RESUME MODUL
LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

Disusun oleh:

Nama : Ahmad Rizki Maulana
NIM : 121140105
Kelas : PBO RB

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI SUMATERA

LAMPUNG SELATAN

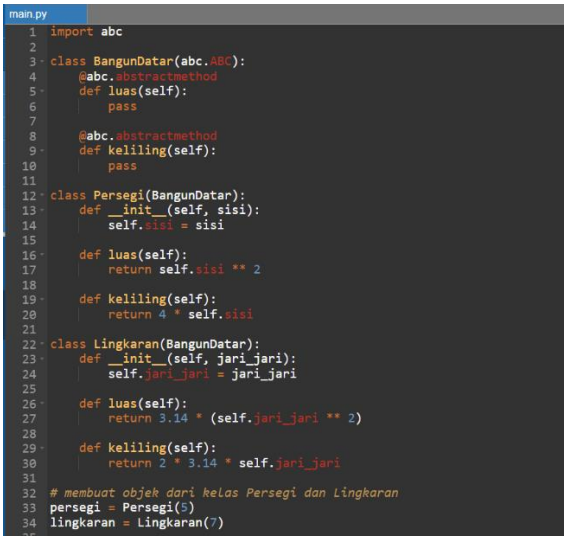
202

BAB I
RESUME

A. Kelas Abstrak

Kelas abstrak adalah kelas yang mempunyai setidaknya satu abstract method, kelas abstrak sendiri ditandai dengan mewarisi kelas ABC (Abstract Base Class). Method didalam kelas abstrak yang tidak mempunyai implementasi dinamakan method abstract. Abstract method adalah method yang tidak memiliki body (hanya deklarasi method), Contohnya : *public* *abstract* *void someMethod();*. Kelas abstrak tidak bisa dibuat objeknya, maka dari itu suatu kelas harus dapat diturunkan dimana pada subclass tersebut berisi implementasi dari abstract method yang ada di super class-nya.

Berikut adalah contoh sederhana pembuatan kelas abstraksi :



Dalam contoh di atas, terdapat kelas abstraksi BangunDatar yang memiliki dua metode abstrak yaitu luas() dan keliling(). Kelas Persegi dan Lingkaran merupakan turunan dari kelas BangunDatar, sehingga harus mengimplementasikan kedua metode abstrak tersebut.

B. Interface

Interface merupakan kumpulan deklarasi fungsi tanpa implementasi yang mendefinisikan aturan pemanggilan fungsi oleh sembarang class lain. Interface berisi method kosong dan konstanta, dan didalamnya tidak mempunyai statement sehingga deklarasi method dalam interface sama dengan deklarasi abstract method pada abstract class. Public dan abstract merupakan method yang dideklarasi oleh interface, dan variable yang digunakan adalah public, static, dan final. Berikut merupakan contoh pengimplementasian konsep interface :

main.py

```python
from abc import ABC, abstractmethod

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

    @abstractmethod
    def perimeter(self):
        pass

class Rectangle(Shape):
    def __init__(self, width, height):
        self.width = width
        self.height = height

    def area(self):
        return self.width * self.height

    def perimeter(self):
        return 2 * (self.width + self.height)

class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius ** 2

    def perimeter(self):
        return 2 * 3.14 * self.radius

```

```
...Program finished with exit code 0
Press ENTER to exit console.
```

Pada contoh di atas, kelas Shape merupakan sebuah kelas abstrak yang mewakili sebuah interface, yang memiliki dua metode abstrak, yaitu area dan perimeter. Kelas Rectangle dan Circle merupakan kelas turunan yang mengimplementasikan kelas abstrak Shape, sehingga wajib mengimplementasikan kedua metode abstrak tersebut. Kelas Rectangle dan Circle masing-masing mengimplementasikan metode area dan perimeter sesuai dengan kebutuhan objek yang mereka wakili, dan memastikan bahwa keduanya memenuhi kontrak yang telah didefinisikan oleh kelas Shape. Dalam contoh tersebut, penggunaan kelas abstrak Shape sebagai interface memastikan bahwa kelas-kelas turunan yang menerapkannya memiliki perilaku yang konsisten dan memenuhi kontrak yang telah ditentukan.

C. Metaclass

Metaclass adalah kelas yang mendefinisikan perilaku sebuah kelas, sedangkan kelas mendefinisikan perilaku sebuah objek. Metaclass biasanya digunakan sebagai pabrik kelas, sehingga memungkinkan untuk melakukan hal-hal ekstra saat membuat kelas seperti mendaftarkan kelas baru dengan beberapa registri atau mengganti kelas dengan sesuatu yang lain sama sekali. Dalam Python, metaclass dibuat dengan membuat subclass dari kelas type. Ketika pernyataan class dieksekusi, Python pertama mengeksekusi tubuh pernyataan class sebagai blok kode yang normal, kemudian metaclass dipanggil dengan nama, pangkalan, dan atribut dari kelas untuk melakukan instansiasi. Metaclass juga memungkinkan untuk mendefinisikan metode normal dan metode 'sihir' normal pada metaclass itu sendiri.