**LAPORAN RESMI**

**MODUL I**

**OOP  
PEMROGRAMAN BERBASIS OBJEK**



**NAMA : RIZKY YUDHA PRATAMA**

**N.R.P : 240441100082**

**DOSEN : YUDHA DWI PUTRA NEGARA S.KOM,M.KOM  
ASISTEN : AKBAR MAULANA HUSADA**

**TGL PRAKTIKUM : JUM’AT , 21 MARET 2025**

**Disetujui : 01 Oktober 2024**

**Asisten**

**AKBAR MAULANA HUSADA**

**23.044.11.00060**

**AR’RAFFI ABQORI NURAZIZI 23.04.411.00026**

**LABORATORIUM BISNIS INTELIJEN SISTEM**

**PRODI SISTEM INFORMASI**

**JURUSAN SISTEM INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# PENDAHULUAN

## Latar Belakang

Di era yang semakin maraknya teknologi ini, mempelajari Python dan pemrograman berbasis objek (OOP) menjadi sangat penting karena Python terus mendominasi sebagai salah satu bahasa pemrograman yang paling banyak digunakan dan fleksibel. Python banyak diterapkan dalam berbagai sektor, seperti pengembangan perangkat lunak, analisis data, kecerdasan buatan, dan pengembangan web. Dengan menggunakan konsep OOP, pengembang dapat menulis kode yang lebih terorganisir, efisien, dan mudah untuk dikelola, yang sangat berguna dalam proyek-proyek besar dan kompleks. Seiring dengan pesatnya perkembangan teknologi, menguasai Python dan OOP akan membuka berbagai peluang karier dan membantu para pengembang untuk menghadapi tantangan yang terus berkembang di dunia teknologi.

## Tujuan

* Mahasiswa mampu memahami konsep Class, Object dan Constructor dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.
* Mahasiswa mampu memahami atribut dan method dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.

# DASAR TEORI

## 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

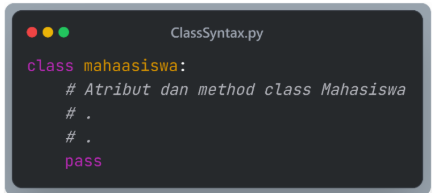
rumah, seperti jumlah kamar, warna tembok, dll. Namun, blueprint itu sendiri bukanlah rumah, melainkan hanya rancangan atau panduan untuk membangun

rumah.

### Ciri-Ciri Class

* + Mendefinisikan atribut (properti) dan method (fungsi) yang bisa dimiliki oleh objek.
  + Menjadi cetak biru blueprint untuk membuat objek.
  + Penulisan Nama pada class diawali dengan huruf kapital sesuai konvensi pemrograman Python.

### 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).

### Ciri-Ciri Object:

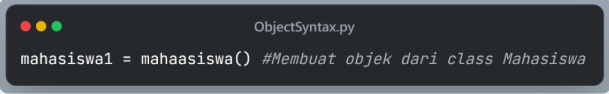
- Dibuat menggunakan class.

- Memiliki nilai konkret untuk atribut yang didefinisikan oleh class.

- Dapat memanggil methoddari class untuk berinteraksi atau memanipulasi atributnya.

### 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.

### Ciri-Ciri Constructor

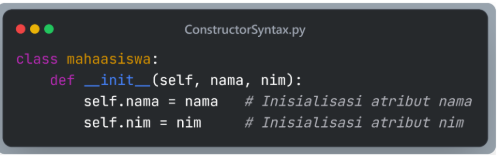
- Didefinisikan menggunakan init (self, ...).

- Secara otomatis dipanggil saat objek dibuat.

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

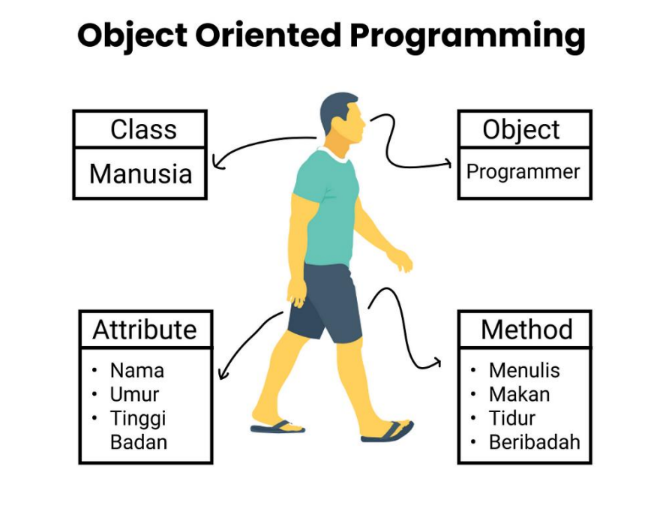
### Syntax Penulisan pada Constructor

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 efektiftentang 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:

- 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.

- 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..

- 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 :

- Nama

- Umur

- Tinggi Badan

- 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 :

- Menulis

- Makan

- Tidur

- Beribadah

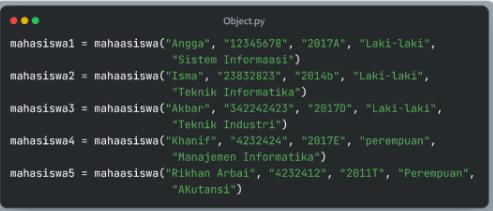
## Program Python Dalam Penerapan Class, Object, dan Constructor

Berikut merupakan program Perkenalan Mahasiswa menggunakan Class

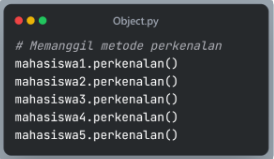
dengan bahasa Python.



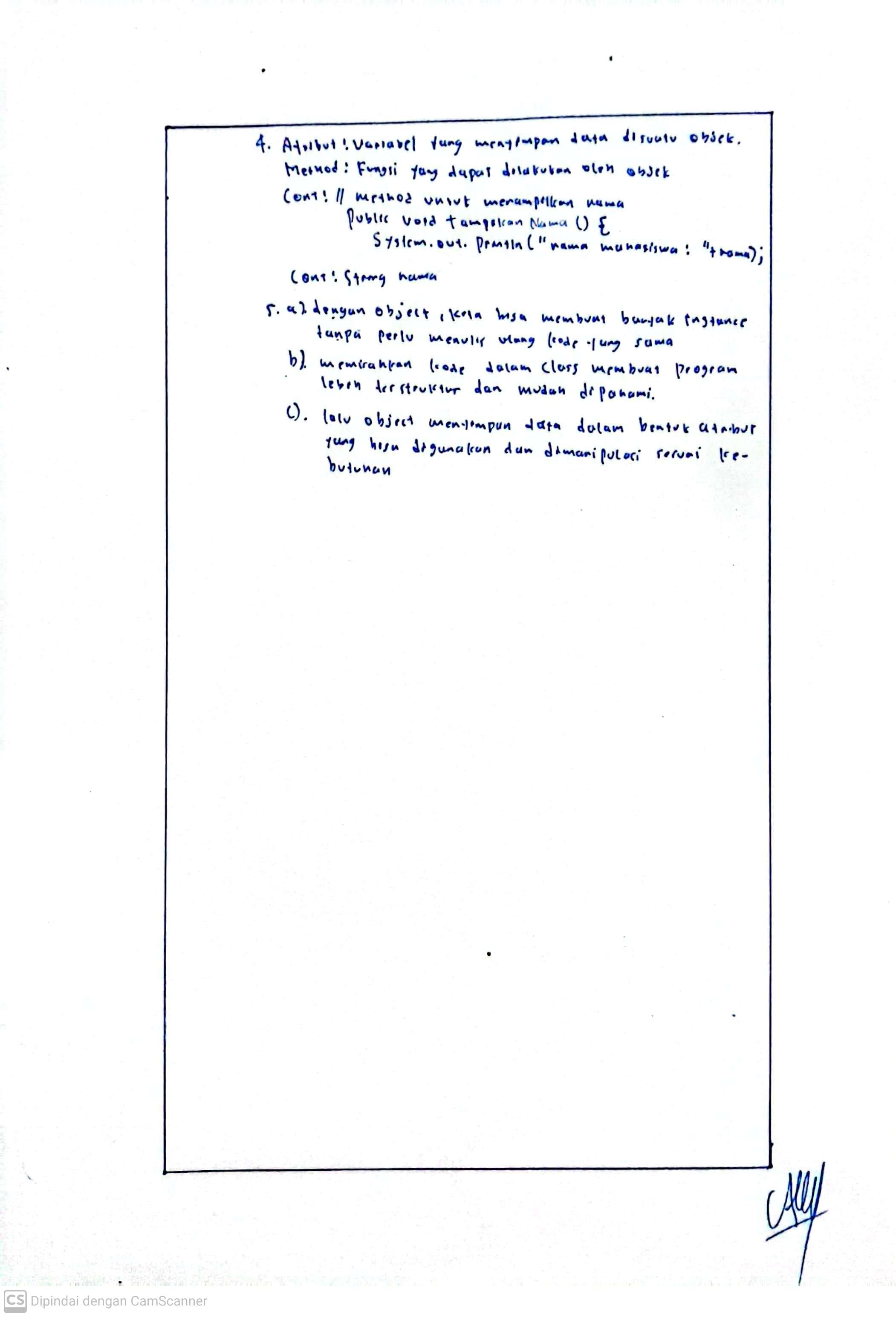
Berikut merupakan program Perkenalan Mahasiswa menggunakan Object dengan bahas Python.



Berikut merupakan program Perkenalan Mahasiswa menggunakan Constructor dengan bahas Python.



# TUGAS PENDAHULUAN



# IMPLEMENTASI

## Tugas Praktikum

### Soal 1

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

### Soal 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.

### Soal 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):

        return f"{self.nama} sedang berjalan."

    def berlari(self):

        return f"{self.nama} sedang berlari."

    # methodd untuk menampilkan

    def tampilkan\_umur\_dan\_alamat(self):

        return f"Umur {self.nama}: {self.umur} tahun, Alamat: {self.alamat}"

# membuat objeknya

manusia1 = Manusia("RAFLY", 20, "SIDOARJO")

manusia2 = Manusia("ISMA” , 20, "SIDOARJO")

manusia3 = Manusia("FAHRI", 20, "JOMBANG")

manusia4 = Manusia("YOGA", 20, “SIDOARJO")

manusia5 = Manusia("HAIDAR", 20, "SIDOARJO")

#output an nyaa

print(manusia1.berjalan())

print(manusia1.tampilkan\_umur\_dan\_alamat())

print(manusia2.berlari())

print(manusia2.tampilkan\_umur\_dan\_alamat())

print(manusia3.berjalan())

print(manusia3.tampilkan\_umur\_dan\_alamat())

print(manusia4.berlari())

print(manusia4.tampilkan\_umur\_dan\_alamat())

print(manusia5.berjalan())

print(manusia5.tampilkan\_umur\_dan\_alamat())

### 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("Nama    :", self.nama)

        print("NIM     :", self.nim)

        print("Jurusan :", self.jurusan)

        print("Alamat  :", self.alamat)

        print("-" \* 30)

mahasiswa\_list = []

# looping nyaa

for i in range(5):

    print(f"Masukkan data mahasiswa ke-{i+1}:")

    nama = input("Nama       : ")

    nim = input("NIM        : ")

    jurusan = input("Jurusan    : ")

    alamat = input("Alamat     : ")

    mahasiswa = Mahasiswa(nama, nim, jurusan, alamat)

    mahasiswa\_list.append(mahasiswa)

    print()

# Menampilkan informasi semua mahasiswa

print("Informasi Mahasiswa:")

for mahasiswa in mahasiswa\_list:

    mahasiswa.tampilkan\_info()

### Source code soal 3

class Hewan:

    def init(self, nama, jenis, habitat):

        self.nama = nama

        self.jenis = jenis

        self.habitat = habitat

    def suara(self):

        pass

    def informasi(self):

        print(f"Nama Hewan: {self.nama}")

        print(f"Jenis: {self.jenis}")

        print(f"Habitat: {self.habitat}")

        self.suara()

        print("-" \* 30)

class Kucing(Hewan):

    def init(self, nama, jenis, habitat):

        self.nama = nama

        self.jenis = jenis

        self.habitat = habitat

    def suara(self):

        print(f"{self.nama} mengatakan: Meowwwzz")

class Anjing(Hewan):

    def init(self, nama, jenis, habitat):

        self.nama = nama

        self.jenis = jenis

        self.habitat = habitat

    def suara(self):

        print(f"{self.nama} mengatakan: hruff hruff")

class Burung(Hewan):

    def init(self, nama, jenis, habitat):

        self.nama = nama

        self.jenis = jenis

        self.habitat = habitat

    def suara(self):

        print(f"{self.nama} mengatakan: crhiztt")

# Misalkan data\_hewan adalah sebuah list yang berisi data seperti ini

data\_hewan = [

    ("jakki", "Kucing Persia", "Kucing"),

    ("ryoo", "Anak anjing", "Anjing"),

    ("milli", "Burung lovebird", "Burung"),

    ("millu", "Kucing Anggoraa", "Kucing"),

    ("anap", "Anjing Darjo", "Anjing")

]

hewan\_list = []

for data in data\_hewan:

    if data[2] == "Kucing":

        hewan\_list.append(Kucing(data[0], data[1], data[2]))

    elif data[2] == "Anjing":

        hewan\_list.append(Anjing(data[0], data[1], data[2]))

    elif data[2] == "Burung":

        hewan\_list.append(Burung(data[0], data[1], data[2]))

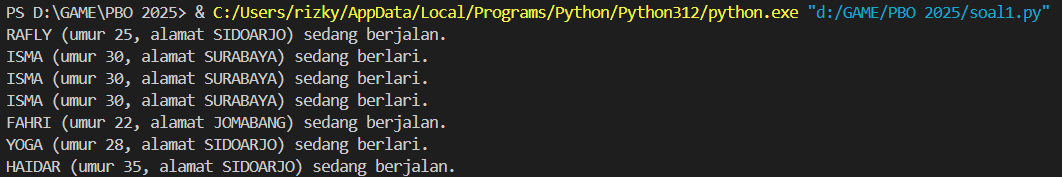
print("Informasi Hewan:")

for hewan in hewan\_list:

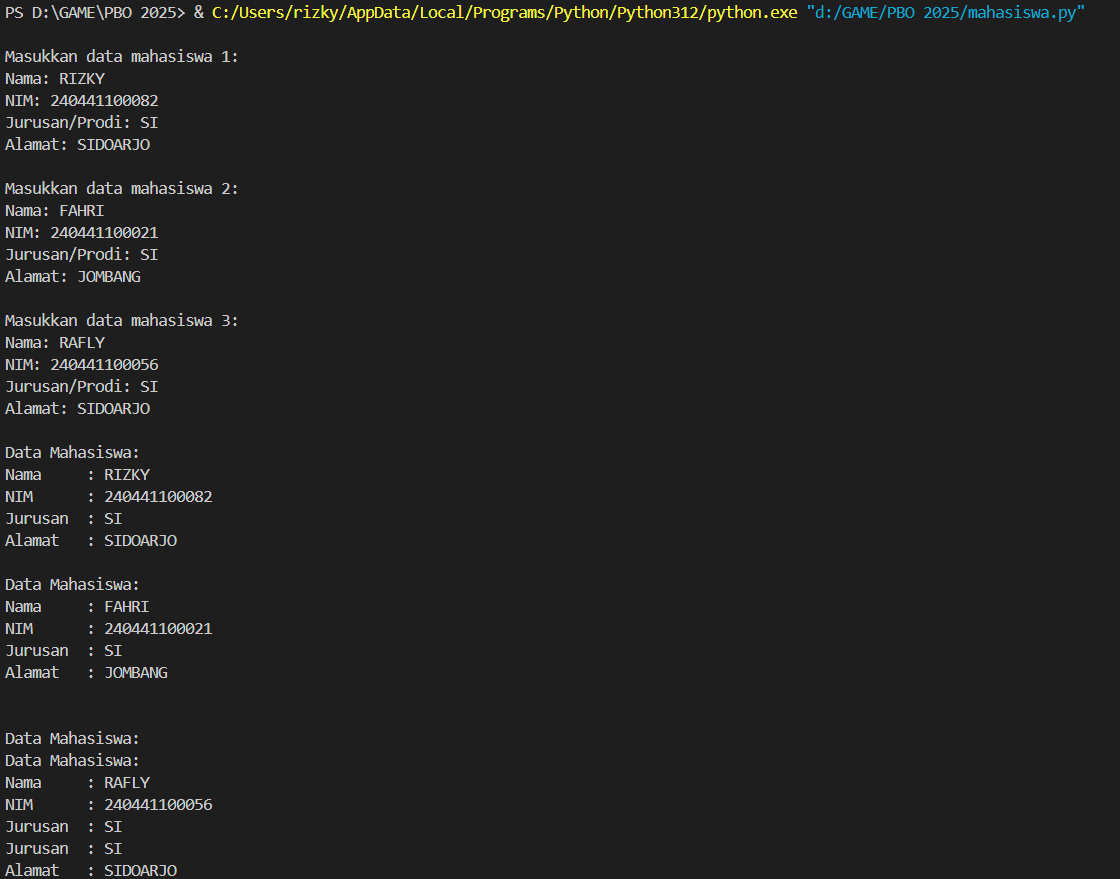
hewan.informasi()

## Hasil

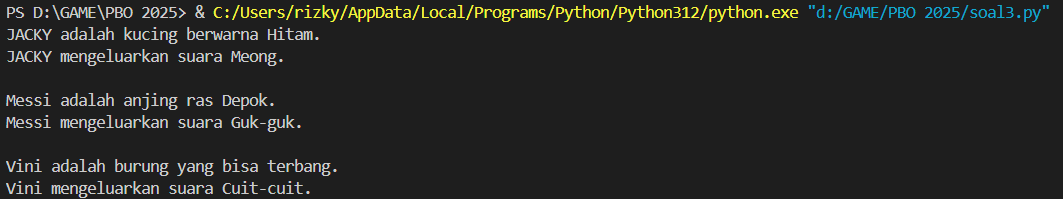
### Output soal 1



### Output soal 2



### Output soal 3



## Penjelasan

### Penjelasan soal 1

Program ini mendefinisikan kelas Manusia dengan atribut nama, umur, dan alamat, serta beberapa metode untuk menggambarkan aktivitas seperti berjalan dan berlari, serta menampilkan umur dan alamat. Lima objek Manusia dibuat, masing-masing dengan data berbeda, dan program menampilkan aktivitas serta informasi pribadi untuk setiap objek. Ini menunjukkan penerapan konsep OOP dalam pembuatan objek dan penggunaan metode untuk mengelola data.

### Penjelasan soal 2

Program ini mendefinisikan kelas Mahasiswa yang memiliki atribut nama, NIM, jurusan, dan alamat, serta metode untuk menampilkan informasi mahasiswa. Dalam program utama, data 5 mahasiswa diminta melalui input pengguna dan disimpan dalam list. Setiap data mahasiswa dimasukkan ke dalam objek Mahasiswa, yang kemudian disimpan dalam list mahasiswa\_list. Setelah itu, informasi setiap mahasiswa ditampilkan dengan memanggil metode tampilkan\_info pada setiap objek dalam list, sehingga program menampilkan data lengkap dari kelima mahasiswa tersebut.

### Penjelasan soal 3

Program ini mendefinisikan kelas Hewan dengan atribut nama, jenis, dan habitat, serta metode suara yang belum diimplementasikan di kelas induk. Kemudian, terdapat tiga kelas turunan yaitu Kucing, Anjing, dan Burung, yang masing-masing mengimplementasikan metode suara sesuai dengan suara hewan tersebut. Program meminta input data dari pengguna untuk membuat objek hewan berdasarkan jenisnya, kemudian menyimpan objek hewan tersebut dalam list hewan\_list. Setelah data hewan terkumpul, program menampilkan informasi tentang setiap hewan, termasuk suara mereka, menggunakan metode informasi dari setiap objek hewan.

# PENUTUP

## Analisa

Dari hasil praktikum, praktikan menganalisa bahwasannya praktikum ini mengajarkan penerapan konsep pemrograman berbasis objek (OOP) melalui tiga tugas utama yang saling berhubungan. Pembuatan class "Manusia" dengan atribut dan method mengajarkan cara mendefinisikan objek dan perilakunya, serta penerapan instansiasi objek. Program untuk menampilkan data mahasiswa dengan input dinamis mengajarkan penggunaan constructor dan setter untuk mengelola data yang dapat diubah oleh pengguna, serta memperkenalkan pengelolaan objek berdasarkan input. Selain itu, pembuatan class hewan dengan atribut dan method serta objek yang dibuat melalui looping memberikan pemahaman tentang konsep pewarisan dan komposisi dalam OOP, yang memungkinkan pengelompokan data dan perilaku objek secara efisien. Praktikum ini memberikan pemahaman yang kuat tentang dasar-dasar OOP dan penerapannya dalam berbagai kasus.

## Kesimpulan

Dari hasil praktikum, praktikan menyimpulkan Pemrograman berbasis objek (OOP) memberikan manfaat besar dalam pengembangan perangkat lunak dengan memungkinkan struktur kode yang lebih terorganisir, modular, dan mudah dipelihara. Dengan menggunakan konsep class dan objek, OOP memudahkan pengelolaan data dan perilaku objek, serta memungkinkan penggunaan kembali kode (reuse) yang lebih efisien melalui pewarisan dan enkapsulasi. OOP juga memfasilitasi pengembangan aplikasi yang lebih kompleks dengan membagi masalah menjadi bagian-bagian lebih kecil yang dapat diatur dan dikendalikan, sehingga meningkatkan produktivitas dan mengurangi kemungkinan kesalahan dalam pemrograman.