

**LAPORAN PRAKTIKUM PEMOGRAMAN  
BERORIENTASI OBJEK (PBO)  
PRAKTIKUM 2**

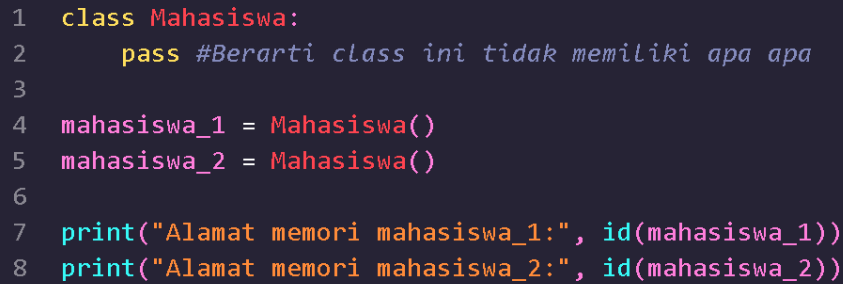


**2411102441249**

**HERVINO ISLAMI FASHA**

**FAKULTAS SAINS DAN TEKNOLOGI  
PROGRAM STUDI S1 TEKNIK INFORMATIKA  
UNIVERSITAS MUHAMMADIYAH KALIMANTAN  
TIMUR**

## 1. Membuat Class Sederhana



```

1 class Mahasiswa:
2     pass #Berarti class ini tidak memiliki apa apa
3
4 mahasiswa_1 = Mahasiswa()
5 mahasiswa_2 = Mahasiswa()
6
7 print("Alamat memori mahasiswa_1:", id(mahasiswa_1))
8 print("Alamat memori mahasiswa_2:", id(mahasiswa_2))

```

## A. Deklarasi class kosong

```
class Mahasiswa:
```

```
    pass # Berarti class ini tidak memiliki apa apa
```

- class Mahasiswa: Membuat sebuah class dengan nama **Mahasiswa**
- pass: class ini belum memiliki atribut atau method apapun. Jadi class hanya "kosong".

## B. Membuat objek (Instansiasi Class)

```
mahasiswa_1 = Mahasiswa()
```

```
mahasiswa_2 = Mahasiswa()
```

- mahasiswa\_1: adalah objek pertama dari class **Mahasiswa**
- mahasiswa\_2: adalah objek kedua dari class **Mahasiswa**
- Setiap kali membuat objek dengan **Mahasiswa()**, Python akan memberikan Alamat memori (ID) yang berbeda.

## C. Menampilkan Alamat memori objek

```
print("Alamat memori mahasiswa_1:", id(mahasiswa_1))
```

```
print("Alamat memori mahasiswa_2:", id(mahasiswa_2))
```

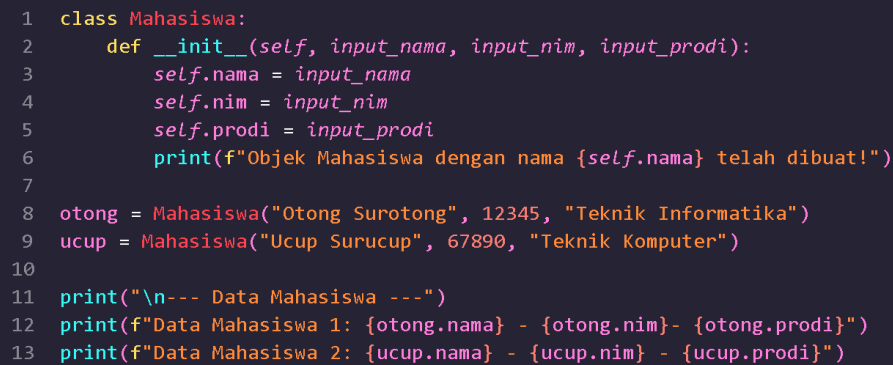
- Fungsi **id()** digunakan untuk mendapatkan Alamat memori (unik) dari objek.

- Karena **mahasiswa\_1** dan **mahasiswa\_2** adalah dua objek yang berbeda, maka **id**-nya juga berbeda.

OUTPUT:

```
D:\smt 3\PB0\P2>C:/Python313/python.exe "d:/smt 3/PB0/P2/Latihan_P2.py"
Alamat memori mahasiswa_1: 2491966122576
Alamat memori mahasiswa_2: 2491968801680
```

## 2. Menambahkan Atribut dengan Constructor `__init__`



```
1 class Mahasiswa:
2     def __init__(self, input_nama, input_nim, input_prodi):
3         self.nama = input_nama
4         self.nim = input_nim
5         self.prodi = input_prodi
6         print(f"Objek Mahasiswa dengan nama {self.nama} telah dibuat!")
7
8 otong = Mahasiswa("Otong Surotong", 12345, "Teknik Informatika")
9 ucup = Mahasiswa("Ucup Surucup", 67890, "Teknik Komputer")
10
11 print("\n--- Data Mahasiswa ---")
12 print(f>Data Mahasiswa 1: {otong.nama} - {otong.nim}- {otong.prodi}")
13 print(f>Data Mahasiswa 2: {ucup.nama} - {ucup.nim} - {ucup.prodi}")
```

### A. Membuat class Mahasiswa

class Mahasiswa:

```
def __init__(self, input_nama, input_nim, input_prodi):
    self.nama = input_nama
    self.nim = input_nim
    self.prodi = input_prodi
    print(f'Objek Mahasiswa dengan nama {self.nama} telah dibuat!')
```

- **`__init__`** : Constructor akan dipanggil otomatis setiap kali membuat objek baru dari class.
- **`self.nama`, `self.nim`, dan `self.prodi`** : Atribut yang disimpan di objek.

- **print(...)** : Menampilkan pesan setiap kali objek baru dibuat.

B. Membuat objek mahasiswa

```
otong = Mahasiswa("Otong Surotong", 12345, "Teknik Informatika")
```

```
ucup = Mahasiswa("Ucup Surucup", 67890, "Teknik Komputer")
```

- Membuat objek **otong** dengan data nama, NIM, dan prodi.
- Membuat objek **ucup** dengan data berbeda.
- Saat objek dibuat, constructor (**\_\_init\_\_**) akan otomatis mencetak pesan.

C. Menampilkan data mahasiswa

```
print("\n--- Data Mahasiswa ---")
```

```
print(f'Data Mahasiswa 1: {otong.nama} - {otong.nim} - {otong.prodi}')
```

```
print(f'Data Mahasiswa 2: {ucup.nama} - {ucup.nim} - {ucup.prodi}')
```

- Mengambil data dari atribut masing-masing objek.
- Menggunakan f-string untuk format teks.

OUTPUT:

```
D:\smt 3\PB0\P2>C:/Python313/python.exe "d:/smt 3/PB0/P2/Latihan_P2.py"
Objek Mahasiswa dengan nama Otong Surotong telah dibuat!
Objek Mahasiswa dengan nama Ucup Surucup telah dibuat!

--- Data Mahasiswa ---
Data Mahasiswa 1: Otong Surotong - 12345- Teknik Informatika
Data Mahasiswa 2: Ucup Surucup - 67890 - Teknik Komputer
```

### 3. Menambahkan perilaku dengan Method

```

1  class Mahasiswa:
2      def __init__(self, input_nama, input_nim, input_prodi):
3          self.nama = input_nama
4          self.nim = input_nim
5          self.prodi = input_prodi
6          print(f"Objek Mahasiswa dengan nama {self.nama} telah dibuat!")
7
8      def sapa(self):
9          print(f"Halo! Nama saya {self.nama}. Senang berkenalan!")
10
11     def tampilkan_info(self):
12         print("--- Informasi Mahasiswa ---")
13         print(f>Nama      : {self.nama}")
14         print(f"NIM       : {self.nim}")
15         print(f"Prodi      : {self.prodi}")
16         print("-----")
17
18     otong = Mahasiswa("Otong Surotong", 12345, "Teknik Informatika")
19     ucup = Mahasiswa("Ucup Surucup", 67890, "Teknik Komputer")
20
21     print("\n--- Interaksi dengan objek ---")
22     otong.sapa()
23     ucup.sapa()
24
25     print("\n--- Data Mahasiswa ---")
26     print(f>Data Mahasiswa 1: {otong.nama} - {otong.nim} - {otong.prodi}")
27     print(f>Data Mahasiswa 2: {ucup.nama} - {ucup.nim} - {ucup.prodi}")

```

#### A. Deklarasi class Mahasiswa

class Mahasiswa:

```

def __init__(self, input_nama, input_nim, input_prodi):
    self.nama = input_nama
    self.nim = input_nim
    self.prodi = input_prodi
    print(f"Objek Mahasiswa dengan nama {self.nama} telah dibuat!")

```

- **\_\_init\_\_** : constructor, otomatis dijalankan saat objek dibuat.
- **self.nama, self.nim, dan self.prodi** : adalah atribut yang menyimpan data mahasiswa.
- **print(...)** : menandakan objek berhasil dibuat.

#### B. Method Sapa

```

def sapa(self):
    print(f"Halo! Nama saya {self.nama}. Senang berkenalan!")

```

- Digunakan untuk menyapa dengan nama mahasiswa
- Memanfaatkan atribut **self.nama**

#### C. Method **tampilkan\_info**



```
def tampilkan_info(self):
    print("--- Informasi Mahasiswa ---")
    print(f>Nama : {self.nama}")
    print(f"NIM : {self.nim}")
    print(f"Prodi : {self.prodi}")
    print("-----")
    - Menampilkan informasi lengkap mahasiswa
    - Format dibuat rapi agar mudah dibaca
```

#### D. Membuat objek

```
otong = Mahasiswa("Otong Surotong", 12345, "Teknik Informatika")
ucup = Mahasiswa("Ucup Surucup", 67890, "Teknik Komputer")
    - Membuat otong dan ucup dari class Mahasiswa.
    - Constructor otomatis dijalankan, sehingga muncul pesan bahwa objek berhasil dibuat.
```

#### E. Interaksi dengan objek

```
otong.sapa()
ucup.sapa()
    - Memanggil method sapa() untuk masing-masing objek.
```

#### F. Menampilkan data mahasiswa

```
print(f'Data Mahasiswa 1: {otong.nama} - {otong.nim} - {otong.prodi}')
print(f'Data Mahasiswa 2: {ucup.nama} - {ucup.nim} - {ucup.prodi}')
    - Mengakses atribut secara langsung dari objek.
```

Output:

```
D:\smt 3\PB0\P2>C:/Python313/python.exe "d:/smt 3/PB0/P2/Latihan_P2.py"
Objek Mahasiswa dengan nama Otong Surotong telah dibuat!
Objek Mahasiswa dengan nama Ucup Surucup telah dibuat!

--- Interaksi dengan objek ---
Halo! Nama saya Otong Surotong. Senang berkenalan!
Halo! Nama saya Ucup Surucup. Senang berkenalan!

--- Data Mahasiswa ---
Data Mahasiswa 1: Otong Surotong - 12345- Teknik Informatika
Data Mahasiswa 2: Ucup Surucup - 67890 - Teknik Komputer
```

## TUGAS

## A. Code

```

1  class Buku:
2      def __init__(self, judul, penulis, tahun_terbit):
3          self.judul = judul
4          self.penulis = penulis
5          self.tahun_terbit = tahun_terbit
6          self.status_pinjam = False
7
8      def tampilkan_info(self):
9          status = "Dipinjam" if self.status_pinjam else "Tersedia"
10         print(f"Judul      : {self.judul}")
11         print(f"Penulis     : {self.judul}")
12         print(f"Tahun Terbit: {self.tahun_terbit}")
13         print(f"Status      : {status}")
14         print("_" * 40)
15
16     def pinjam_buku(self):
17         if not self.status_pinjam:
18             self.status_pinjam = True
19             print(f"Buku '{self.judul}' telah dipinjam.")
20         else:
21             print(f"Buku '{self.judul}' sudah dalam status pinjam.")
22
23     def kembalikan_buku(self):
24         if self.status_pinjam:
25             self.status_pinjam = False
26             print(f"Buku '{self.judul}' telah dikembalikan.")
27         else:
28             print(f"Buku '{self.judul}' memang sudah tersedia.")
29
30 if __name__ == "__main__":
31     buku1 = Buku("Laut Bercerita", "Leila S. Chudori", 2017)
32     buku2 = Buku("Madilog", "Tan Malaka", 1943)
33
34     print("Informasi Awal Buku:")
35     buku1.tampilkan_info()
36     buku2.tampilkan_info()
37
38     print("Simulasi Peminjaman:")
39     buku1.pinjam_buku()
40     buku1.tampilkan_info()
41
42     print("Simulasi Pengembalian:")
43     buku1.kembalikan_buku()
44     buku1.tampilkan_info()

```

## B. Output

```

D:\smt 3\PB0\P2>C:/Python313/python.exe "d:/smt 3/PB0/P2/P2.py"
Informasi Awal Buku:
Judul      : Laut Bercerita
Penulis    : Laut Bercerita
Tahun Terbit: 2017
Status     : Tersedia

Judul      : Madilog
Penulis    : Madilog
Tahun Terbit: 1943
Status     : Tersedia

Simulasi Peminjaman:
Buku 'Laut Bercerita' telah dipinjam.
Judul      : Laut Bercerita
Penulis    : Laut Bercerita
Tahun Terbit: 2017
Status     : Dipinjam

Simulasi Pengembalian:
Buku 'Laut Bercerita' telah dikembalikan.
Judul      : Laut Bercerita
Penulis    : Laut Bercerita
Tahun Terbit: 2017
Status     : Tersedia

```

## C. Refleksi

Dengan menggunakan class, pendataan buku jadi lebih terstruktur karena setiap buku yg memiliki atribut dan perilaku yg jelas. Hal ini lebih baik dibanfingkan hanya memakai variabel biasa, Karena class memudahkan pengelolah data dalam skala besar serta meningkatkan reusabilitas kode