**LAPORAN RESMI**

**MODUL I**

**(ClASS, OBJECT DAN CONSTRUCTOR)**

**PEMROGRAMAN BERBASIS OBJEK**

**NAMA : ADYTTA PUTRA TARIGAN**

**N.R.P : 240441100139**

**DOSEN : YUDHA DWI PUTRA NEGARA, S.KOM., M.KOM.**

**ASISTEN :** **AHMAD RIKHAN ARBA’I**

**TGL PRAKTIKUM : 22 MARET 2025**

**Disetujui : 10 MARET 2025**

**Asisten**

**AHMAD RIKHAN ARBA’I**

**23.04.411.00192**



**LABORATORIUM TEKNOLOGI INFORMASI**

**PRODI SISTEM INFORMASI**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# **BAB I PENDAHULUAN**

## **Latar Belakang**

Dalam dunia pemrograman modern, konsep Class, Object, dan Constructor menjadi fondasi utama dalam paradigma Pemrograman Berorientasi Objek (OOP). Dengan adanya class, programmer dapat mendefinisikan blueprint atau cetak biru dari suatu objek, yang berisi atribut dan metode untuk mengelola data serta perilaku objek tersebut. Object adalah instansi dari class yang dapat digunakan untuk merepresentasikan entitas nyata dalam dunia nyata, seperti mobil, mahasiswa, atau akun pengguna. Konsep ini memungkinkan pengembangan perangkat lunak yang lebih terstruktur, modular, dan mudah dikelola.

Salah satu elemen penting dalam OOP adalah constructor, yaitu metode khusus yang dieksekusi saat objek dibuat. Constructor digunakan untuk menginisialisasi nilai awal dari atribut suatu objek, sehingga tidak perlu diatur secara manual setelah objek dibuat. Dengan adanya constructor, kode program menjadi lebih efisien dan mudah dibaca. Selain itu, constructor juga memungkinkan pembuatan objek yang konsisten dan bebas dari kesalahan pengaturan awal.

Pemahaman yang baik tentang class, object, dan constructor sangat penting bagi programmer untuk mengembangkan aplikasi yang skalabel dan mudah diperbaiki. Dengan menerapkan prinsip OOP secara tepat, pengembangan perangkat lunak menjadi lebih fleksibel, mendukung penggunaan kembali kode (code reusability), serta memudahkan proses debugging dan pemeliharaan jangka panjang.

## **Tujuan**

* Mahasiswa mampu memahami konsep Class
* Mahasiswa mampu menggunakan Object dalam program
* Mahasiswa mampu mengimplementasikan Constructor secara efektif

# **BAB II DASAR TEORI**

## **Class, Object dan Constructor**

### **Class**

Class adalah blueprint atau cetak biru yang digunakan untuk mendefinisikan struktur dan perilaku dari suatu objek dalam pemrograman berorientasi objek (OOP). Class berisi atribut (variabel) dan method (fungsi) yang menjelaskan karakteristik dan aksi yang dapat dilakukan oleh objek yang dibuat dari class tersebut.

Contoh Analogis, Bayangkan sebuah class sebagai cetak biru atau blueprint adalah rumah. Dalam blueprint ini, kita menentukan apa saja yang akan dimiliki oleh rumah, seperti jumlah kamar, warna tembok, dll. Namun, blueprint itu sendiri bukanlah rumah, melainkan hanya rancangan atau panduan untuk membangun rumah.

1. Ciri-Ciri Class.

a. Mendefinisikan atribut (properti) dan method (fungsi) yang bisa dimiliki oleh objek.

b. Menjadi cetak biru blueprint untuk membuat objek.

c. Penulisan Nama pada class diawali dengan huruf kapital sesuai konvensi pemrograman Python.

2. Syntax Penulisan pada Class

Dalam Python, Class merupakan blueprint atau cetak biru untuk membuat objek. Class mendefiniskan atribut (variabel yang dimiliki oleh objek) dan metode (fungsi yang dapat dilakukan oleh objek).



### **Object**

Object adalah instansiasi atau perwujudan konkret dari suatu Class. Ketika kita membuat Object, kita menggunakan struktur yang sudah didefinisikan oleh Class dan memberikan nilai nyata untuk atribut-atribut tersebut. Objek dapat memanfaatkan method yang ada di class untuk melakukan aksi atau manipulasi data.

Contoh Analogis. Jika Class adalah cetak biru atau blueprint rumah, maka Object adalah rumah yang sebenarnya dibangun berdasarkan cetak biru tersebut. Setiap rumah yang dibangun dari blueprint yang sama bisa memiliki detail yang berbeda (misalnya warna yang berbeda, jumlah kamar yang berbeda).

1. Ciri-Ciri Object:

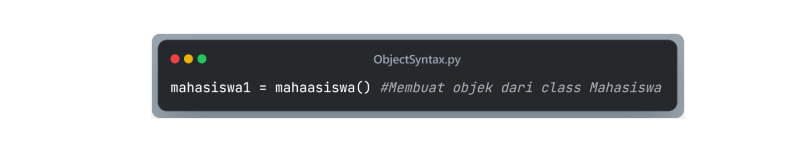
a. Dibuat menggunakan class.

b. Memiliki nilai konkret untuk atribut yang didefinisikan oleh class.

c. Dapat memanggil methoddari class untuk berinteraksi atau memanipulasi atributnya.

2. Syntax Penulisan pada Object.

Object merupakan instansi (instance) dari sebuah kelas. Setiap objek memiliki nilai atribut yang unik.



### **Constructor**

Constructor adalah method khusus yang digunakan untuk menginisialisasi

atribut dari Object saat objek tersebut dibuat. Dalam Python, constructor ditandai dengan method init (). Fungsi ini akan dijalankan secara otomatis saat kita membuat objek dari class. Biasanya, constructor digunakan untuk menetapkan nilai awal dari atribut-atribut objek.

Contoh Analogis: Constructor adalah pekerja yang menyiapkan rumah saat dibangun dari blueprint (class). Pekerja ini akan menyiapkan semua hal yang diperlukan (seperti memasang pintu, jendela, dll.) agar rumah siap digunakan.

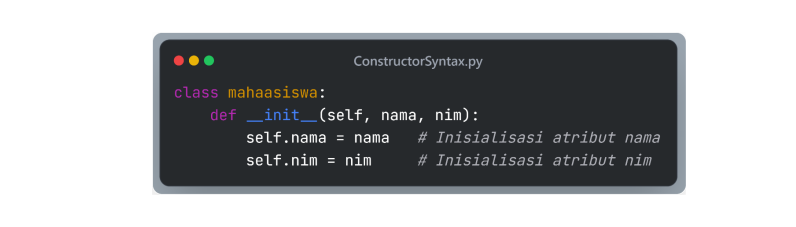
1. Ciri-Ciri Constructor

a. Didefinisikan menggunakan init (self, ...).

b. Secara otomatis dipanggil saat objek dibuat.

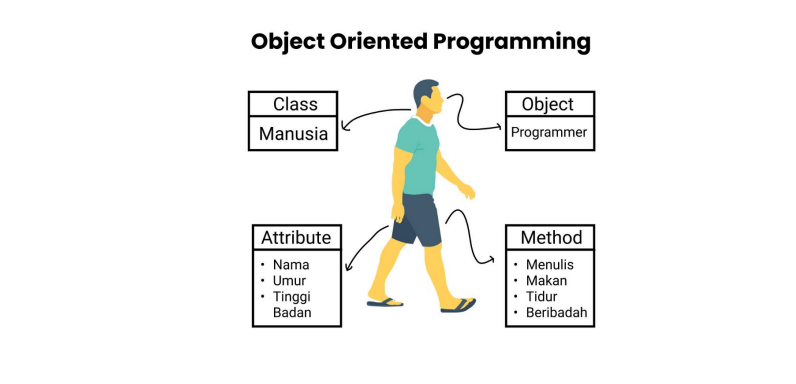
c. Biasanya digunakan untuk menginisialisasi atribut objek dengan nilai yang diberikan saat pembuatan objek.

2. Syntax Penulisan pada Constructor

Constructor merupakan metode khusus dalam sebuah kelas yang secara otomatis dipanggil ketika sebuah objek dibuat. Constructor digunakan untuk menginisialisasi atribut-atribut objek.

### **Ilustrasi Class, Object, dan Constructor**

Ilustrasi gambar yang memberikan representasi visual sederhana namun efektif tentang konsep dasar kelas, objek, dan konstruktor dalam OOP Python. Dengan memahami konsep-konsep ini, Anda dapat mulai membangun aplikasi yang lebih kompleks dan terstruktur dengan baik menggunakan bahasa pemrograman Python.



Gambar di atas menjelaskan konsep Object Oriented Programming (OOP) atau Pemrograman Berorientasi Objek dengan menggunakan contoh manusia. Berikut adalah penjelasan dari elemen-elemen yang ada dalam gambar:

1. Class(kelas) “Manusia”:

Di bagian kiri atas gambar terdapat sebuah kotak dengan label Class

yang berisi kata "Manusia". Ini menunjukkan bahwa dalam konsep OOP, Manusia adalah kelas yang menjadi cetak biru bagi berbagai objek yang dapat dibuat darinya.

1. Object (Objek) - "Programmer"

Di bagian kanan atas gambar, terdapat kotak dengan label Object yang berisi kata "Programmer". Ini menunjukkan bahwa Programmer adalah salah satu contoh objek yang dibuat dari kelas Manusia. Dalam dunia nyata, manusia bisa memiliki berbagai profesi, dan salah satu perwujudan dari manusia adalah seorang programmer.

1. Attribute (Atribut) :

Di bagian kiri bawah gambar, terdapat kotak dengan label Attribute yang berisi daftar karakteristik. Atribut adalah properti atau karakteristik yang dimiliki oleh sebuah objek.

Contoh atribut dalam kelas manusia :

1. Nama
2. Umur
3. Tinggi badan
4. Method (Metode) :

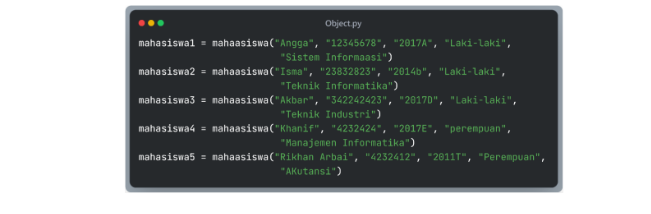
Di bagian kanan bawah gambar, terdapat kotak dengan label Method yang berisi daftar perilaku atau tindakan yang dapat dilakukan oleh objek.

Contoh metode dalam kelas manusia :

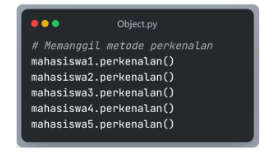
1. Menulis
2. Makan
3. Tidur
4. Beribadah

### **Program Python Dalam Penerapan Class, Object, dan Constructor**

Berikut merupakan program Perkenalan Mahasiswa menggunakan Class dengan bahasa Python.

Berikut merupakan program Perkenalan Mahasiswa menggunakan Object dengan bahas Python.

Berikut merupakan program Perkenalan Mahasiswa menggunakan Constructor dengan bahas Python.



# **BAB III TUGAS PENDAHULUAN**

# **BAB IV IMPLEMENTASI**

## **Tugas Praktikum**

### **Tugas Praktikum No. 1**

Buatlah class manusia dengan dengan attribut nama, umur dan alamat serta memiliki method berjalan dan berlari. Buatlah (minimal 5) objek dari class tersebut.

### **Tugas Praktikum No. 2**

Buatlah program untuk menampilkan nama, nim, jurusan/prodi, dan alamat mahasiswa menggunakan konsep OOP. Nama, nim, jurusan/prodi, dan alamat bersifat dinamis, dengan kata lain menggunakan input dari pengguna. Buat minimal 3 objek berdasarkan input pengguna.

### **Tugas Praktikum No. 3**

Buatlah tiga class hewan dengan attribut yang dimiliki hewan tersebut beserta method-methodnya juga. Buatlah beberapa objek dari class-class tersebut (minimal 3) menggunakan looping.

## **Source Code**

### **Source Code Soal 1**

|  |
| --- |
| **class manusia:**  **def \_\_init\_\_(self, nama, umur, alamat):**  **self.nama = nama**  **self.umur = umur**  **self.alamat = alamat**  **def berjalan(self):**  **print(f"{self.nama} sedang berjalan.")**  **def berlari(self):**  **print(f"{self.nama} sedang berlari.")**  **orang1 = manusia("Adytta", 19, "Jakarta")**  **orang2 = manusia("Fauzi", 20, "Jakarta")**  **orang3 = manusia("Abdul", 19, "Jakarta")**  **orang4 = manusia("Ilham", 20, "Jakarta")**  **orang5 = manusia("Teddy", 19, "Jakarta")**  **orang6 = manusia("Rehan", 20, "Jakarta")**  **orang1.berjalan()**  **orang2.berlari()**  **orang3.berjalan()**  **orang4.berlari()**  **orang5.berjalan()**  **orang6.berlari()** |

### **Source Code Soal 2**

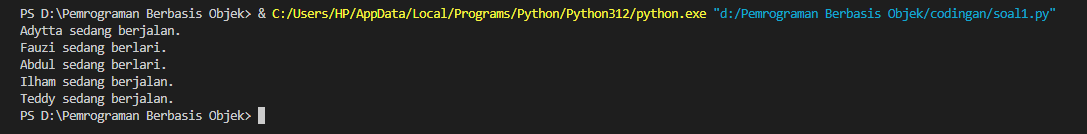
|  |
| --- |
| **class Mahasiswa:**  **def \_\_init\_\_(self, nama, nim, jurusan, alamat):**  **self.nama = nama**  **self.nim = nim**  **self.jurusan = jurusan**  **self.alamat = alamat**  **def tampilkan\_info(self):**  **print("\nData Mahasiswa")**  **print(f"Nama      : {self.nama}")**  **print(f"NIM       : {self.nim}")**  **print(f"Jurusan   : {self.jurusan}")**  **print(f"Alamat    : {self.alamat}")**  **mahasiswa\_list = []**  **for i in range(3):**  **print(f"\nMasukkan data mahasiswa ke-{i+1}:")**  **nama = input("Nama: ")**  **nim = input("NIM: ")**  **jurusan = input("Jurusan/Prodi: ")**  **alamat = input("Alamat: ")**  **mahasiswa = Mahasiswa(nama, nim, jurusan, alamat)**  **mahasiswa\_list.append(mahasiswa)**  **print("\nData Mahasiswa yang Dimasukkan")**  **for mhs in mahasiswa\_list:**  **mhs.tampilkan\_info()** |

### **Source Code Soal 3**

|  |
| --- |
| **class Kucing:**  **def \_\_init\_\_(self, nama, umur, warna):**  **self.nama = nama**  **self.umur = umur**  **self.warna = warna**  **def suara(self):**  **return f"{self.nama} mengeong: Meong!"**  **def info(self):**  **return f"Kucing bernama {self.nama}, umur {self.umur} tahun, warna {self.warna}."**  **class Anjing:**  **def \_\_init\_\_(self, nama, ras, umur):**  **self.nama = nama**  **self.ras = ras**  **self.umur = umur**  **def suara(self):**  **return f"{self.nama} menggonggong: Guk Guk!"**  **def info(self):**  **return f"Anjing bernama {self.nama}, ras {self.ras}, umur {self.umur} tahun."**  **class Burung:**  **def \_\_init\_\_(self, nama, jenis, warna\_bulu):**  **self.nama = nama**  **self.jenis = jenis**  **self.warna\_bulu = warna\_bulu**  **def suara(self):**  **return f"{self.nama} berkicau: Cuit cuit!"**  **def info(self):**  **return f"Burung bernama {self.nama}, jenis {self.jenis}, warna bulu {self.warna\_bulu}."**  **daftar\_hewan = []**  **for i in range(3):**  **print(f"\nInput data hewan ke-{i+1}:")**  **jenis = input("Pilih jenis hewan (kucing/anjing/burung): ")**  **if jenis == "kucing":**  **nama = input("Nama kucing: ")**  **umur = input("Umur kucing: ")**  **warna = input("Warna kucing: ")**  **kucing = Kucing(nama, umur, warna)**  **daftar\_hewan.append(kucing)**  **elif jenis == "anjing":**  **nama = input("Nama anjing: ")**  **ras = input("Ras anjing: ")**  **umur = input("Umur anjing: ")**  **anjing = Anjing(nama, ras, umur)**  **daftar\_hewan.append(anjing)**  **elif jenis == "burung":**  **nama = input("Nama burung: ")**  **jenis\_burung = input("Jenis burung: ")**  **warna = input("Warna bulu burung: ")**  **burung = Burung(nama, jenis\_burung, warna)**  **daftar\_hewan.append(burung)**  **else:**  **print("Jenis hewan tidak dikenali, lewati input ini.")**  **print("\n--- Informasi Semua Hewan ---")**  **for hewan in daftar\_hewan:**  **print(hewan.info())**  **print(hewan.suara())** |

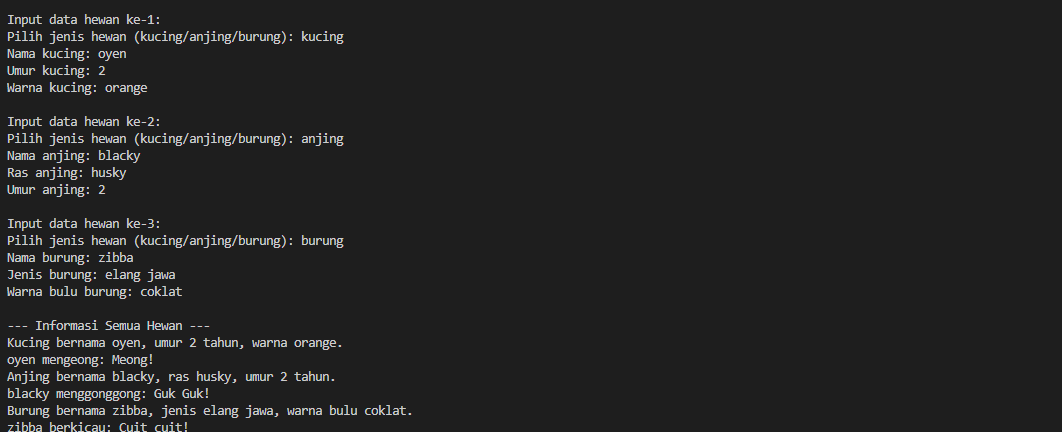
## **Hasil**

### **Hasil Soal 1**



### **Hasil Soal 2**

### **Hasil Soal 3**

****

## **Penjelasan**

### **Penjelasan Soal 1**

Pertama dibuat sebuah class bernama manusia yang merepresentasikan manusia secara umum. Di dalam class ini terdapat sebuah constructor \_\_init\_\_ yang digunakan untuk menginisialisasi atribut nama, umur, dan alamat ketika sebuah objek baru dibuat. Selain itu, ada dua metode yang mewakili perilaku manusia, yaitu berjalan() dan berlari(), yang akan mencetak aktivitas berdasarkan nama objek yang dipanggil.

Selanjutnya, kode membuat enam objek dari class manusia, masing-masing merepresentasikan individu berbeda dengan nama, umur, dan alamat yang sama (dalam contoh ini semua beralamat di Jakarta). Setelah objek dibuat, beberapa objek dipanggil metode berjalan(), sementara yang lainnya dipanggil metode berlari().

### **Penjelasan Soal 2**

Kode program yang dibuat merupakan contoh praktis penggunaan konsep class, object, dan constructor dalam program yang berinteraksi langsung dengan input pengguna. Di sini, dibuat sebuah class bernama Mahasiswa yang merepresentasikan data mahasiswa. Constructor (\_\_init\_\_) digunakan untuk menginisialisasi atribut seperti nama, nim, jurusan, dan alamat saat objek dibuat. Selain itu, terdapat metode tampilkan\_info() yang bertugas untuk mencetak informasi lengkap dari masing-masing mahasiswa.

Program kemudian meminta pengguna untuk memasukkan data tiga mahasiswa melalui perulangan for. Setiap data yang dimasukkan langsung digunakan untuk membuat objek Mahasiswa, lalu objek tersebut disimpan ke dalam list mahasiswa\_list. Setelah semua data dikumpulkan, program menampilkan seluruh informasi mahasiswa menggunakan perulangan kedua yang memanggil metode tampilkan\_info() untuk setiap objek.

### **Penjelasan Soal 3**

Program yang saya buat adalah contoh penerapan konsep OOP (Object-Oriented Programming) dalam Python, melibatkan beberapa class berbeda untuk merepresentasikan berbagai jenis hewan: Kucing, Anjing, dan Burung. Masing-masing class memiliki atribut unik yang sesuai dengan ciri khas hewan tersebut, serta metode info() untuk menampilkan deskripsi lengkap dan suara() untuk menampilkan suara khas dari hewan tersebut. Setiap class memiliki constructor (\_\_init\_\_) untuk menginisialisasi data saat objek dibuat.

Dalam bagian utama program, pengguna diminta untuk menginput data sebanyak tiga hewan. Melalui perulangan for, program akan menanyakan jenis hewan yang ingin dimasukkan, lalu meminta data spesifik sesuai jenisnya.

Jika pengguna memasukkan jenis yang dikenali (kucing, anjing, atau burung), maka akan dibuat objek dari class yang sesuai dan ditambahkan ke list daftar\_hewan. Setelah seluruh data dikumpulkan, program akan mencetak informasi lengkap dan suara dari setiap hewan dalam list tersebut dengan memanggil metode info() dan suara() secara dinamis.

# **BAB V PENUTUP**

## **Analisa**

Dari hasil praktikum, saya menganalisis bahwa class adalah kerangka atau cetakan yang digunakan untuk membuat objek dalam pemrograman berorientasi objek (OOP). Class mendefinisikan atribut dan method yang akan dimiliki oleh objek. Dengan menggunakan class, pengelolaan kode menjadi lebih terstruktur dan modular, sehingga memudahkan pengembangan dan pemeliharaan program.

Selain itu, object adalah instance dari sebuah class yang memiliki nilai atribut dan dapat menjalankan method yang telah didefinisikan dalam class. Untuk menginisialisasi objek, digunakan constructor, yaitu method khusus yang secara otomatis dipanggil saat objek dibuat. Constructor memungkinkan pemberian nilai awal pada atribut tanpa harus mengaturnya secara manual setelah objek dibuat.

## **Kesimpulan**

Dari hasil praktikum,saya menyimpulkan bahwa class, object, dan constructor adalah konsep fundamental dalam pemrograman berorientasi objek (OOP). Class berfungsi sebagai blueprint untuk membuat objek dengan atribut dan method tertentu. Object adalah instance dari class yang memiliki nilai atribut sendiri dan dapat menjalankan method sesuai dengan definisi class.

Selain itu, constructor mempermudah inisialisasi objek dengan mengatur nilai awal atribut saat objek dibuat. Dengan memahami konsep ini, pengembangan program menjadi lebih efisien dan terstruktur.

1. Class digunakan untuk mendefinisikan atribut dan method yang akan dimiliki oleh objek.
2. Object adalah hasil instansiasi dari class yang memiliki nilai atribut dan dapat menjalankan method.
3. Constructor secara otomatis menginisialisasi objek dengan nilai awal saat objek dibuat.