
PEMROGRAMAN JARINGAN



Oleh:

NAMA : RAGIL AFDA TRIPRADANA

NPM : 24782091

PRODI : S1 Teknologi Rekayasa Internet

KELAS : 2C

Dosen:

Ir. Nurul Qomariyah, M.Kom.

TEKNOLOGI REKAYASA INTERNET

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI LAMPUNG

2025

1. Deskripsi tugas

Pada tugas saat ini kita akan melanjutkan dari Lab Session. Dimana kita akan menerapkan OOP (*Object Oriented Programming*) di kode yang sudah kita buat dan dengan kasus yang sama. Ketentuan tugas :

- Menambahkan class Sensor dengan atribut nama_sensor, jenis_sensor, dan data_sensor. Lalu , menambahkan method untuk mendapatkan informasi dari kualitas sensor dalam bentuk format string.
- Menambahkan class Location pada modul sensor yang memiliki atribut kelurahan, kecamatan, object sensor PM25, object sensor CO, dan object sensor kebisingan. Lalu , menambahkan method untuk mendapatkan informasi mengenai data kualitas udara pada lokasi yang dinyatakan dalam object.
- Menginstansiasi beberapa objek Location yang kemudian disimpan dalam sebuah list.
- Memodifikasi modul main, agar program berjalan sesuai dengan implementasi OOP.

2. Implementasi di modul Sensor_Simulator

```
6
7 import random
8
9 class Sensor:
10     """
11     Class untuk merepresentasikan sebuah sensor kualitas udara.
12
13     Atribut:
14     nama_sensor (str): Nama sensor.
15     jenis_sensor (str): Jenis sensor (PM25, CO, Kebisingan).
16     data_sensor (float): Data hasil pengukuran sensor.
17     """
18
19     def __init__(self, nama_sensor, jenis_sensor):
20         """Inisialisasi atribut dan generate data sensor."""
21         self.nama_sensor = nama_sensor
22         self.jenis_sensor = jenis_sensor
23         self.data_sensor = self.generate_data()
24
25     def generate_data(self):
26         """Menghasilkan data acak berdasarkan jenis sensor."""
27         if self.jenis_sensor == "PM25":
28             return round(random.uniform(0, 150), 2)
29         elif self.jenis_sensor == "CO":
30             return round(random.uniform(0, 10), 2)
31         elif self.jenis_sensor == "Kebisingan":
32             return round(random.uniform(30, 100), 2)
33         else:
34             return 0
35
36     def get_info(self):
37         """Mengembalikan informasi sensor dalam format string."""
38         return f"{self.nama_sensor} ({self.jenis_sensor}): {self.data_sensor}"
39
```

sensor dibuat sebagai objek yang memiliki atribut nama, jenis, dan data pengukuran.

Terdapat tiga komponen utama:

- (1) Atribut class yang terdiri dari nama_sensor untuk identifikasi, jenis_sensor untuk menentukan tipe pengukuran ("PM25", "CO", atau "Kebisingan"), dan data_sensor untuk menyimpan nilai hasil pengukuran;
- (2) Method `__init__` yang berfungsi sebagai constructor untuk inisialisasi objek sekaligus memanggil method `generate_data()`;

Method `generate_data()` ini berfungsi untuk menghasilkan data atau nilai acak berdasarkan jenis sensor tertentu. Method ini menggunakan kondisi untuk menentukan rentang nilai yang sesuai dengan jenis sensor. Jika jenis sensor tidak dikenali, method akan mengembalikan nilai default 0.

Method `get_info()` berfungsi untuk menampilkan informasi sensor dalam bentuk string yang terformat dengan rapi.

```
class Location:
    """
    Class untuk merepresentasikan lokasi pemantauan kualitas udara.

    Atribut:
        kelurahan (str): Nama kelurahan.
        kecamatan (str): Nama kecamatan.
        sensor_pm25 (Sensor): Objek sensor PM25.
        sensor_co (Sensor): Objek sensor CO.
        sensor_kebisingan (Sensor): Objek sensor Kebisingan.
    """

    def __init__(self, kelurahan, kecamatan, sensor_pm25, sensor_co, sensor_kebisingan):
        """Inisialisasi data lokasi dan sensor-sensornya."""
        self.kelurahan = kelurahan
        self.kecamatan = kecamatan
        self.sensor_pm25 = sensor_pm25
        self.sensor_co = sensor_co
        self.sensor_kebisingan = sensor_kebisingan

    def get_quality_info(self):
        return (
            f"Lokasi: {self.kelurahan}, {self.kecamatan}\n"
            f"  - {self.sensor_pm25.get_info()} {chr(181)}g/m{chr(179)}\n"
            f"  - {self.sensor_co.get_info()}PPM\n"
            f"  - {self.sensor_kebisingan.get_info()}dB\n"
        )
```

Class Location memiliki atribut yang terdiri dari kelurahan, kecamatan, serta tiga objek sensor (`sensor_`, `sensor_co`, dan `sensor_kebisingan`).

Method `get_quality_info()` berfungsi untuk menampilkan informasi kualitas udara di suatu lokasi dalam format string. Method ini mengembalikan string yang terdiri dari:

- (1) Nama lokasi (kelurahan dan kecamatan),
- (2) Data sensor PM2.5 dengan satuan $\mu\text{g}/\text{m}^3$ (menggunakan karakter khusus Unicode melalui `chr(181)` untuk simbol mikro dan `chr(179)` untuk pangkat tiga),

- (3) Data sensor CO dengan satuan PPM, dan
- (4) Data sensor kebisingan dengan satuan dB.

3. Implementasi di modul main

```

1 from sys import path
2 path.append('..\projek1one\SmartCity')
3
4 from Sensor.Sensor_Simulator import Sensor, Location
5
6 def tampilkan_menu():
7     print("\n=== Sistem Pemantauan Kualitas Udara ===")
8     print("1. Tambah Data Lokasi")
9     print("2. Tampilkan Semua Lokasi")
10    print("3. Keluar")
11
12    lokasi_list = []
13
14    while True:
15        tampilkan_menu()
16        pilihan = input("Pilih menu (1/2/3): ")
17
18        if pilihan == "1":
19            kelurahan = input("Masukkan nama kelurahan: ")
20            kecamatan = input("Masukkan nama kecamatan: ")
21
22            sensor_pm25 = Sensor(f"Sensor PM25 -", "PM25")
23            sensor_co = Sensor(f"Sensor CO -", "CO")
24            sensor_kebisingan = Sensor(f"Sensor Kebisingan -", "Kebisingan")
25
26            lokasi_baru = Location(kelurahan, kecamatan, sensor_pm25, sensor_co, sensor_kebisingan)
27            lokasi_list.append(lokasi_baru)
28            print("\nData lokasi berhasil ditambahkan!")
29
30        elif pilihan == "2":
31            print("\n=== Data Kualitas Udara Semua Lokasi ===")
32            for lokasi in lokasi_list:
33                print(lokasi.get_quality_info())
34
35        elif pilihan == "3":
36            print("\nTerima kasih telah menggunakan sistem ini!")
37            break
38        else:
39            print("Pilihan tidak valid. Silakan pilih 1, 2, atau 3.")

```

Pertama, program mengimpor class Sensor dan Location dari modul Sensor_Simulator setelah menyesuaikan path sistem. Program akan menampilkan menu dengan tiga opsi:

```

=== Sistem Pemantauan Kualitas Udara ===
1. Tambah Data Lokasi
2. Tampilkan Semua Lokasi
3. Keluar
Pilih menu (1/2/3): █

```

- (1) Menambahkan data lokasi baru dengan menginput nama kelurahan dan kecamatan, lalu secara otomatis membuat tiga objek sensor (PM2.5, CO, dan Kebisingan) yang berasal dari class Sensor. Lalu hasil input user dan nilai yang sudah dibuat akan diolah berdasarkan class Location dan hasilnya akan disimpan di list kosong.

```

=== Sistem Pemantauan Kualitas Udara ===
1. Tambah Data Lokasi
2. Tampilkan Semua Lokasi
3. Keluar
Pilih menu (1/2/3): 1
Masukkan nama kelurahan: tanjung
Masukkan nama kecamatan: senang

Data lokasi berhasil ditambahkan!

```

- (2) Menampilkan semua data lokasi yang tersimpan beserta informasi kualitas udaranya dalam format terstruktur menggunakan method `get_quality_info()`;

```

Data lokasi berhasil ditambahkan!

=== Sistem Pemantauan Kualitas Udara ===
1. Tambah Data Lokasi
2. Tampilkan Semua Lokasi
3. Keluar
Pilih menu (1/2/3): 2

=== Data Kualitas Udara Semua Lokasi ===
Lokasi: tanjung, senang
  - Sensor PM25 - (PM25): 61.35 µg/m³
  - Sensor CO - (CO): 2.05PPM
  - Sensor Kebisingan - (Kebisingan): 50.17dB

Lokasi: balai, agung
  - Sensor PM25 - (PM25): 92.86 µg/m³
  - Sensor CO - (CO): 1.83PPM
  - Sensor Kebisingan - (Kebisingan): 67.24dB

```

- (3) Keluar dari program.

```

=== Sistem Pemantauan Kualitas Udara ===
1. Tambah Data Lokasi
2. Tampilkan Semua Lokasi
3. Keluar
Pilih menu (1/2/3): 3

Terima kasih telah menggunakan sistem ini!
PS D:\projekgit\projek3>

```