah mengelola banyak objek Mahasiswa dalam satu struktur data.

Menampilkan Informasi Mahasiswa :

print("Daftar Mahasiswa:")

for mahasiswa in daftar\_mahasiswa:

    print(mahasiswa.informasi())

Dalam bagian ini, kode akan menampilkan informasi lengkap tentang setiap mahasiswa dalam daftar\_mahasiswa menggunakan loop for. Metode informasi() dari setiap objek Mahasiswa dipanggil untuk mencetak informasi tentang nama, usia, dan jurusan masing-masing mahasiswa.

Hasil keluaran dari kode ini akan mencetak daftar mahasiswa beserta atribut-atribut mereka seperti berikut:

Daftar Mahasiswa:

Mahasiswa Yusron, usia 20 tahun, jurusan: Teknik Komputer

Mahasiswa Indriani, usia 22 tahun, jurusan: Manajemen

Mahasiswa Fajar, usia 20 tahun, jurusan: Psikologi

1. Kesimpulan

Dengan kombinasi penggunaan kelas, objek, dan struktur data seperti array, kita dapat dengan mudah mengelola informasi tentang berbagai entitas dan objek dalam program Python. Hal ini sangat berguna dalam pengembangan perangkat lunak yang kompleks, di mana pengorganisasian data dan fungsionalitas yang baik sangat penting. Dengan demikian, praktikum ini memberikan dasar yang kuat dalam penggunaan OOP dalam pemrograman Python.