

MODUL 1



Disusun oleh :
Nama : Imam Ardi Perdana
Nim : 24241020
Kelas : PTIA
Prodi : FSTT

Dosen Pembimbing :
Adam Bachtiar, S.Kom., M.MT.

PROGRAM STUDI PENDIDIKAN TEKNOLOGI
FAKULTAS SAINS DAN ILMU TERAPAN (FSTT)
UNIVERSITAS MANDALIKA MATARAM
2020/2025

Aarif
12/10/25

Catatan :

- (1) Laporan sesuai
Struktur
- (2) tetapi tidak
ada garis tepi
shg tidak rapi
- (3) Judul Praktikum
Praktikum Modul - 1st

Pratikum 1 -class dan object

```
1 #Pratikum 1 - Class dan Object
2 class Hero:
3     pass
4
5 # Proses penciptaan object
6 hero_1 = Hero()
7 hero_2 = Hero()
8
9 # Memberikan atribut baru name untuk masing-masing object.
10 hero_1.name = "goku"
11 hero_2.name = "Vegeta"
12
13 # Menambahkan atribut power ke object.
14 hero_1.power = 100
15 hero_2.power = 200
16
17 # Menampilkan nilai atribut name dari kedua object.
18 print(hero_1.name)
19 print(hero_2.name)
20
21 # menampilkan atribut apa saja yang dimiliki object dalam bentuk dictionary (key = nama atribut, value = nilainya).
22 print(hero_1.__dict__)
23
```

Output

```
PS E:\algoritma\Tugas-semester-3> &
odul1/2.Class dan Object.py"
goku
Vegeta
{'name': 'goku', 'power': 100}
PS E:\algoritma\Tugas-semester-3>
```

Penjelasan: Pratikum 1 - Class Object

- Baris 1

Ini adalah komentar,

- Baris 2-3

Class Hero: digunakan untuk membuat class bernama Hero
Class bisa kamu anggap seperti cetak biru atau template
untuk membuat objek

Pas artinya "tidak ada isi dulu" - ini hanya Placeholder
agar kode tidak eror walau belum ada isi class-nya.

Baris 5

Komentar lagi. Menandakan bagian ini akan membuat object dari class Hero

Baris 6-7

Disini kita membuat dua object dari class Hero yaitu.

- Hero_1

- Hero_2

Keduanya mempunyai struktur yang sama karena berasal dari class Hero, tapi nanti bisa punya atribut dan nilai berbeda

Baris 9-11

Kita menambahkan atribut baru bernama name ke masing-masing object. Object Hero_1 sekarang punya atribut name dengan nilai "goku"

Object Hero_2 sekarang punya atribut name dengan nilai "vegeta"

Baris 13-15

Sama seperti sebelumnya, kita tambahkan atribut baru bernama power :

- Hero_1 punya Power : 100

- Hero_2 punya Power : 200

Sekarang setiap object (hero_1 dan hero_2) punya dua atribut

- name

- power

Baris 17-19

menampilkan isi dari atribut name:

- Print (hero_1.name) akan menampilkan "goku"

- Print (hero_2.name) akan menampilkan "vegeta"

Baris 21-22

- Dict - adalah atribut bawaan python yang menyimpan semua atribut milik suatu objek dalam bentuk dictionary

Jadi - Dict - Fungsinya adalah membuat daftar atribut apa saja yang dimiliki object, lengkap dengan nilainya (key: nama atribut, value: nilainya)

Praktikum 2 - Constructor `__init__()`

```
1 # Praktikum 2 - Constructor __init__()
2 class Mahasiswa:
3     def __init__(self, name, nim, prodi):
4         self.name = name
5         self.nim = nim
6         self.prodi = prodi
7
8 mhs1 = Mahasiswa("iman", "21051214005", "Informatika")
9 mhs2 = Mahasiswa("Budi", "21051214006", "Sistem Informasi")
10
11 print("Nama:", mhs1.name)
12 print("NIM:", mhs1.nim)
13 print("Prodi:", mhs1.prodi)
14 print(mhs1.__dict__)
15 print()
16 print("Nama:", mhs2.name)
17 print("NIM:", mhs2.nim)
18 print("Prodi:", mhs2.prodi)
19 print(mhs2.__dict__)
```

Output :

```
PS E:\algoritma\tugas-semester-3> & C:/Users/USER/AppData/Local/Programs/Python/Python313/python.exe
odul1/3.Constructor __init__.py"
Nama: iman
NIM: 21051214005
Prodi: Informatika
{'name': 'iman', 'nim': '21051214005', 'prodi': 'Informatika'}

Nama: Budi
NIM: 21051214006
Prodi: Sistem Informasi
{'name': 'Budi', 'nim': '21051214006', 'prodi': 'Sistem Informasi'}
PS E:\algoritma\tugas-semester-3>
```

Penjelasan:

- Baris 1
ini adalah komentar
- Baris 2
membuat sebuah class bernama mahasiswa.
Class ini seperti cetak biru untuk membuat object mahasiswa.
- Baris 3
membuat method constructor bernama `__init__`.
method ini otomatis dijalankan saat object dibuat.
Digunakan untuk mengisi data awal ke dalam object.
Metode ini menerima Parameter
 - `self` → mewakili object yang sedang dibuat
 - `name nim prodi` → data yang akan di simpan

Baris 4-6

Setiap baris ini menyimpan nilai ke atribut object:

- `self.name : name` → menyimpan nama mahasiswa.
- `self.nim : nim` → menyimpan nim.
- `self.Prodi : Prodi` → menyimpan Program studi

Jadi, setiap kali kita membuat object baru, atribut ini otomatis diisi sesuai nilai yang diberikan

Baris 8-9

membuat dua object dari class mahasiswa:

- mhs 1 Punya:

`name : "Imam"`
`nim : "21051214005"`
`Prodi : "Informatika"`

- mhs 2 Punya:

`name : "Budi"`
`nim : "21051214006"`
`Prodi : "Sistem Informasi"`

Setelah object dibuat, Python otomatis memanggil `__init__` dan mengisi atribut dengan nilai yang diberikan.

Baris 11-13

menampilkan isi atribut milik object mhs 1

Baris 14

menampilkan semua atribut milik object mhs 1 dalam bentuk dictionary (key: nama atribut, value: nilainya).

Baris 15

Hanya untuk membuat baris kosong di terminal agar output lebih rapi.

Baris 16-18

menampilkan atribut object ~~at~~ mhs 2.

Baris 19

menampilkan atribut object mhs 2 dalam bentuk dictionary

PENJELASAN TAMBAHAN

Apakah `__init__`?

- `__init__` adalah method khusus (constructor) di Python
- Dipanggil otomatis setiap kali object dibuat dari class
- Fungsinya untuk mengisi atribut milik object (bukan variabel lokal)
- misalnya

`self.name = name`

berarti: buat atribut bernama `name` di dalam object, dan isi dengan dari parameter `name`.

Kalau tanpa `self`, maka variabel hanya hidup di dalam function saja dan tidak disimpan di object

Apakah `__dict__`?

- `__dict__` adalah atribut bawaan Python yang menyimpan semua atribut object dalam bentuk dictionary

Praktikum 3 - Class dan Instance Attribute

```
1 # Praktikum 3 - Class dan Instance Attribute
2 class Mahasiswa:
3
4     jumlah_mahasiswa = 0
5
6     def __init__(self, name, nim, prodi):
7         self.name = name
8         self.nim = nim
9         self.prodi = prodi
10        Mahasiswa.jumlah_mahasiswa += 1
11        print("Membuat Object Mahasiswa dengan nama " + self.name)
12
13 mhs1 = Mahasiswa("Adam", "21051214005", "Teknik Informatika")
14 print("Total Mahasiswa : " + str(mhs1.jumlah_mahasiswa))
15 mhs2 = Mahasiswa("Budi", "21051214006", "Teknik Industri")
16 print("Total Mahasiswa : " + str(mhs2.jumlah_mahasiswa))
17
18 print()
19 print("Total Mahasiswa : " + str(Mahasiswa.jumlah_mahasiswa))
20
```

Output:

```
PS E:\algoritma\tugas-semester-3> & C:/Users/USER/AppData/Local/Programs/Python/Python313/python.exe
odul1/4. Class dan Instance Attribute.py"
Membuat Object Mahasiswa dengan nama Adam
Total Mahasiswa : 1
Membuat Object Mahasiswa dengan nama Budi
Total Mahasiswa : 2

Total Mahasiswa : 2
PS E:\algoritma\tugas-semester-3>
```

Penjelasan

- Baris 1
Ini adalah komentar
- Baris 2
membuat class bernama mahasiswa. Class ini berfungsi sebagai template untuk membuat object mahasiswa.
- Baris 4
ini adalah class atribut
 - Miliknya milik bersama semua object
 - Di simpan di class. bukan di masing-masing object
 - Bisa diakses melalui :
 - mahasiswa.jumlah_mahasiswa (cara utama)
 - atau object.jumlah_mahasiswa (cara alternatif)
- Nilai awalnya adalah 0
- Baris 6

membuat constructor `-init-()`, yang otomatis dijalankan setiap kali membuat object baru.

menerima Parameter

- `Self` → mengacu pada object yang sedang dibuat
- `name, nim, Prodi` → data yang akan & disimpan ke object

Baris 7-9

Baris ini membuat instance attribute, yaitu:

- `Self.name`
- `Self.nim`
- `Self.Prodi`

instance attribute adalah data yang spesifik untuk setiap object.
Jadi mhs 1 dan mhs 2 punya nilai berbeda.

Baris 10

setiap kali object baru dibuat, nilai class attribute `Jumlah - mahasiswa` akan bertambah 1. Ini artinya setiap kali `-init-()` dipanggil, jumlah mahasiswa bertambah.

Baris 11

menampilkan Pesan Setiap kali object baru dibuat

Contoh output:

"membuat object mahasiswa dengan nama Adam"

Baris 13

membuat object pertama bernama mahasiswa 1.

Ketika baris ini dieksekusi:

- `-init-()` dijalankan
- `name, nim, Prodi` disimpan ke object
- `Jumlah - mahasiswa` naik menjadi 1

PENJELASAN ISTILAH:

Jenis attribute contoh	Penjelasan
Class attribute <code>Jumlah - mahasiswa</code>	Dimiliki bersama oleh semua object nilainya sama di seluruh class
Instance Attribute <code>Self.name, Self.nim, Self.Prodi</code>	Dimiliki oleh masing-masing object, nilainya bisa beda

PERBEDAAN:

mhs 1. `Jumlah - mahasiswa`

- mengakses class attribute ~~langsung dari kelas~~ melalui object mhs 1.
- Python akan mencari di object dulu, kalau tidak ada baru ke class
- Jadi nilainya sama seperti `jumlah - mahasiswa`

mahasiswa `Jumlah - mahasiswa`

- mengakses class attribute langsung dari kelas
- ini cara yang lebih disarankan karena lebih jelas bahwa itu atribut milik class

Keduanya menampilkan nilai yang sama, tapi cara aksesnya berbeda

- `jumlah - mahasiswa` → lebih jelas dan direkomendasikan
- `mhs 1.jumlah - mahasiswa` → bisa juga, tapi kurang disarankan

Praktikum 4 : Instance Method

```
1 class Hero:
2     # class variabel
3     jumlah = 0
4
5     def __init__(self, inputName, inputHealth, inputPower, inputArmor):
6         # instance variabel
7         self.name = inputName
8         self.health = inputHealth
9         self.power = inputPower
10        self.armor = inputArmor
11        Hero.jumlah += 1
12
13    def siapa(self):
14        return f"{self.name} memiliki power {self.power}"
15
16    def healthUp(self, up):
17        self.health += up
18
19    def gethealth(self):
20        return self.health
21
22    # method baru dengan parameter dan return
23    def attack(self, musuh):
24        damage = self.power - musuh.armor
25        if damage < 0:
26            damage = 0 # supaya tidak negatif
27        musuh.health -= damage
28        return f"{self.name} menyerang {musuh.name} dengan damage {damage}"
29
30    # Membuat objek
31    hero1 = Hero("Sniper", 100, 10, 5)
32    hero2 = Hero("Axe", 200, 15, 0)
33
34    # Memakai method
35    print(hero1.siapa())
36    hero1.armor = 10
37    print(f"A armor {hero1.name} = {hero1.armor}")
38    print(f"Jumlah hero = {Hero.jumlah}")
39    hero1.healthUp(20)
40    print(f"Health {hero1.name} = {hero1.gethealth()}")
41    print(f"Health {hero2.name} = {hero2.gethealth()}")
42
43    # Coba serangan
44    print(hero1.attack(hero2))
45    print(f"Health {hero2.name} sekarang = {hero2.gethealth()}")
46
```

Output:

```
PS E:\algoritma\tugas-semester-3> & C:/Users/USER/AppData/Local/Programs/Python/Python313/python.exe
odu11/5. Method.py
Sniper memiliki power 10
Armor Sniper = 10
Jumlah hero = 2
Health Sniper = 120
Health Axe = 200
Sniper menyerang Axe dengan damage 10
Health Axe sekarang = 190
PS E:\algoritma\tugas-semester-3>
```

Penjelasan:

- Baris 1
membuat class bernama Hero
- Baris 2
Class Variable
Komentar Penelus
- Baris 3
Jumlah adalah class variable, milik bersama semua object. Fungsinya untuk menghitung berapa banyak hero yang dibuat
- Baris 5
-init- adalah fungsi khusus yang otomatis dijalankan saat kamu membuat hero baru. Fungsinya ini seperti konstruktor.
- Baris 6-10
ini adalah instance variable. Artinya setiap hero punya data sendiri-sendiri :
 - self.name → nama hero
 - self.health → darah hero
 - self.power → kekuatan serang
 - self.armor → kekuatan pertahanan
- Baris 11
setiap kali hero baru dibuat, Hero.jumlah akan bertambah 1.
- Baris 13-14
method ini menambah darah (health) sebanyak nilai up.
contoh : Hero 1.health up (20) → darah bertambah 20
- Baris 19-20
method ini mengembalikan (return) jumlah darah saat ini
- Baris 22
method baru dengan parameter dan return komentar penanda method berikutnya
- Baris 23-28
method ini untuk menyerang hero lain. menerima parameter musuh (yang juga object hero).
 - menghitung damage : self.power - musuh.armor
 - jika damage negatif, di-set 0 (agar tidak menjadi Healing)
 - mengurangi musuh.health sebesar damage (side-effect)
 - lalu return string yang menjelaskan hasil serangan
- Baris 30
membuat objek komentar.
- Baris 31-32
hero 1 = Hero(....) : hero 2 = Hero(....), membuat object Hero. saat dibuat, -init- dipanggil dan Hero.jumlah bertambah tiap kali
- Baris 34
memulai method. komentar
- Baris 35
menampilkan berapa hero 1. dan berapa powernya
- Baris 36
mengurangi nilai armor hero 1 menjadi 10

TUGAS PRAKTIKUM MODUL 1

```
1 # TUGAS PRAKTIKUM MODUL 1
2 class PersegiPanjang:
3     # Constructor dengan parameter panjang dan lebar
4     def __init__(self, panjang, lebar):
5         self.panjang = panjang
6         self.lebar = lebar
7
8     # Method untuk menghitung luas
9     def hitung_luas(self):
10         return self.panjang * self.lebar
11
12     # Method untuk menghitung keliling
13     def hitung_keliling(self):
14         return 2 * (self.panjang + self.lebar)
15
16
17 # Membuat object dari class PersegiPanjang
18 pp1 = PersegiPanjang(10, 5)
19 pp2 = PersegiPanjang(7, 3)
20
21 # Menampilkan hasil perhitungan
22 print("Persegi Panjang 1")
23 print("Panjang:", pp1.panjang)
24 print("Lebar:", pp1.lebar)
25 print("Luas:", pp1.hitung_luas())
26 print("Keliling:", pp1.hitung_keliling())
27
28 print("\nPersegi Panjang 2")
29 print("Panjang:", pp2.panjang)
30 print("Lebar:", pp2.lebar)
31 print("Luas:", pp2.hitung_luas())
32 print("Keliling:", pp2.hitung_keliling())
```

Output:

```
PS E:\algoritma\Tugas-semester-3> & C:/Users/USER/AppData/Local/
odul1/TUGAS PRAKTIKUM MODUL 1.py"
Persegi Panjang 1
Panjang: 10
Lebar: 5
Luas: 50
Keliling: 30

Persegi Panjang 2
Panjang: 7
Lebar: 3
Luas: 21
Keliling: 20
PS E:\algoritma\Tugas-semester-3>
```

Penjelasan:

Baris 1

ini adalah komentar. Fungsinya hanya untuk memberi keterangan Pakai kode. Tidak dijalankan oleh Program

Baris 2

membuat class bernama Persegi Panjang. Class adalah cetaklan/blueprint untuk membuat object. Disini kelas ini akan menyimpan data Panjang, lebar, serta rumus luas dan keliling.

Baris 4-6

ini adalah constructor (-init-) yaitu fungsinya akan otomatis jalan saat kita buat objek dari class.

- Self : mewakili objek itu sendiri
- Panjang dan lebar : Parameter yang kita kirim saat membuat object
- Self.Panjang dan Self.lebar : instance attribute, artinya milik masing-masing objek

Contoh :

Kalau kita buat PP_1 = Persegi Panjang (10,5)

maka PP_1 .Panjang = 10 dan PP_1 .lebar = 5

Baris 9-10

method (Fungsinya dalam class) untuk menghitung luas Persegi Panjang.

Rumus: Panjang x lebar

Hasil dicembarkan Pakai return.

Baris 13-14

method untuk menghitung keliling persegi panjang. Rumus: $2 \times (\text{Panjang} + \text{lebar})$

Baris 17-18

membuat 2 objek dari class:

- PP_1 dengan Panjang 10 dan lebar 5
- PP_2 dengan Panjang 7 dan lebar 3

Setiap objek punya data sendiri

Baris 21-26

menampilkan hasil dari objek PP_1 :

- cetak nilai Panjang dan lebar
- Panggil fungsi ~~hitung~~ - luas () → tampilkan hasilnya
- Panggil fungsi ~~hitung~~ - keliling () → tampilkan hasilnya

sama seperti di atas, tapi untuk objek PP_2 /n artinya baris baru supaya tampil rapi

Kesimpulan

- class PersegiPanjang: → membuat cetaklan objek
- -init-() → mengisi data awal (Panjang dan lebar)
- Self.Panjang, Self.lebar → instance attribute (milik tiap objek)
- hitung_luas () & hitung_keliling () → method (fungsi dalam class)
- PP_1, PP_2 → objek (bentuk nyata dari class)