

LAPORAN PRAKTIKUM ALGORITMA PEMROGRAMAN KE 11

CLASS IN PYTHON



DISUSUN OLEH :

Oktario Mufti Yudha

2320506044

JURUSAN TEKNOLOGI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS TIDAR

2023

LAPORAN

ALGORITMA PEMROGRAMAN DAN STRUKTUR DATA



Diisi Mahasiswa Praktikan									
Nama Praktikan	Oktario Mufti Yudha								
NPM	2320506044								
Rombel	4								
Judul Praktikum	Class in Python								
Tanggal Praktikum	22 November 2023								
Diisi Asisten Praktikum									
Tanggal Pengumpulan	<table border="1"><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>								
Catatan									

PENGESAHAN		NILAI
Diperiksa oleh :	Disahkan oleh :	
Asisten Praktikum	Dosen Pengampu	
(Kurnadi)	(Imam Adi Nata M.kom)	

BAB I

TUJUAN

1. Mahasiswa mampu memahami dan mengetahui apa itu class, method, dan atribut pada python
2. Mahasiswa dapat menerapkan class sesuai dengan kebutuhan
3. Mahasiswa dapat menggunakan sumber daya yang telah tersedia untuk memperluas kemampuan Python secara signifikan

BAB II

DASAR TEORI

Class atau kelas-kelas adalah prototipe untuk wadah menghimpun data dan kebergunaan yang kemudian menghasilkan objek. Setiap class baru akan menghasilkan objek baru yang kemudian bisa dibuat *instance* dengan memiliki atribut yang ada.

Misal mobil adalah template sedangkan Toyota Rush, Mitsubishi Xpander, Datsun GO adalah **objek/instance**. Setiap instance dapat memiliki semua atribut mobil. Atau class manusia, Anda dan saya adalah instance, antara saya dan Anda bisa berinteraksi dan memiliki atribut yang ada.

Beberapa istilah yang sering digunakan dalam OOP

1. **Class**: protoripe untuk mendefinisikan seperangkat atribut.
2. **Class Variable**: Variabel yang berada diluar method. Anda bisa membaca kembali halaman variable pada python
3. **Data Member**: Variable Class yang menyimpan data contoh
4. **Function overloading**: FUnksi yang bisa melakukan beberapa hal berbeda dan memiliki nama yang sama dalam class.
5. Instance Variable: variabel yang didefinisikan dalam methon dan hanya dimiliki oleh instance class.
6. **Inheritance**: Pewarisan karakteristik dari kelas tertentu dan menjadi turunannya.
7. **Instance**: Objek individu dan juga merupakan sebutan lain dari objek yang masuk dalam lingkaran kelas.
8. **Instantiation**: Pembuatan instansi suatu kelas.
9. **Method**: Fungsi yang didefinisikan didalam kelas.
10. **Object**: Wujud dari class yang merupakan prototipe, dan object adalah barang jadi.

11. **Operator overloading:** penugasan suatu fungsi operator tertentu lebih dari satu fungsi.

BAB III

ALAT DAN BAHAN

1. Laptop
2. Visual Studio Code
3. Extention Python
4. Extention Jupyter

BAB IV

LANGKAH KERJA

1. Siapkan alat dan bahan yang akan digunakan ketika praktikum.
2. Melakukan percobaan mengenai try except seperti berikut:



```
class Kucing:
    def __init__(self, nama, umur, warna):
        self.nama = nama
        self.umur = umur
        self.warna = warna

    def bersuara(self):
        print("Meow")

kucing1 = Kucing("Tom", 3, "Hitam")

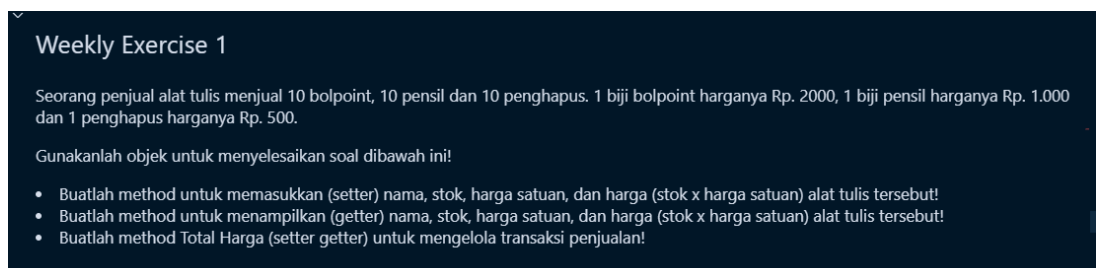
print(kucing1.nama)
```

✓ 0.0s

Tom

Gambar 4.1 Percobaan

3. Mulai mengerjakan weekly exercise 1 dengan soal sebagai berikut:



Weekly Exercise 1

Seorang penjual alat tulis menjual 10 bolpoint, 10 pensil dan 10 penghapus. 1 biji bolpoint harganya Rp. 2000, 1 biji pensil harganya Rp. 1.000 dan 1 penghapus harganya Rp. 500.

Gunakanlah objek untuk menyelesaikan soal dibawah ini!

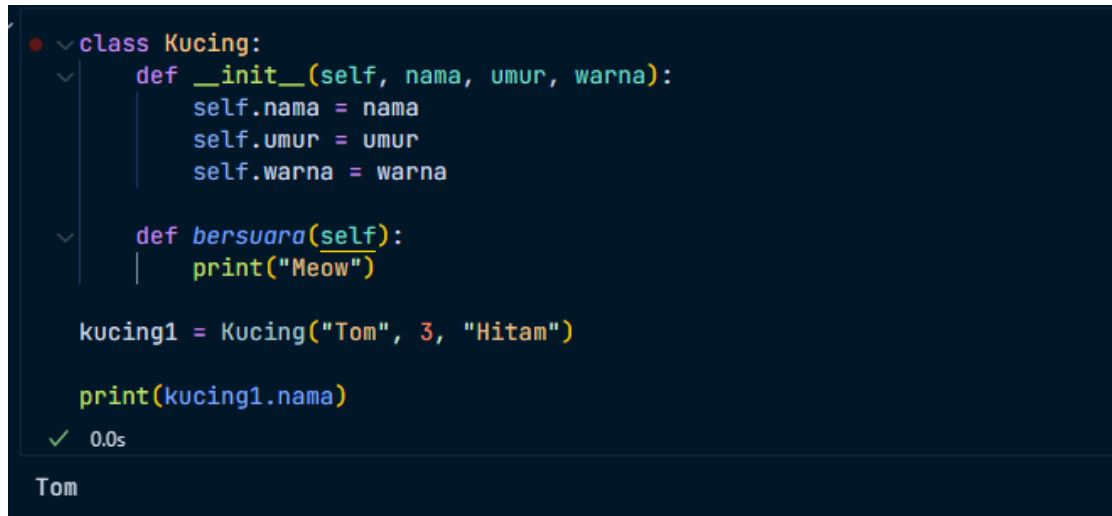
- Buatlah method untuk memasukkan (setter) nama, stok, harga satuan, dan harga (stok x harga satuan) alat tulis tersebut!
- Buatlah method untuk menampilkan (getter) nama, stok, harga satuan, dan harga (stok x harga satuan) alat tulis tersebut!
- Buatlah method Total Harga (setter getter) untuk mengelola transaksi penjualan!

Gambar 4.2 Weekly Exercise 1

BAB V

HASIL DAN ANALISIS

A. Percobaan



```
class Kucing:
    def __init__(self, nama, umur, warna):
        self.nama = nama
        self.umur = umur
        self.warna = warna

    def bersuara(self):
        print("Meow")

kucing1 = Kucing("Tom", 3, "Hitam")

print(kucing1.nama)
```

✓ 0.0s

Tom

Gambar 5.1 Code Percobaan

4. **class Kucing:** Ini adalah awal dari pembuatan sebuah kelas dalam Python yang disebut Kucing. Ini adalah sebuah template untuk membuat objek kucing.
5. **def __init__(self, nama, umur, warna):** **__init__** adalah metode khusus yang dipanggil saat objek kucing dibuat. Saat objek kucing dibuat, metode **__init__** ini akan dijalankan untuk menginisialisasi atribut-atribut dari objek kucing dengan nilai yang diberikan.
6. **self.nama = nama, self.umur = umur, self.warna = warna:** Ini menyimpan nilai-nilai ini ke dalam objek kucing yang dibuat.
7. **def bersuara(self):** Ini adalah sebuah metode di dalam kelas Kucing yang bernama bersuara. Metode ini akan mencetak suara kucing, dalam hal ini "Meow".
8. **kucing1 = Kucing("Tom", 3, "Hitam"):** Membuat objek kucing pertama dari kelas Kucing. Objek ini disimpan dalam variabel kucing1. Saat objek ini dibuat, **__init__** dipanggil dengan nilai "Tom" untuk nama, 3 untuk umur, dan "Hitam" untuk warna.
9. **print(kucing1.nama):** Baris ini mencetak nilai atribut nama dari objek kucing1, yang dalam hal ini akan mencetak "Tom" karena nilai nama objek kucing1 adalah "Tom".

B. Weekly Exercise 1

```
class AlatTulis():
    def __init__(self, nama, stok, hargaSatuan):
        self.nama = nama
        self.stok = stok
        self.hargaSatuan = hargaSatuan
        self.harga = stok * hargaSatuan

    def setNama(self, nama):
        self.nama = nama

    def setStok(self, stok):
        self.stok = stok
        self.harga = self.hargaSatuan * self.stok

    def setHargaSatuan(self, hargaSatuan):
        self.hargaSatuan = hargaSatuan
        self.harga = self.hargaSatuan * self.stok

    def getNama(self):
        return self.nama

    def getStok(self):
        return self.stok

    def getHargaSatuan(self):
        return self.hargaSatuan

    def getHarga(self):
        return self.harga

ballpoint = AlatTulis("Ballpoint", 10, 2000)
pensil = AlatTulis("Pensil", 10, 1000)
penghapus = AlatTulis("Penghapus", 10, 500)
```

Gambar 5.2 Weekly Exercise 1

1. **class AlatTulis():** Ini adalah pembuatan sebuah kelas dalam Python yang disebut AlatTulis. Kelas ini berfungsi sebagai blueprint atau template untuk menciptakan objek yang mewakili alat tulis.
2. **def __init__(self, nama, stok, hargaSatuan):** `__init__` adalah metode khusus yang dipanggil saat objek AlatTulis dibuat. Metode ini memiliki parameter `self` yang merujuk pada objek itu sendiri, serta parameter lainnya seperti `nama`, `stok`, dan `hargaSatuan`. Saat objek AlatTulis dibuat, metode `__init__` akan dijalankan untuk menginisialisasi atribut-atribut dari objek AlatTulis dengan nilai-nilai yang diberikan.
3. **self.nama = nama, self.stok = stok, self.hargaSatuan = hargaSatuan:** Baris-baris ini mengatur atribut-atribut dari objek AlatTulis (`nama`, `stok`, `hargaSatuan`) dengan nilai yang diterima saat pembuatan objek AlatTulis.
4. **self.harga = stok * hargaSatuan:** Inisialisasi atribut `harga` berdasarkan hasil perkalian `stok` dengan `hargaSatuan`. Ini menghitung harga total berdasarkan `stok` dan `harga` satuan saat objek AlatTulis dibuat.
5. **def setNama(self, nama):, def setStok(self, stok):, def setHargaSatuan(self, hargaSatuan):** Ini adalah metode-metode untuk mengubah nilai dari atribut-atribut objek AlatTulis (`nama`, `stok`, `hargaSatuan`). Metode-metode ini memungkinkan penggunaan untuk mengubah nilai-nilai ini setelah objek dibuat.

6. **def getName(self):, def getStok(self):, def getHargaSatuan(self):, def getHarga(self)::** Ini adalah metode-metode untuk mengambil nilai-nilai atribut dari objek AlatTulis (nama, stok, hargaSatuan, harga). Metode-metode ini memungkinkan penggunaan untuk mendapatkan nilai-nilai ini dari objek.
7. **ballpoint = AlatTulis("Ballpoint", 10, 2000), pensil = AlatTulis("Pensil", 10, 1000), penghapus = AlatTulis("Penghapus", 10, 500):** Baris-baris ini membuat tiga objek dari kelas AlatTulis yaitu ballpoint, pensil, dan penghapus. Masing-masing objek ini diinisialisasi dengan nama, stok, dan harga satuan yang berbeda.

```
print("===== BALLPOINT =====")

print("Nama\t\t\t\t\t: ", ballpoint.getName())
print("Stok\t\t\t\t\t: ", ballpoint.getStok())
print("Harga Satuan\t\t\t\t\t: ", ballpoint.getHargaSatuan())
print("Total harga\t\t\t\t\t: ", ballpoint.getHarga())
print("===== RESTOK BALLPOINT COY =====")

ballpoint.setStok(20)
print("Stok setelah restok\t\t\t\t\t: ", ballpoint.getStok())
print("Total harga setelah restok\t\t\t\t\t: ", ballpoint.getHarga())

print()
print("===== PENSIL =====")

print("Nama\t\t\t\t\t: ", pensil.getName())
print("Stok\t\t\t\t\t: ", pensil.getStok())
print("Harga Satuan\t\t\t\t\t: ", pensil.getHargaSatuan())
print("Total harga\t\t\t\t\t: ", pensil.getHarga())
print("===== RESTOK PENSIL COY =====")

pensil.setStok(20)
print("Stok setelah restok\t\t\t\t\t: ", pensil.getStok())
print("Total harga setelah restok\t\t\t\t\t: ", pensil.getHarga())

print()
print("===== PENGHAPUS =====")

print("Nama\t\t\t\t\t: ", penghapus.getName())
print("Stok\t\t\t\t\t: ", penghapus.getStok())
print("Harga Satuan\t\t\t\t\t: ", penghapus.getHargaSatuan())
print("Total harga\t\t\t\t\t: ", penghapus.getHarga())
print("===== RESTOK PENGHAPUS COY =====")

penghapus.setStok(20)
print("Stok setelah restok\t\t\t\t\t: ", penghapus.getStok())
print("Total harga setelah restok\t\t\t\t\t: ", penghapus.getHarga())
```

8. Menampilkan informasi awal tentang beberapa objek yang sudah di buat seperti ballpoint, penghapus, dan pensil.
9. Melakukan restok kepada penghapus, ballpoint, dan pensil dengan menaikkan stok dan menampilkan informasi baru tentang stok dan total harga setelah restok.

BAB VI

KESIMPULAN

Dalam praktikum dengan materi class pada Python, konsep kelas dan objek digunakan untuk mengorganisir data dan perilaku terkait dalam sebuah struktur yang terstruktur. Penggunaan metode dan atribut dalam kelas memungkinkan pengelompokan fungsi dan data terkait bersama, sementara inisialisasi objek dan konsep encapsulation membantu dalam mengatur akses terhadap data. Selain itu, konsep OOP memungkinkan untuk merepresentasikan objek dunia nyata secara lebih terstruktur, sambil menerapkan pewarisan dan polimorfisme untuk mengembangkan hierarki kelas dan mendukung fleksibilitas dalam kode.

BAB VII

DAFTAR PUSTAKA

1. *Alfian Ma'arif(2020).Buku Ajar Pemrograman Lanjut Bahasa Pemrograman Python: Universitas Ahmad Dahlan*
2. *belajarpython.com, “Object & Class Python”, 18 November 2018, < <https://belajarpython.com/tutorial/object-class-python/>> [28 November 2023].*