RESUME
CLASS ABSTRAK, INTERFACE, METACLASS
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

FRANSISKUS XAVERIUS GUNAWAN

121140010

RB

INSTITUT TEKNOLOGI SUMATERA

TEKNIK INFORMATIKA

2023

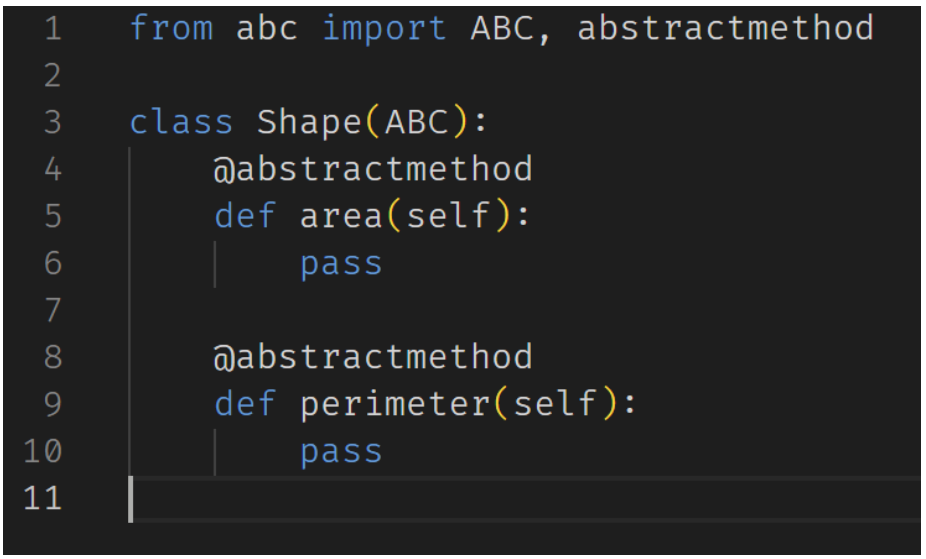
DAFTAR ISI

ABSTRAK CLASS

Kelas abstrak dalam Python adalah kelas yang tidak dapat dibuat instance-nya secara langsung, tetapi digunakan sebagai kerangka kerja untuk membuat kelas turunan yang mengimplementasikan metode abstrak yang didefinisikan di dalamnya. Kelas abstrak menyediakan kontrak atau cetak biru yang harus diikuti oleh kelas turunannya.

Dalam Python, kelas abstrak didefinisikan oleh modul abc (Kelas Basis Abstrak). Modul ini menyediakan dekorator @abstractmethod, yang digunakan untuk menandai metode yang harus diterapkan oleh kelas turunan.

Sebagai contoh, perhatikan kelas abstrak Shape, yang memiliki metode abstrak area() dan perimeter(). Kelas ini tidak dapat dibuat instance-nya, tetapi berfungsi sebagai basis untuk kelas turunan seperti Circle dan Rectangle, yang harus mengimplementasikan metode abstrak ini.



```python
from abc import ABC, abstractmethod

class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

    @abstractmethod
    def perimeter(self):
        pass

```

Abstrak

Kelas Shape di atas adalah kelas abstrak karena memiliki satu atau lebih metode abstrak yang dianotasi dengan dekorator @abstractmethod. Saat kelas turunan Shape dibuat, ia harus mengimplementasikan metode area() dan perimeter() dengan tepat.

Dengan menggunakan kelas abstrak, kita dapat memastikan bahwa kelas turunan mengikuti konvensi yang ditentukan oleh kelas abstrak. Ini membantu menciptakan struktur yang terorganisir dan konsisten dalam kode yang lebih besar dan membuatnya lebih mudah untuk mengembangkan sistem yang dapat diperluas.

INTERFACE

Dalam bahasa pemrograman Java, terdapat konsep Interface yang memungkinkan implementasi Multiple Inheritance. Java tidak mendukung Multiple Inheritance langsung, tetapi dengan menggunakan Interface, kita dapat menerapkan multiple inheritance secara tidak langsung. Interface dalam Java berfungsi sebagai kontrak yang mendefinisikan metode-metode yang harus diimplementasikan oleh kelas-kelas yang mengimplementasikan interface tersebut.

Namun, dalam bahasa pemrograman Python, konsep Interface tidak menjadi komponen tersendiri. Python mendukung Multiple Inheritance secara langsung tanpa memerlukan sintaks khusus seperti Interface. Dalam Python, kelas-kelas dapat mewarisi dari beberapa kelas secara langsung, yang memungkinkan penggunaan metode dan atribut dari semua kelas yang diwarisi.

Meskipun demikian, memahami konsep Interface tetap penting dalam desain program skala besar. Meskipun Python tidak memerlukan deklarasi formal untuk Interface, programmer dapat menggunakan pola desain atau konvensi tertentu untuk mencapai tujuan yang sama. Penggunaan dokumentasi yang jelas dan mengikuti kontrak yang telah ditetapkan dalam interface memudahkan pemeliharaan dan pengembangan kode dalam sistem yang kompleks.

```python
class PdfParser(InformalParserInterface):
    """Extract text from a PDF"""
    def load_data_source(self, path: str, file_name: str) → str:
        """Overrides InformalParserInterface.load_data_source()"""
        pass

    def extract_text(self, full_file_path: str) → dict:
        """Overrides InformalParserInterface.extract_text()"""
        pass
```

Implementasi dari Informal Parser Interface sekarang memungkinkan Anda untuk mengekstrak teks dari file PDF.

Interface digunakan untuk mendefinisikan kontrak yang menguraikan apa saja yang dapat dilakukan oleh sebuah kelas, tanpa perlu memperhatikan implementasi rinci dari setiap fungsi. Konsep interface mirip dengan konsep kelas dan fungsi abstrak, tetapi dengan perbedaan bahwa semua fungsi dalam interface didefinisikan sebagai abstrak.

Misalnya, fitur seperti GPS dan Bluetooth dapat diimplementasikan baik pada Smartphone maupun Smartwatch. Interface dapat digunakan untuk mendefinisikan kontrak yang menentukan bahwa kelas-kelas tersebut harus memiliki fungsi-fungsi terkait GPS dan Bluetooth, tetapi tidak memperhatikan bagaimana implementasinya.

Hal yang sama berlaku untuk fitur lain seperti dapat disewakan, yang dapat diimplementasikan pada objek seperti Mobil, Rumah, atau Buku. Selain itu, konsep interface juga dapat diterapkan pada fitur “Dapat Dimakan”, yang dapat diimplementasikan pada objek seperti Jamur atau Ayam.

META CLASS

Metaclass dalam Python adalah kelas yang digunakan untuk membuat kelas. Dalam istilah sederhana, metaclass adalah "kelas dari kelas". Ketika sebuah kelas dibuat, Python menggunakan metaclass untuk menghasilkan definisi kelas tersebut.

Metaclass memungkinkan Anda untuk mempengaruhi cara kelas-kelas didefinisikan dan perilaku mereka. Dengan menggunakan metaclass, Anda dapat memodifikasi atau memeriksa atribut dan metode yang didefinisikan dalam kelas, mengubah pewarisan, atau bahkan menambahkan perilaku baru ke kelas-kelas yang dibuat.

Untuk menggunakan metaclass, Anda perlu mendefinisikan kelas dengan metaclass tertentu dengan menggunakan sintaks khusus. Metaclass biasanya merupakan turunan dari kelas bawaan `type`. Dengan mendefinisikan metaclass, Anda dapat mengontrol bagaimana kelas tersebut akan didefinisikan dan berinteraksi dengan objek lain.

Namun, penggunaan metaclass tidak umum dalam pengembangan aplikasi Python yang umum. Biasanya, metaclass digunakan dalam skenario yang kompleks atau ketika Anda perlu melakukan pemrosesan kelas secara dinamis. Penggunaan yang bijaksana dan pemahaman yang mendalam tentang metaclass sangat penting untuk menghindari kompleksitas yang tidak perlu dalam kode.

```python
class MyMeta(type):
    def __new__(cls, name, bases, attrs):
        # Modifikasi atribut kelas sebelum dibuat
        attrs['extra_attr'] = 'Nilai tambahan dari metaclass'
        return super().__new__(cls, name, bases, attrs)

class MyClass(metaclass=MyMeta):
    def __init__(self):
        self.data = 'Data kelas'

obj = MyClass()
print(obj.data)  # Output: Data kelas
print(obj.extra_attr)  # Output: Nilai tambahan dari metaclass

```

MetaClass

Daftar Pustaka

https://www.python.org/dev/peps/pep-3119/

https://www.geeksforgeeks.org/abstract-classes-in-python/

https://drive.google.com/file/d/1Rz0uXfUc5CfguLP0AGpMJinjnR4NbuIu/view?usp=drive_web&authuser=0

https://realpython.com/python-metaclasses/