

Laporan Praktikum Algoritma Dan Struktur Data Python

(Class, Object And Array)

Modul I – Algoritma & Struktur Data

Muhammad Arlianto/22103001009

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

Tanggal praktikum: 5 Oktober 2023

Arlianto9999@gmail.com

Teknik Komputer

Institut Teknologi Dan Sains Nahdatul Ulama Kalimantan

Abstrak—Praktikum ini bertujuan untuk memahami konsep *class*, *object*, dan penggunaan array dalam pemrograman Python. Pemahaman tentang konsep dasar ini sangat penting karena membentuk dasar pemrograman berorientasi *object* (OOP) dan pengelolaan data yang efisien dalam Python. Selama praktikum ini, peserta akan mempelajari cara mendefinisikan *class*, menginstansiasi *object*, dan mengakses atribut dan metode yang terkait dengan *object* tersebut. Peserta juga akan belajar tentang konsep pewarisan dan penggunaan *class* dan *object* dalam pemrograman sehari-hari. Selain itu, praktikum ini juga akan mengajarkan peserta tentang penggunaan array dalam Python. Peserta akan memahami bagaimana array dapat digunakan untuk menyimpan, mengakses, dan memanipulasi data dalam struktur yang terstruktur. Melalui praktikum ini, peserta diharapkan dapat mengembangkan pemahaman yang kuat tentang OOP dan pengelolaan data dalam Python. Keterampilan ini sangat relevan dalam pengembangan perangkat lunak dan pemecahan masalah yang kompleks. Praktikum ini juga akan memberikan dasar yang kuat bagi peserta dalam pengembangan aplikasi Python yang lebih efisien dan mudah dipelihara. Praktikum ini merupakan langkah awal yang penting dalam memahami Python, sebuah bahasa pemrograman yang serbaguna dan kuat. Dengan pemahaman konsep dasar ini, peserta akan siap untuk menjelajahi berbagai bidang pengembangan perangkat lunak yang melibatkan Python.

Kata kunci—*Class, Object And Array Python; format; Python; Class, Object And Array Python*

I. PENDAHULUAN

Pemrograman Python adalah salah satu bahasa pemrograman yang paling populer dan serbaguna digunakan di berbagai bidang, mulai dari pengembangan web hingga ilmu data. Dalam pemrograman berorientasi *object* (OOP) Python, konsep dasar yang penting untuk dipahami adalah *class*, *object*, dan array. *Class* digunakan untuk mendefinisikan entitas yang memiliki atribut dan metode yang terkait. *Object* adalah instansi dari *class* dengan sifat dan perilaku yang sama. Sedangkan array adalah tipe data yang memungkinkan kita untuk menyimpan, mengelola, dan memanipulasi data dalam struktur yang terorganisir[1].

Dalam laporan praktikum ini, kami akan membahas konsep dasar *class*, *object*, dan array dalam pemrograman Python. Praktikum ini dirancang untuk memberikan pemahaman yang

kuat tentang cara mendefinisikan *class*, membuat *object*, dan mengelola data dengan menggunakan array. Praktikum ini juga akan membantu memahami pentingnya OOP dalam pengembangan perangkat lunak dan bagaimana struktur data seperti array dapat memudahkan pengelolaan data

Melalui praktikum ini, diharapkan akan dapat merancang dan mengimplementasikan *class* yang sesuai untuk menyelesaikan permasalahan yang diberikan, membuat *object* dari *class* tersebut, dan menggunakan array untuk mengelola data. Pemahaman ini akan membantu dalam pengembangan perangkat lunak yang lebih efisien dan memudahkan pemeliharaan *code*[2].

II. HASIL DAN ANALISIS

A. Classs

Class sebagai tempat dimana semua function *code* berlangsung. Sehingga akan melakukan inisiasi terhadap perintah yang diberikan. Berikut adalah beberapa class yang dibuat dalam praktikum ini:

1. Class Mobil

```
1 class Mobil :
2     def __init__(self, merk, model, production, price):
3         self.merk = merk
4         self.model = model
5         self.production = production
6         self.price = price
7     def __str__(self):
8         return f'{self.merk}, {self.model}, {self.production}, {self.price}'
```

Gambar 2.1.1 Class Mobil

a. Definisi Class

Code pertama (*class Mobil*;) adalah definisi *class*. Ini berarti kita membuat suatu entitas yang disebut "Mobil." *Class* ini akan

memiliki atribut dan metode tertentu yang mendefinisikan perilakunya.

b. Konstruktor `__init__`

Metode `__init__` adalah konstruktor *class*. Ketika membuat *object* dari *class* ini, metode ini akan dijalankan. Ini digunakan untuk menginisialisasi atribut-atribut *object*. Dalam hal ini, konstruktor menerima empat argumen: merk, model, production, dan price. Kemudian, atribut-atribut *object* seperti `self.merk`, `self.model`, `self.production` dan `self.price` akan diinisialisasi dengan nilai yang sesuai.

c. Metode `__str__`

Ini adalah metode bawaan yang digunakan untuk mengonversi *object* menjadi string. Ketika mencetak *object* Mobil, metode `__str__` akan dipanggil. Dalam *code* ini, metode ini mengembalikan string yang berisi nilai dari atribut-atribut *object* seperti `merk`, `model`, `production`, dan `price`.

2. Class Mahasiswa

```
1 class Mahasiswa:
2     def __init__(self, name, age, department):
3         self.name = name
4         self.age = age
5         self.department = department
6     def __str__(self):
7         return f"{self.name}, {self.age}, {self.department}"
```

Gambar 2.1.2 Class Mahasiswa

a. Definisi Class

Kode pertama (class Mahasiswa:) adalah definisi *class*. Ini berarti kita membuat suatu entitas yang disebut "Mahasiswa." *Class* ini akan memiliki atribut dan metode tertentu yang mendefinisikan perilakunya.

b. Konstruktor `__init__`

Metode `__init__` adalah konstruktor *class*. Ketika membuat objek dari *class* ini, metode ini

akan dijalankan. Tujuan dari konstruktor ini adalah untuk menginisialisasi atribut-atribut objek. Dalam hal ini, konstruktor menerima tiga argumen: name (nama mahasiswa), age (usia mahasiswa), dan department (jurusan mahasiswa). Kemudian, atribut-atribut objek seperti `self.name`, `self.age`, dan `self.department` akan diinisialisasi dengan nilai yang sesuai.

c. Metode `__str__`

Ini adalah metode bawaan yang digunakan untuk mengonversi objek menjadi string. Ketika mencetak objek Mahasiswa, metode `__str__` akan dipanggil. Dalam kode ini, metode ini mengembalikan string yang berisi nilai dari atribut-atribut objek seperti name, age, dan department.

3. Class Buku

```
1 class Buku :
2     def __init__(self, title, writer, yearpublish, price):
3         self.title = title
4         self.writer = writer
5         self.yearpublish = yearpublish
6         self.price = price
7     def __str__(self):
8         return f"{self.title}, {self.writer}, {self.yearpublish}, {self.price}"
```

Gambar 2.1.3 Class Buku

a. Definisi Class

Kode pertama (class Buku:) adalah definisi *class*. Ini berarti kita membuat suatu entitas yang disebut "Buku." *Class* ini akan memiliki atribut dan metode tertentu yang mendefinisikan perilakunya.

b. Konstruktor `__init__`

Metode `__init__` adalah konstruktor *class*. Ketika membuat objek dari *class* ini, metode ini akan dijalankan. Tujuan dari konstruktor ini adalah untuk menginisialisasi atribut-atribut objek. Dalam hal ini, konstruktor menerima empat argumen: title (judul buku), writer (penulis buku), yearpublish (tahun penerbitan buku), dan

price (harga buku). Kemudian, atribut-atribut objek seperti *self.title*, *self.writer*, *self.yearpublish* dan *self.price* akan diinisialisasi dengan nilai yang sesuai.

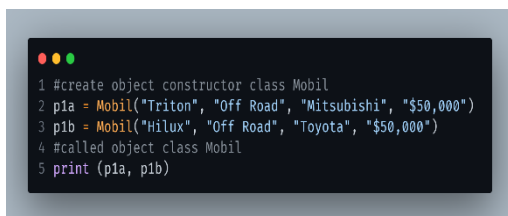
c. Metode `__str__`

Ini adalah metode bawaan yang digunakan untuk mengonversi objek menjadi string. Ketika mencetak objek Buku, metode `__str__` akan dipanggil. Dalam kode ini, metode ini mengembalikan string yang berisi nilai dari atribut-atribut objek seperti *title*, *writer*, *yearpublish* dan *price*.

B. Create And Call Object

Object dalam hal ini adalah untuk menampilkan dari semua property dan method dari property yang ada di *class*.

1. Create And Call Object Class Mobil



```
1 #create object constructor class Mobil
2 p1a = Mobil("Triton", "Off Road", "Mitsubishi", "$50,000")
3 p1b = Mobil("Hilux", "Off Road", "Toyota", "$50,000")
4 #called object class Mobil
5 print (p1a, p1b)
```

Gambar 2.2.1 Object Class Mobil

a. Object Class

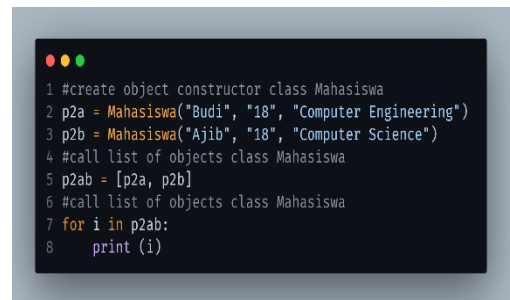
Dalam masing-masing pemanggilan konstruktor, Memberikan argumen-argumen yang sesuai dengan parameter konstruktor *class* "Mobil." Misalnya, p1a adalah objek yang memiliki atribut merk = "Triton", model = "Off Road", production = "Mitsubishi", dan price = "\$50,000". Demikian pula, p1b memiliki atribut-atribut yang sesuai.

b. Call Code Class Mobil

Dalam pernyataan ini, mencetak objek-objek tersebut. Ketika mencetak objek dalam Python, Python akan mencoba menggunakan metode `__str__` dari objek tersebut untuk mengonversinya menjadi string. Karena *class* "Mobil" memiliki metode `__str__` yang

mengembalikan string yang berisi informasi tentang mobil.

1. Create And Call Object Class Mahasiswa



```
1 #create object constructor class Mahasiswa
2 p2a = Mahasiswa("Budi", "18", "Computer Engineering")
3 p2b = Mahasiswa("Ajib", "18", "Computer Science")
4 #call list of objects class Mahasiswa
5 p2ab = [p2a, p2b]
6 #call list of objects class Mahasiswa
7 for i in p2ab:
8     print (i)
```

Gambar 2.2.2 Object Class Mahasiswa

a. Object Class

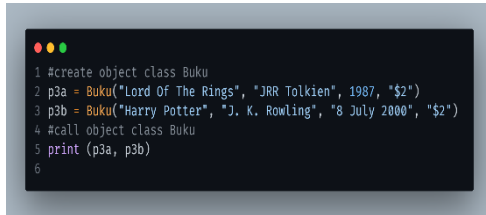
Dalam masing-masing pemanggilan konstruktor, Memberikan argumen-argumen yang sesuai dengan parameter konstruktor *class* "Mobil." Misalnya, p2a adalah objek yang memiliki atribut name = "Budi", age = "18" dan departement = "Computer Engineering". Demikian pula, p2b memiliki atribut-atribut yang sesuai. Kemudian dilanjutkan dengan pembuatan object p2ab yang mana memiliki list object p2a dan p2b dalam bentuk [p2a, p2b].

b. Call Code Class Mobil

```
for i in p2ab:
    print(i)
```

Dalam pernyataan ini, Mencetak objek-objek pada object p2ab. Python akan mencoba menggunakan metode `__str__` pada setiap object p2ab untuk mengonversinya menjadi string. Karena *class* "Mahasiswa" memiliki metode `__str__` yang mengembalikan string yang berisi informasi tentang mahasiswa.

2. Create And Call Object Class Buku



```
1 #create object class Buku
2 p3a = Buku("Lord Of The Rings", "JRR Tolkien", 1987, "$2")
3 p3b = Buku("Harry Potter", "J. K. Rowling", "8 July 2000", "$2")
4 #call object class Buku
5 print (p3a, p3b)
6
```

Gambar 2.2.3 Object Class Buku

a. Object Class

p3a = Buku("Lord Of The Rings", "JRR Tolkien", 1987, "\$2") adalah objek pertama dengan judul "Lord Of The Rings," penulis "JRR Tolkien," tahun terbit 1987, dan harga \$2. Begitu juga dengan p3b sama halnya dengan p3a.

b. Call Code Class Mobil

Mencetak objek-objek tersebut dengan pernyataan print (p3a, p3b). Saat mencetak objek seperti ini, Python akan memanggil metode `__str__` yang telah definisikan pada *class* "Buku." Oleh karena itu, melihat keluaran yang berisi informasi tentang buku seperti judul, penulis, tahun terbit, dan harga.

III. KESIMPULAN

Object dibuat, metode khusus `__init__` (constructor) digunakan untuk menginisialisasi atribut-atribut objek. Constructor ini akan dipanggil secara otomatis saat objek dibuat. Array adalah tipe data yang digunakan untuk menyimpan banyak nilai dalam satu variabel. Dalam Python, kita menggunakan list untuk membuat array. *List* dapat berisi objek dengan berbagai tipe data. *Class* digunakan untuk membuat struktur data yang lebih kompleks dan terorganisir. Ini membantu dalam pemeliharaan dan manajemen kode yang lebih baik. *Object* adalah wujud konkret dari *class* yang digunakan untuk mengakses atribut dan metode *class*. *Object* memungkinkan kita untuk menciptakan dan memanipulasi banyak entitas yang serupa dengan cara yang terstruktur.

DAFTAR PUSTAKA

- [1] T. Penulis *et al.*, *ALGORITMA DAN STRUKTUR DATA*. 2022. [Online]. Available: www.penerbitwidina.com
- [2] P. Wentworth, J. Elkner, A. B. Downey, and C. Meyers, "How to Think Like a Computer Scientist: Learning with Python 3 Documentation Release 3rd Edition," 2020.