UNIVERSITAS SAM RATULANGI MANADO
FAKULTAS TEKNIK, JURUSAN TEKNIK ELEKTRO
Program Studi S-1 Teknik Informatika

Kelas dan Objek (Part 2)

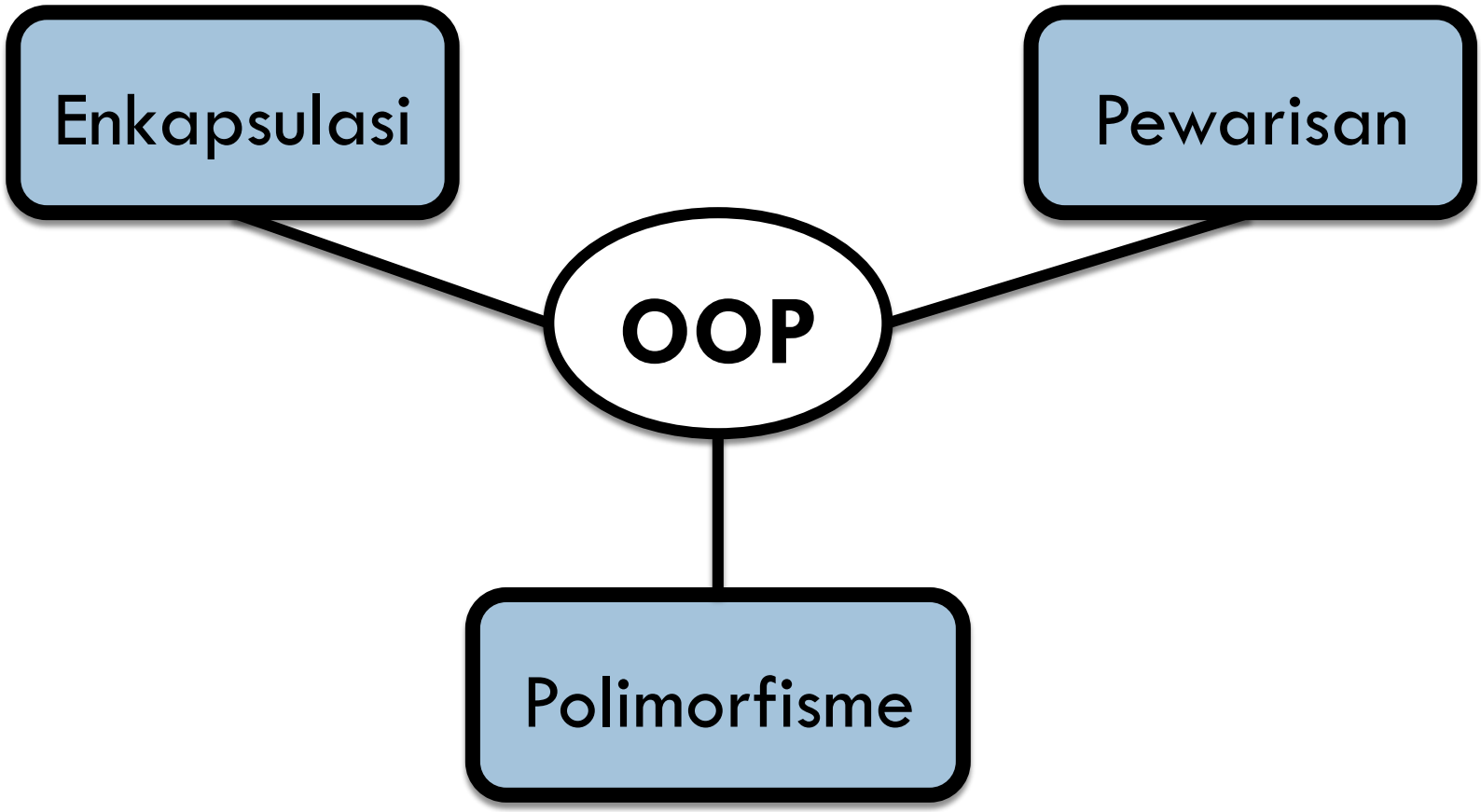
Mata Kuliah: Algoritma & Logika Informatika (IFC3504)

Alwin M. Sambul, S.T., M.Eng., Ph.D.

0 Review

Demo: Kelas & Objek

Tiga Fitur Dasar OOP



1 Encapsulation (Enkapsulasi)

Definisi Kelas

kalkulator4.py - /Users/asambul/Programming/python/kuliah12/kalkulator4.py (3.4.3)

```
class Kalkulator:
    def __init__(self,x,y):
        self.A = x
        self.B = y
        print ("A="+str(x)+",B="+str(y))
    def tambah(self):
        self.hasil = self.A + self.B
        print("A+B=" +str(self.hasil))
    def kurang(self):
        self.hasil = self.A - self.B
        print("A-B=" +str(self.hasil))
    def kali(self):
        self.hasil = self.A * self.B
        print("AxB=" +str(self.hasil))
    def bagi(self):
        if self.B==0:
            print ("Pembagian dengan nol")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))

# Program Utama
Objek1 = Kalkulator(1,2)
Objek1.tambah()
Objek1.kurang()
Objek1.kali()
Objek1.bagi()
Objek2 = Kalkulator(2,0)
Objek2.bagi()
```

Ln: 23 Col: 0

Encapsulation

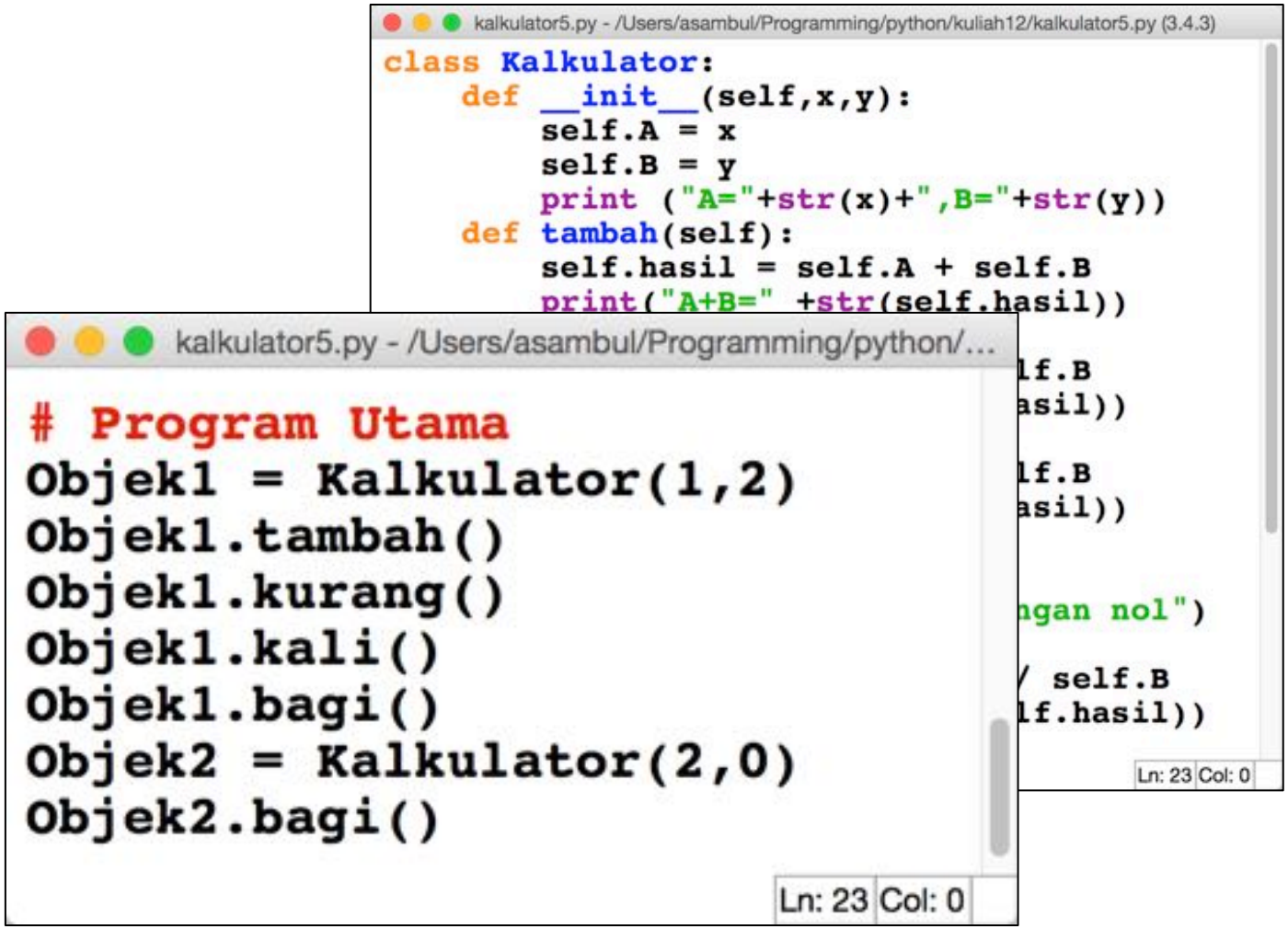
- **Encapsulation**: usaha untuk *memaketkan* **atribut2** dan **metode2 instansi** untuk membentuk sebuah kelas.
- Tujuannya = Information Hiding.

kalkulator5.py - /Users/asambul/Programming/python/kuliah12/kalkulator5.py (3.4.3)

```
class Kalkulator:
    def __init__(self,x,y):
        self.A = x
        self.B = y
        print ("A="+str(x)+",B="+str(y))
    def tambah(self):
        self.hasil = self.A + self.B
        print("A+B=" +str(self.hasil))
    def kurang(self):
        self.hasil = self.A - self.B
        print("A-B=" +str(self.hasil))
    def kali(self):
        self.hasil = self.A * self.B
        print("AxB=" +str(self.hasil))
    def bagi(self):
        if self.B==0:
            print ("Pembagian dengan nol")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))
```

Ln: 23 Col: 0

Information Hiding



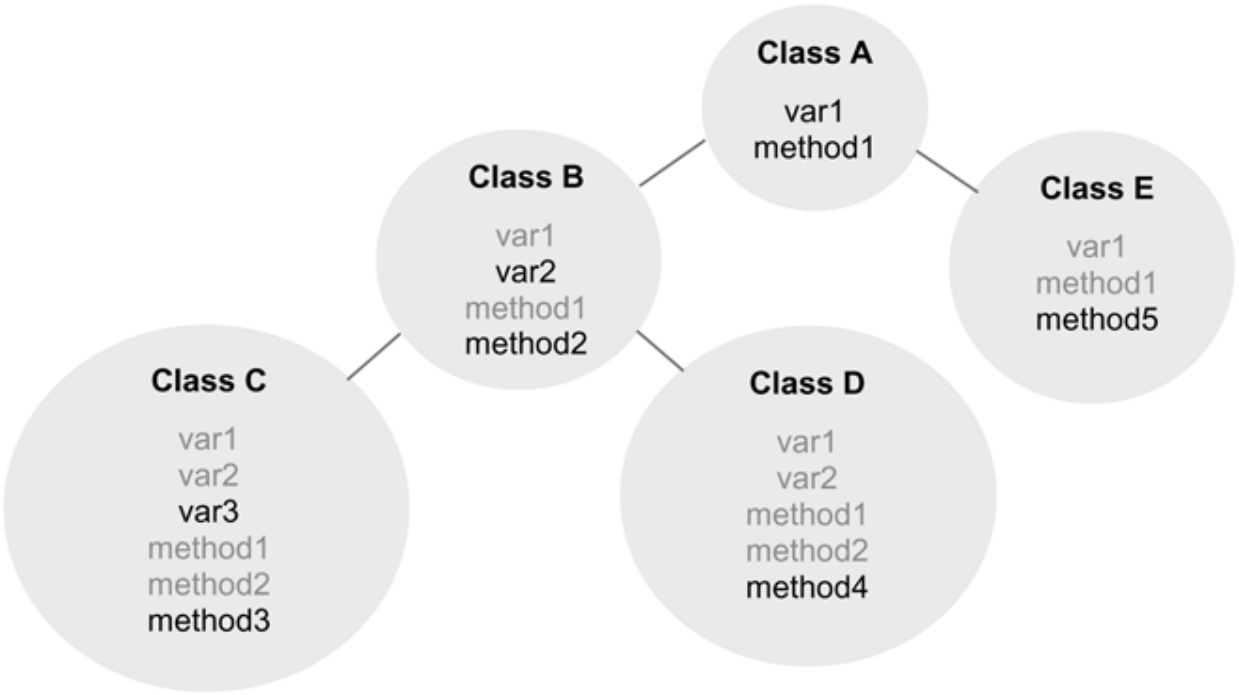
Latihan 1:

- Buatlah kelas “Segiempat” dengan anggota2:
  - Atribut = Panjang, Lebar
  - Metode = HitungLuas, HitungKeliling
- Akses anggota2 kelas tersebut dari program utama

2 Inheritance (Pewarisan)

Inheritance

Inheritance: kemampuan suatu kelas (*superclass*) untuk mewariskan anggotanya ke kelas lain (*subclass*) sebagai bagian definisinya.



Definisi Inheritance

- Bentuk umum:

```
class SubKelas (SuperKelas1[,Superkelas2,...]):
    statemen1
    statemen2
    ......
```

- **SubKelas** akan mewarisi semua anggota dari **SuperKelas**, dan dapat menggunakannya seperti anggotanya sendiri.
- **SubKelas** dapat menambahkan anggotanya sendiri

Contoh:

kalkulator5.py - /Users/asambul/Programming/python/kuliah12/kalkulator5.py (3.4.3)

```
class Kalkulator:
    def __init__(self,x,y):
        self.A = x
        self.B = y
        print ("A="+str(x)+",B="+str(y))
    def tambah(self):
        self.hasil = self.A + self.B
        print("A+B=" +str(self.hasil))
    def kurang(self):
        self.hasil = self.A - self.B
        print("A-B=" +str(self.hasil))
    def kali(self):
        self.hasil = self.A * self.B
        print("AxB=" +str(self.hasil))
    def bagi(self):
        if self.B==0:
            print ("Pembagian dengan nol")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))

class Kalkulator2(Kalkulator):
    def pangkat(self):
        self.hasil = self.A ** self.B
        print("A^B=" +str(self.hasil))

# Program Utama
Objek1 = Kalkulator(1,2)
Objek2 = Kalkulator2(2,0)
Objek2.bagi()
Objek2.pangkat()
```

Ln: 28 Col: 25

Overriding

Overriding = merubah definisi metode yang diwariskan dari SuperClass dalam SubClass.

```
class Kalkulator2(Kalkulator):
    def pangkat(self):
        self.hasil = self.A ** self.B
        print("A^B=" +str(self.hasil))
    def bagi(self):
        if self.B==0:
            print ("A/B = Tidak bisa")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))

# Program Utama
Objek1 = Kalkulator(1,2)
Objek2 = Kalkulator2(2,0)
Objek2.bagi()
```

Metode Khusus __str__

- **__str__** digunakan sebagai representasi string dari objek.
- Kita dapat melakukan overriding terhadap metode ini, misalnya untuk menampilkan informasi mengenai objek.

Contoh:

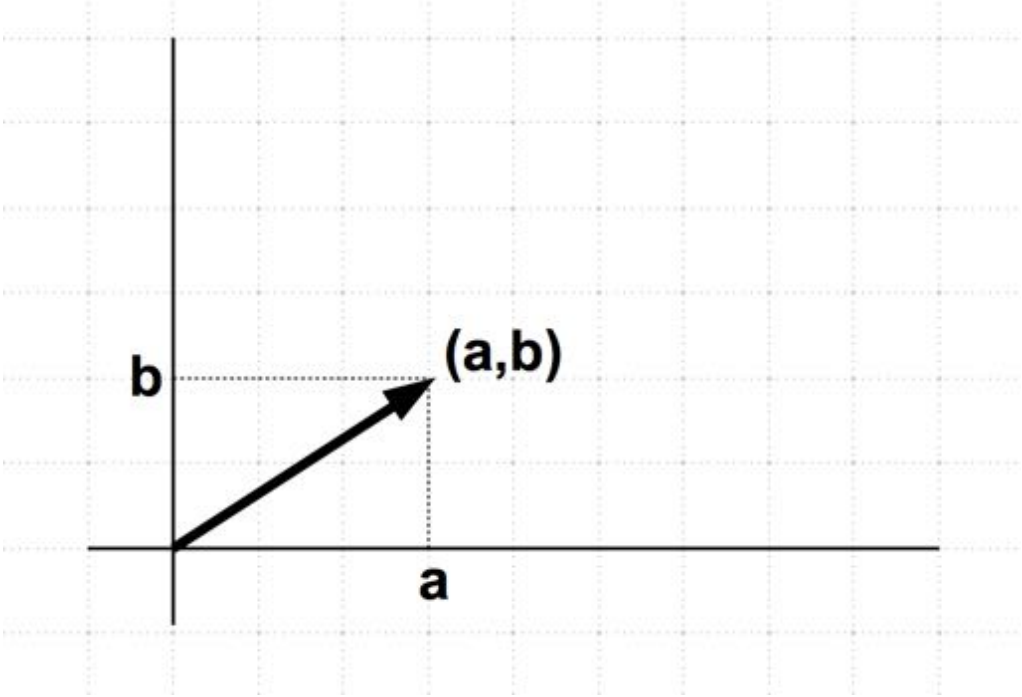
kalkulator7.py - /Users/asambul/Programming/python/kuliah13/kalkulator7.py (3.4.3)

```
    def bagi(self):
        if self.B==0:
            print ("Pembagian dengan nol")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))
    def __str__(self):
        s = "Saya SuperKelas Kalkulator"
        return s

class Kalkulator2(Kalkulator):
    def pangkat(self):
        self.hasil = self.A ** self.B
        print("A^B=" +str(self.hasil))
    def bagi(self):
        if self.B==0:
            print ("A/B = Tidak bisa")
        else:
            self.hasil = self.A / self.B
            print("A/B=" +str(self.hasil))
    def __str__(self):
        s = "Saya SubKelas Kalkulator"
        return s

# Program Utama
Objek1 = Kalkulator(1,2)
print(Objek1)
Objek2 = Kalkulator2(2,0)
print(Objek2)
```

Ln: 37 Col: 8

Latihan:

- Buatlah kelas yang mendefinisikan **vektor**

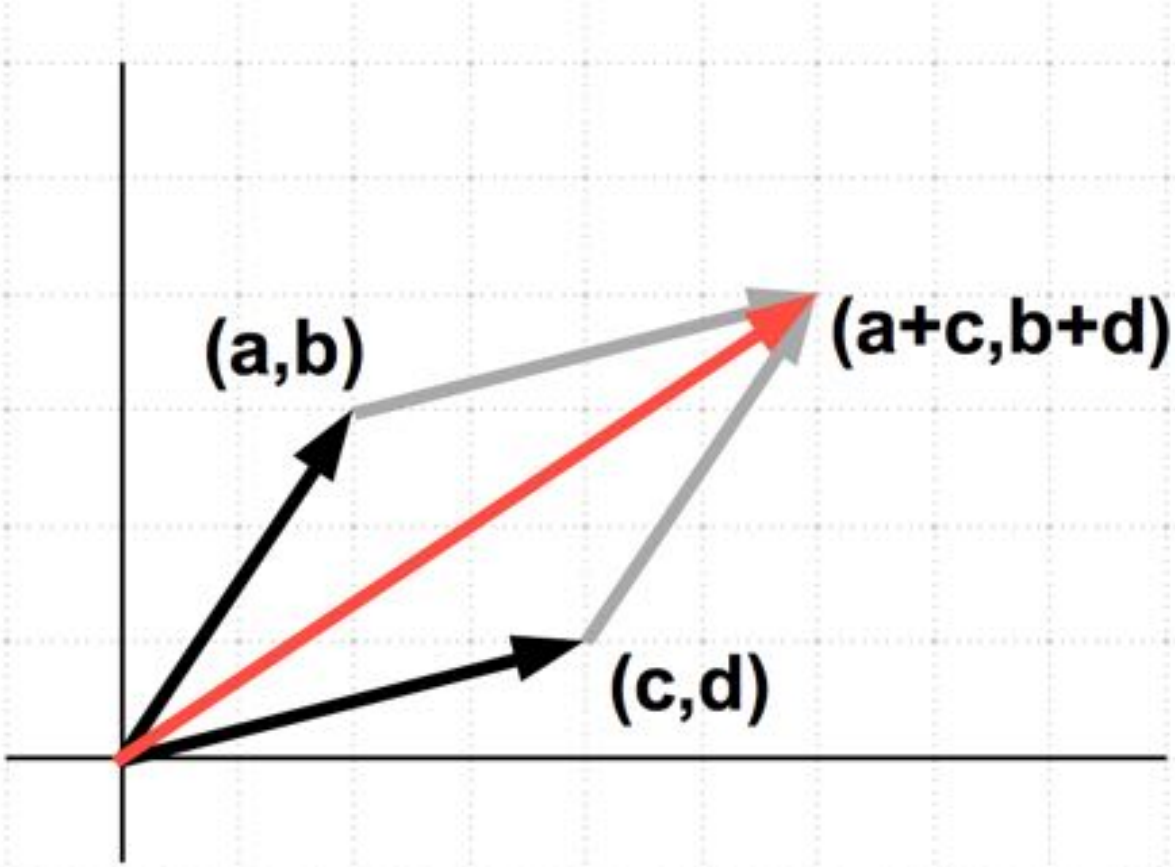


- Override metode khusus __str__ untuk menampilkan informasi vektor

Operator Overloading

- **Operator Overloading** = Overloading metode2 khusus yang dapat digunakan dengan sintaks operator
- Contoh: Kedua representasi di bawah ini melakukan hal yang sama:
  - Fungsi: tambah(a,b)
  - Sintaks operator “+” : a + b

Contoh kasus: Penjumlahan vektor



Menggunakan _add__

vektor_op_overloading.py - /Users/asambul/Programming/python/kuliah13/vektor_op_overloading.py (3.4.3)

```
class Vektor:
    def __init__(self,a,b):
        self.a = a
        self.b = b
    def __str__(self):
        s = "Vektor "+str(self.a)+","+str(self.b)
        return s
    def __add__(self,other):
        return Vektor(self.a + other.a, self.b + other.b)
# program utama
x = Vektor(2,2)
print(x)
y = Vektor(1,2)
print(y)
z = x + y
print(z)
```

Ln: 2 Col: 15

Latihan 2:

- Edit implementasi kelas Vektor pada slide sebelumnya dengan menambahkan mengimplementasikan pengurangan dua vektor
- Untuk melakukan overloading operator “-”, gunakan metode khusus __sub__

3 Polymorphism (Polimorfisme)

Polymorphism (Polimorfisme)

- "Polymorphism": dari bahasa Yunani yg berarti sesuatu yg memiliki banyak wujud
- **Polymorphism**: mekanisme untuk memungkinkan untuk membuat definisi yg berbeda2 dari sebuah metode di kelas2 yang berbeda2.

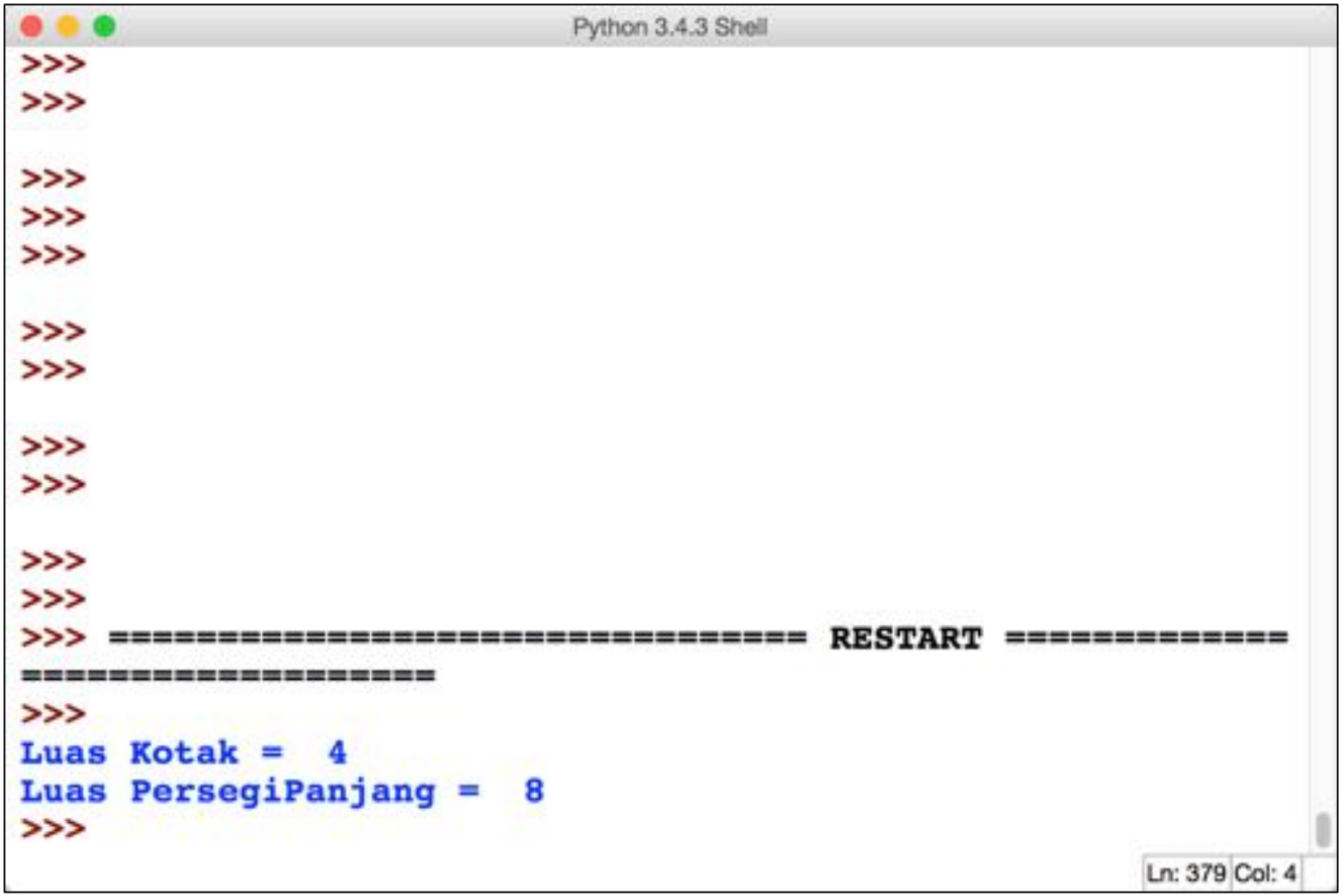
Contoh:

polymorphism.py - /Users/asambul/Programming/python/kuliah12/polymorphism.py (3.4.3)

```
class SegiEmpat:
    def __init__(self,a,b):
        self.panjang = a
        self.lebar = b
    def hitungLuas(self):
        raise NotImplementedError(
            "Metode HitungLuas tidak diimplementasikan")

class Kotak(SegiEmpat):
    def hitungLuas(self):
        return self.panjang ** 2

class PersegiPanjang(SegiEmpat):
    def hitungLuas(self):
        return self.panjang * self.lebar

# program utama
x = Kotak(2,2)
y = PersegiPanjang(2,4)
print("Luas Kotak = ",x.hitungLuas())
print("Luas PersegiPanjang = ",y.hitungLuas())
```

Ln: 15 Col: 0

Keluaran:


Python 3.4.3 Shell
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> ========== RESTART ==========
==========
>>>
Luas Kotak = 4
Luas PersegiPanjang = 8
>>>
Ln: 379 Col: 4

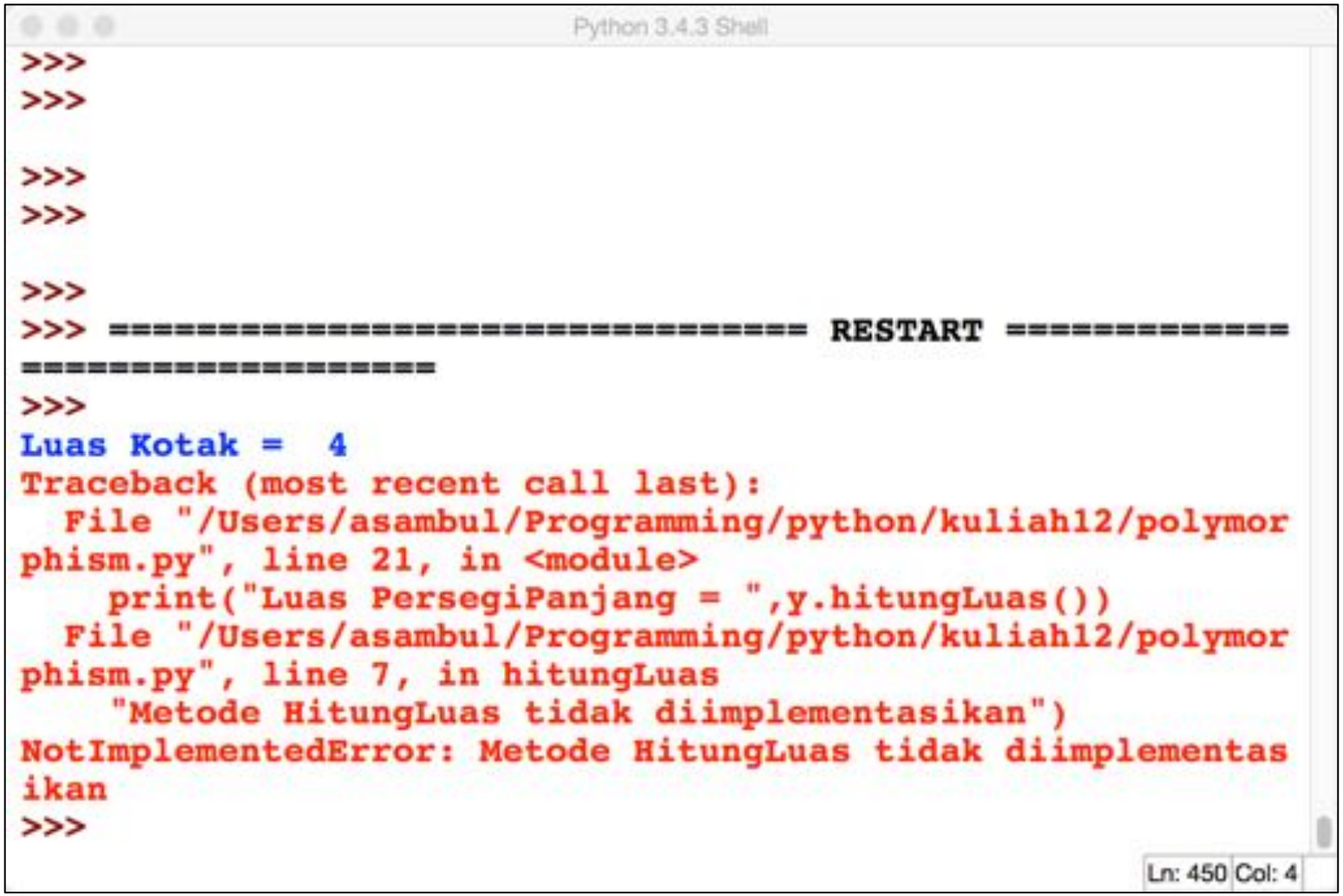
Contoh:

polymorphism.py - /Users/asambul/Programming/python/kuliah12/polymorphism.py (3.4.3)

```
class SegiEmpat:
    def __init__(self,a,b):
        self.panjang = a
        self.lebar = b
    def hitungLuas(self):
        raise NotImplementedError(
            "Metode HitungLuas tidak diimplementasikan")

class Kotak(SegiEmpat):
    def hitungLuas(self):
        return self.panjang ** 2

class PersegiPanjang(SegiEmpat):
    def hitungLuasLain(self):
        return self.panjang * self.lebar

# program utama
x = Kotak(2,2)
y = PersegiPanjang(2,4)
print("Luas Kotak = ",x.hitungLuas())
print("Luas PersegiPanjang = ",y.hitungLuas())
```

Ln: 14 Col: 22

Keluaran:


Python 3.4.3 Shell
>>>
>>>
>>>
>>>
>>>
>>> ========== RESTART ==========
==========
>>>
Luas Kotak = 4
Traceback (most recent call last):
File "/Users/asambul/Programming/python/kuliah12/polymorphism.py", line 21, in <module>
print("Luas PersegiPanjang = ",y.hitungLuas())
File "/Users/asambul/Programming/python/kuliah12/polymorphism.py", line 7, in hitungLuas
"Metode HitungLuas tidak diimplementasikan")
NotImplementedError: Metode HitungLuas tidak diimplementasikan
>>>
Ln: 450 Col: 4