TUTORIAL LAB 7 DDP-1 Kelas B dan D

Jumat, 10 November 2017

Anda diperbolehkan untuk membuka slide materi yang sudah diberikan. Pahami terlebih dahulu maksud dari soal yang diberikan. Dalam pengerjaan soal tutorial ini, Anda diperbolehkan untuk bertanya kepada dosen, asdos, ataupun teman Anda. Tetapi, Anda DILARANG untuk melakukan tindakan plagiarisme! Bagi mahasiswa yang diketahui melakukan tindakan plagiarisme, dalam bentuk apapun (memfoto code teman, copy-paste code teman, dll), akan dikenai sanksi pemberian nilai 0.

Class adalah sebuah prototipe yang merupakan dasar pembuatan objek. Prototipe ini mendefinisikan ciri-ciri objek, yang lalu dapat digunakan ke depannya.

1. Mendefinisikan Suatu Class

Kita bisa mendefinisikan sebuah *class* baru dengan menulis seperti berikut:

```
class ClassName:
    // isi dari class
```

Dalam Python, perlu dicatat bahwa nama *class* sebaiknya dalam bentuk **CamelCase**, atau menulis huruf besar untuk setiap awal kata.

2. `__init__()` function

Kita dapat mendefinisikan fungsi __init__() dalam *class*, yang merupakan fungsi paling pertama yang dipanggil ketika membuat suatu objek dari *class* tersebut. Jika tidak didefinisikan, maka fungsi __init__() secara default tidak akan melakukan apapun.

Kita bisa mendefinisikan fungsi __init__() dengan menulis seperti berikut:

```
class ClassName:     def __init__(self):
        // hal-hal yang ingin dilakukan ketika
objek dibuat
```

Kita akan membahas kegunaan dari fungsi __init__() pada bagian **<3. Class Attributes>**.

3. Attribute

Attribute dari suatu class terdiri dari data-data (instance variables) dan methods.

3.1 Data Attributes

Data attribute adalah *attribute* berbentuk variabel yang khusus terhadap objek tersebut.

Perhatikan contoh pembuatan *data attribute* pada *class* Mobil berikut:

```
class Mobil:
    def __init__(self):
        self.merek     =  'BMW'
        self.pemilik  =  'Paman'
        self.menyala = False
```

Pada contoh di atas, kita membuat 3 variabel yang bernama `merek`, `pemilik` dan `menyala`. Karena kita ingin supaya Mobil tersebut langsung memiliki ketiga ciri-ciri di atas, kita masukkan ke dalam fungsi `__init__()`.

Perhatikan bahwa untuk setiap penulisan variabel kita menulis `self.` terlebih dahulu. Ini menandakan bahwa variabel tersebut merupakan bagian dari objek Mobil, bukan variabel dalam fungsi `__init__()`.

Tentunya kita juga dapat membuat *class* Mobil sedemikian sehingga beberapa variabel ditentukan ketika membuat objek. Perhatikan modifikasi contoh pada *class* Mobil berikut:

```
class Mobil:
    def __init__(self, merek, pemilik):
        self.merek     =  merek
        self.pemilik     =  pemilik
        self.menyala = False
```

Pada modifikasi di atas, kita membuat *class* Mobil sedemikian sehingga variabel `merek` dan `pemilik` bebas ditentukan ketika objek dibuat, dan objek Mobil tersebut tidak menyala pada awalnya. Kita akan membahas cara membuat objek dan memanggil *data attribute* pada bagian **<4. Instance Object>**.

3.2 Methods

Method adalah *attribute* yang berbentuk seperti fungsi.

Perhatikan contoh method `klakson()` pada *class* Mobil berikut:

```
class Mobil:
    def klakson(self):
        print('TIN TIN')
```

Keluaran:

```
TIN TIN
```

Pada contoh di atas, kita membuat sebuah fungsi klakson() yang hanya akan *print* suatu tulisan. Perhatikan bahwa setiap *method* harus memiliki suatu parameter `self` sebagai parameter pertama. Parameter ini merupakan *reference* ke objek yang memanggil *method* tersebut.

Perhatikan contoh penggunaan self pada *class* Mobil berikut:

```
class Mobil:
    def __init__(self, merek, pemilik):
        self.merek = merek
        self.pemilik = pemilik
        self.menyala = False

    def ganti_pemilik(self, pemilik_baru):
        self.pemilik = pemilik_baru
```

Pada contoh di atas, kita membuat sebuah fungsi ganti_pemilik() yang akan mengubah *data attribute* pemilik dari objek sekarang. Kita akan membahas cara memanggil *method* pada bagian **<4. Instance Object>**.

4. Instance Object

5. Contoh *class* Mobil

```
class Mobil:
    def __init__(self, merek, pemilik):
        self.merek = merek
        self.pemilik = pemilik
        self.menyala = False

    def klakson(self):
        print('TIN TIN')

    def ganti_pemilik(self, pemilik_baru):
        self.pemilik = pemilik_baru

    def nyalakan(self):
        self.menyala = True
        print('Mobil menyala, BRUM BRUM')

    def matikan(self):
        self.menyala = False
```

```
        print('Mobil mati X(')

    def  is_mahal(self):
        if (self.merek ==
'BMW'):             return
True            else:
            return False

mobil = Mobil('BMW', 'Paman')
mobil.ganti_pemilik('Saya
sendiri') mobil.nyalakan()
mobil.klakson() mobil.matikan()
 if
(mobil.is_mahal()):
    print('WoOoW, mobil ini mahal!') else:
    print('Hah, mobil pecundang.')
```

Untuk dokumentasi lebih lengkap dapat dicek di:

- https://www.tutorialspoint.com/python/python_classes_objects.html

https://docs.python.org/3/tutorial/classes.html

BenJek™ (Benny Ojek)

"Dari tadi belom nyampe Gerbatama juga? BenJek-in aja!"

Peminat transportasi online sekarang ini semakin bertambah. Hampir setiap hari, jalanan di ibukota selalu dipenuhi oleh transportasi *online*. Melihat peluang yang sangat besar itu, Benny pun berniat untuk membuat sebuah transportasi *online* terbaru, yaitu BenJek, tentunya dengan dukungan dari pendahulunya yang sudah terjun di dunia transportasi *online*, yaitu Nadiem Makara, pendiri Ow-Jek.

Pada tahap pertama peluncuran BenJek, Benny merekrut sejumlah *driver*, dengan ketentuan pendaftaran sebagai berikut:

- **BenJek membuka 2 tipe layanan, yaitu "Normal Ojek" dan "Sport Ojek".**
- **Setiap calon pendaftar diminta untuk melampirkan nama dan layanan BenJek yang ingin diikuti.**

Jika calon pendaftar sudah diterima sebagai *driver* BenJek, mereka sudah bisa mulai untuk melayani penumpang, dengan ketentuan-ketentuan sebagai berikut (disesuaikan dengan tipe layanan yang dimiliki oleh *driver*):

No	Layanan Ojek	Ketentuan
1.	Normal Ojek	- Tidak ada batas minimum perjalanan - Tarif per KM sebesar Rp. 1.000
2.	Sport Ojek	- Minimal perjalanan sejauh 10 KM - Tarif per KM sebesar Rp. 2.500

Setiap akhir bulan, bagian keuangan BenJek memulai perhitungan untuk membagi pendapatan yang diperoleh *driver* BenJek. Adapun ketentuan pembagian pendapatan adalah sebagai berikut:
Driver dari layanan manapun akan mendapatkan 80% dari total pendapatan mereka, BenJek akan mendapatkan 20% dari total pendapatan mereka.

Untuk mempermudah kinerja dari BenJek, Benny memintamu untuk membuat sebuah program yang bekerja sesuai dengan ketentuan-ketentuan yang sudah dijabarkan di atas.

ASUMSI

- **Nama *driver* yang akan mendaftar dijamin unik, sehingga calon *driver* yang memiliki nama yang sama dengan salah satu *driver* yang sudah mendaftar, maka calon *driver* tersebut tidak dapat mendaftar**
- **Satu *driver* hanya bisa punya satu jenis layanan ojek (salah satu dari NORMAL atau SPORT).**

FORMAT INPUT

- **DAFTAR <nama_driver> <NORMAL/SPORT>**
  Mendaftarkan <nama_driver> ke dalam sistem BenJek dan akan melayani NORMAL/SPORT Ojek.

- **MULAI PERJALANAN <nama_driver> <jarak_ditempuh_km>**
  <nama_driver> melakukan perjalanan sejauh <jarak_ditempuh_km>. Hasil yang diperoleh akan dimasukkan ke dalam pendapatan driver.

- **CEK PENDAPATAN <nama_driver>**
  Command ini mengecek jumlah pendapatan sementara yang diperoleh oleh <nama_driver>.

- **AKHIR BULAN**
  Command ini akan mengakhiri program (exit program). Membagi pendapatan antara driver dengan BenJek.

FORMAT OUTPUT

- **DAFTAR <nama_driver> <NORMAL/SPORT>**
  - **Jika driver belum ada di dalam sistem**
    <nama_driver> berhasil mendaftar sebagai driver BenJek layanan <NORMAL/SPORT>
  - **Jika driver sudah ada di dalam sistem**
    <nama_driver> gagal mendaftar sebagai driver BenJek

- **MULAI PERJALANAN <nama_driver> <jarak_ditempuh_km>**
  - **Jika driver sudah ada di dalam sistem dan bisa melakukan perjalanan:**
    <nama_driver> melakukan perjalanan sejauh <jarak_ditempuh_km> dan mendapatkan pendapatan sebesar <pendapatan_diperoleh>
  - **Jika driver sudah ada di dalam sistem dan tidak bisa melakukan perjalanan:**
    <nama_driver> tidak bisa melakukan perjalanan

- **Jika driver tidak ada di dalam sistem:**
  <nama_driver> tidak ada di dalam sistem

- **CEK PENDAPATAN <nama_driver>**
  - **Jika driver sudah ada di dalam sistem**
    <nama_driver> memiliki pendapatan sebesar Rp.<pendapatan_diperoleh>
  - **Jika driver tidak ada di dalam sistem**
    <nama_driver> tidak ada di dalam sistem

- **AKHIR BULAN**
  Sudah akhir bulan! Pendapatan BenJek bulan ini adalah Rp.<pendapatan_BenJek>
  Daftar pendapatan pengemudi:
  <nama_driver_1>: Rp.<pendapatan_diperoleh_1>
  <nama_driver_2>: Rp.<pendapatan_diperoleh_2>
  ...
  <nama_driver_n>: Rp.<pendapatan_diperoleh_n>

BONUS(?)

Setelah Benny sukses meluncurkan layanan BenJek dengan Normal Ojek dan Sport Ojek, Benny berencana untuk membuka layanan baru dengan nama Cruiser Ojek. Adapun ketentuan untuk Cruiser Ojek adalah sebagai berikut:

- **Minimal perjalanan sejauh 25 KM**
- **Tarif per KM sebesar Rp. 7.500**

Silakan implementasikan layanan Cruiser Ojek di dalam kalian untuk menyelesaikan soal bonus!