ALGORITMA PEMROGRAMAN

14. Konsep Dasar OOP pada Python

Fakultas Teknik
Universitas Trunojoyo Madura

Pokok Bahasan

1. Pengenalan Prosedural - OOP
2. Konsep Objek & Class
3. Deklasrasi & Operator Class

1) Pendahuluan

Beberapa bahan pembuatan rumah:

1) Batubata
2) Genteng
3) Jendela
4) Paving
5) Besi

Bagaimana jika rumah dibuat dengan pembuatan bahannya 1 per 1?

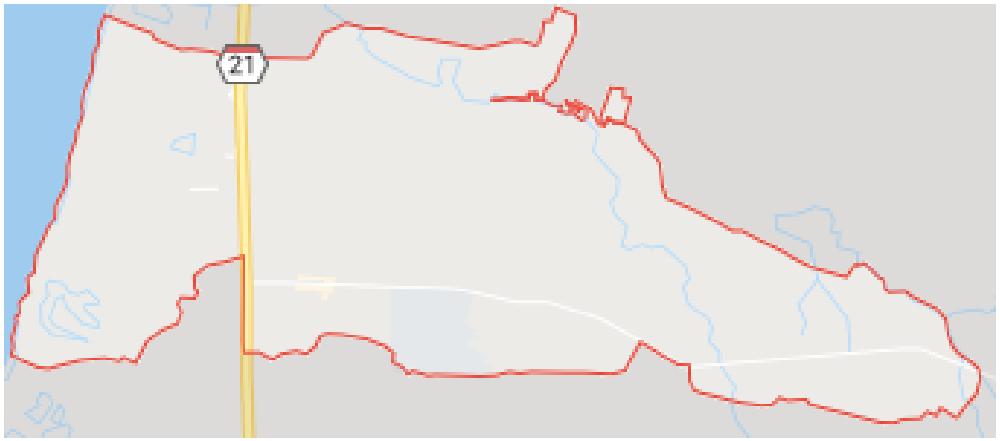
Dibuat satu per satu kah?

Jawabannya pasti, **TIDAK**.

Dalam ukuran yang sama menggunakan cetakan.

Bagaimana jika 1 kompleks perumahan?

Bagaimana jika 1 kompleks perumahan dalam se- …

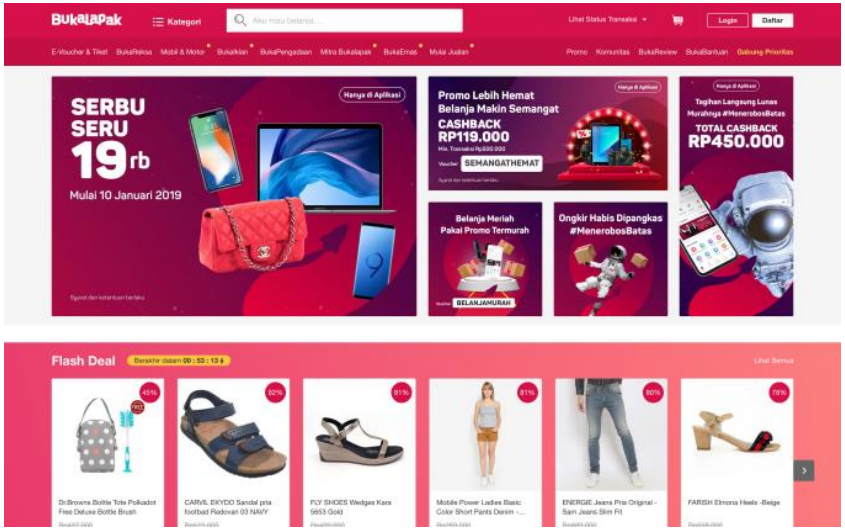


Kec. Telang

Kab Bangkalan

Rumah dan Aplikasi

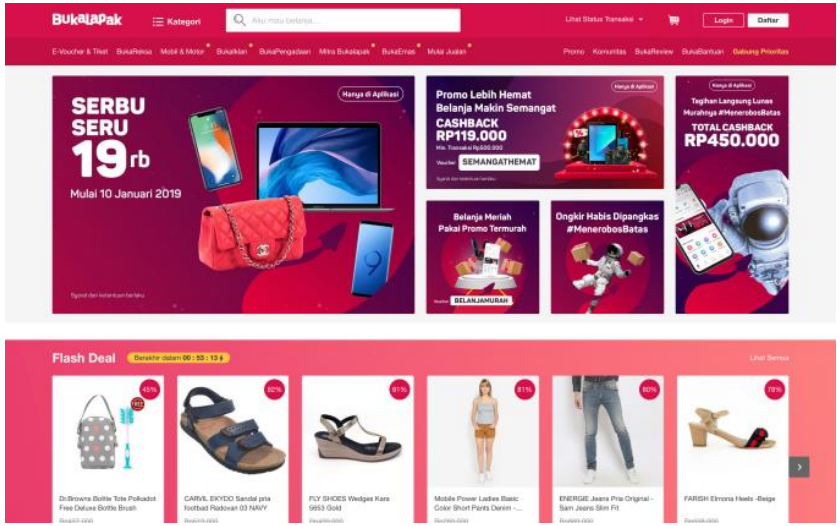
~



1) Batubata
2) Genteng
3) Jendela
4) Paving
5) Besi
6) ….

1) Form handling
2) URI
3) Template
4) …

~

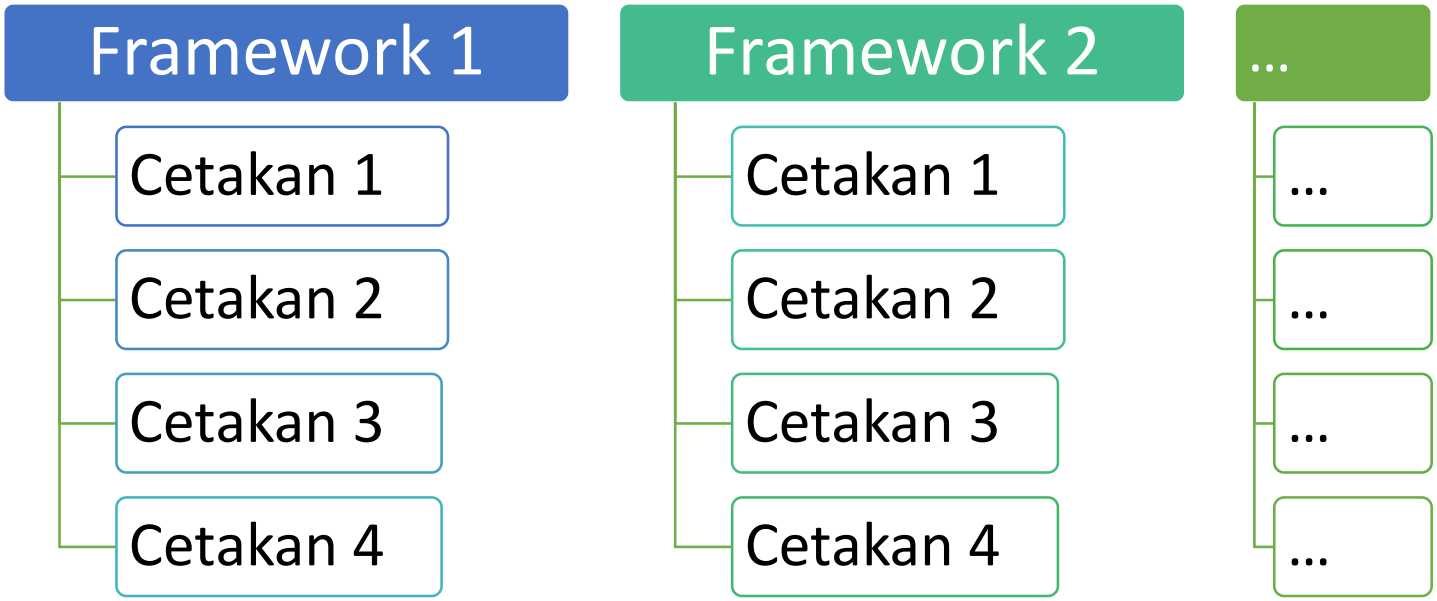


Buat dari nol / buat satu per satu per komponen ~ Gaya programming **Prosedural**.

Buat dari dengan memanfaatkan cetakan-cetakan ~ Gaya programming **OOP/ Framework**.

Framework – OOP

- Framework merupakan kerangka/’cetakan’.
- Pembuatan cetakan menggunakan konsep dari OOP/ *Object Oriented Programming*.
- So, sebelum membuat sebuah kerangka kerja/ ’cetakan’ maka terlebih dahulu harus dipahami dan diterapkan konsep-konsep OOP.



Konsep Procedural – OOP

PROCEDURAL	OOP
❑ Inti dibuatnya program karena pemecahan masalah.	❑ Inti dibuatnya program karena pemecahan masalah.
❑ Procedural merupakan salah satu model pemecahan masalah.	❑ OOP merupakan salah satu model pemecahan masalah.
❑ Procedural menggunakan cara yang algoritmik dalam pemecahan sebuah masalah.	❑ OOP menggunakan pendekatan objek dalam pemecahan masalah dan mampu digunakan dikonsep sejenis.

Analogi Procedural – OOP

PROCEDURAL

Mobil_Ku
1. Kerangka
2. Body
3. Mesin
4. Roda
5. Jok
6. Merk

Mobil_Esti
1. Kerangka
2. Body
3. Mesin
4. Roda
5. Jok
6. Merk

Mobil_Messi
1. Kerangka
2. Body
3. Mesin
4. Roda
5. Jok
6. Merk

Mobil_Ku
1. Maju
2. Mundur
3. Belok
4. Menanjak
5. Menurun

Mobil_Esti
1. Maju
2. Mundur
3. Belok
4. Menanjak
5. Menurun

Mobil_Messi
1. Maju
2. Mundur
3. Belok
4. Menanjak
5. Menurun

OOP

Mobil_Ku

Mobil_Ku

Mobil_Esti

Mobil_Esti

Mobil_Messi

Mobil_Messi

MEMILIKI
1. Kerangka
2. Body
3. Mesin
4. Roda
5. Jok
6. Merk

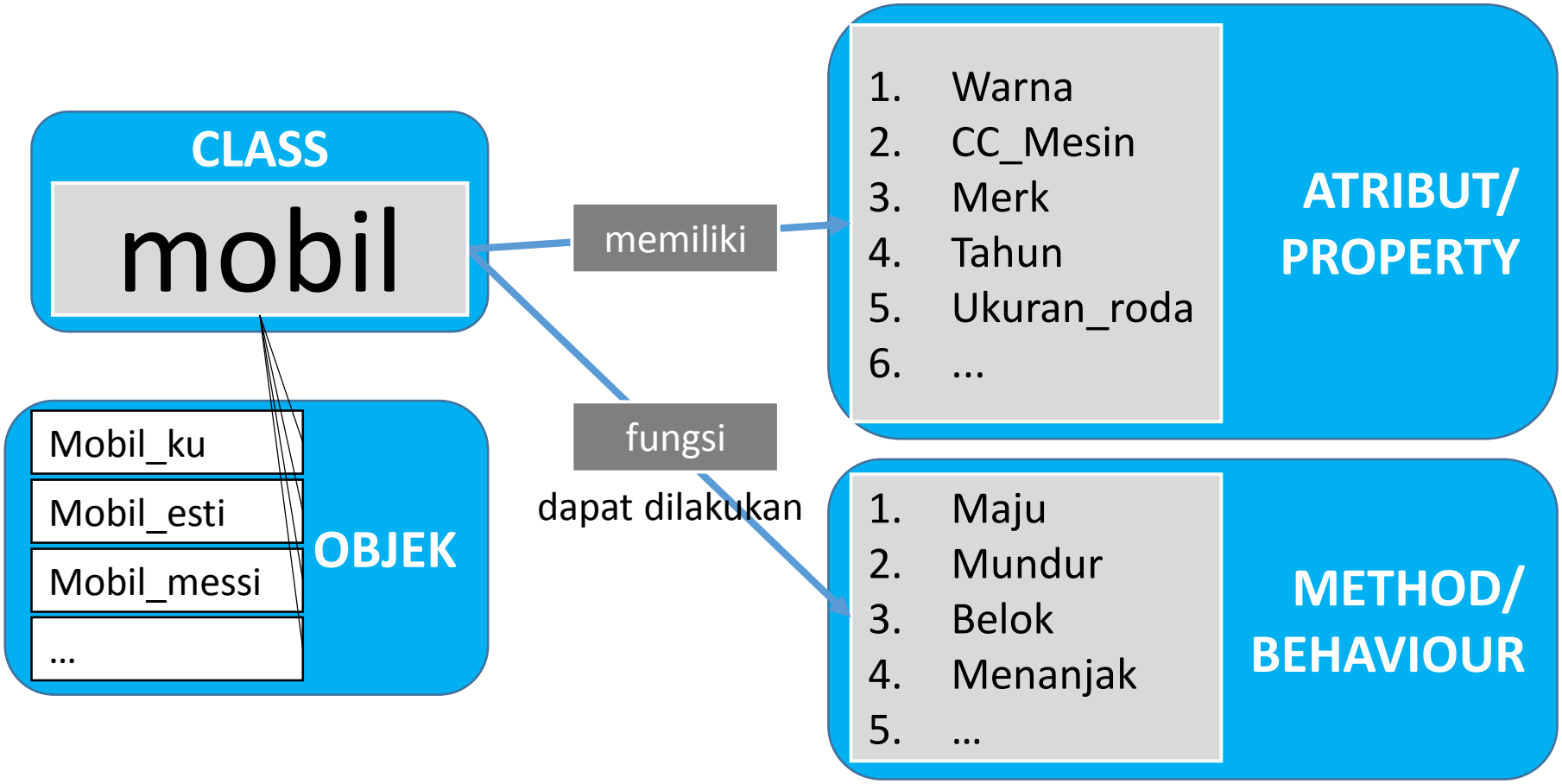
BISA
1. Maju
2. Mundur
3. Belok
4. Menanjak
5. Menurun

MOBIL

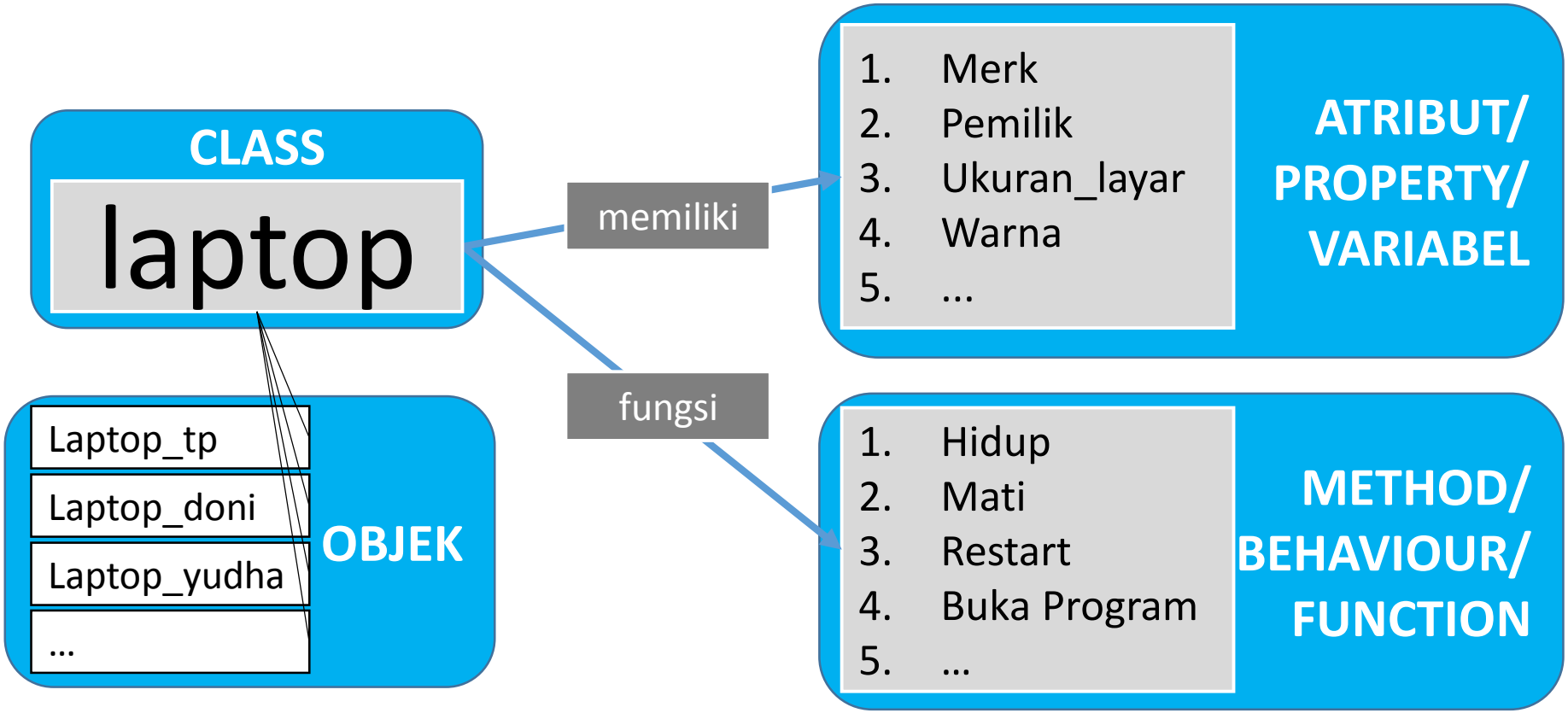
Persamaan dan Perbedaan Procedural – OOP

HAL YANG DIPERTIMBANGKAN	PROCEDURAL	OOP
Kegunaan akhir →	Untuk mengembangkan program	Untuk mengembangkan program
Kontribusi →	Metode/cara berfikir	Metode/cara berfikir
Basis Metode →	Algoritmik	Objek
Level Pengguna →	Pemula	Advanced (Menengah Keatas)
Mulai Berkembang →	-	1950
Keefisienan →	Kurang	Maksimal

OOP – Class, Objek, Property, Method



OOP – Class, Objek, Property, Atribut



OOP – Dalam Python

- Ciri dari OOP di Python dan bahasa pemrograman lainya adanya **Class** dan **Object.**
- Class merupakan *blueprint /* kerangka untuk membentuk suatu objek.
- Class bisa juga disebut sebagai template dari suatu objek.
- Class digunakan untuk mendefinisikan objek yang menyimpan data bersama-sama nilai-nilai dan perilaku (behavior)
- Nilai class tidak boleh kosong, gunakan keyword ***pass*** **untuk mengkosongkan sebuah class**
- isi dari class ini adalah atribut-atribut yang terdiri dari data dan fungsi-fungsi/*methods*
- Fungsi/*method* adalah sifat, misalnya fungsi pada class Mahasiswa itu bisa ke kampus, bayar UKT, cari beasiswa, ikut UKM dsb.
- *constructor* menggunakan *method* bawaan Python yang bernama **init**
- Metode **__init__()** adalah metode konstruktor, yaitu metode khusus yang digunakan Python untuk menginisialisasi pembuatan objek dari kelas tersebut.

OOP – Dalam Python

Cara Membuat Class

- Menggunakan kata kunci ***class*** diikuti **nama kelas**, disarankan mengunakan **huruf kapital diawal kata** nama kelas

```
class ClassName:
      class_body/statemen
```

- Setiap *method* harus memiliki parameter **self** (this pada Java) yang artinya *method* tersebut dimiliki dan terdaftar ke *class* tersebut untuk membedakan dari *method* atau fungsi yang ada di luar *class*
- Untuk Mengakses **Method Objek** menggunakan **operator titik (.)**

OOP – Dalam Python

Nama class

```
class Mobil: # template
    pass
```

Nilai pada class

```
mobil1 = Mobil() # object / instance (instansiate)
mobil2 = Mobil()

mobil1.nama="Toyota"
mobil1.warna="Hitam"

mobil2.nama ="Honda"
mobil2.warna="Merah"

print(mobil1)
print("ini adalah atribut : ",mobil1.__dict__)
print("ini ada Object : ",mobil1.nama)
```

```
========== Kelas Mobil ======================
<__main__.Mobil object at 0x002983A0>
ini adalah atribut :  {'nama': 'Toyota', 'warna': 'Hitam'}
ini ada Object :  Toyota
=================================
```

Membuat atribut

Pemanggilan object

OOP – Dalam Python

Penggunaan Magic Keyword : __init__

```
class Mobil: # template
    def __init__(self, inpmerk, inpwarna, inptahun) :
        #instance Variabel
        self.merk = inpmerk
        self.warna = inpwarna
        self.tahun= inptahun


mobil1 = Mobil ("Toyota"," Hitam", 2010)
mobil2 = Mobil ("Honda", "Merah", 2020)
mobil3 = Mobil ("Kijang", "Putih", 2000)

print("ini adalah atribut : ",mobil1.__dict__)
print("ini adalah atribut : ",mobil2.__dict__)
print("ini adalah atribut : ",mobil3.__dict__)
```

```
========== Kelas Mobil ======================
ini adalah atribut :  {'merk': 'Toyota', 'warna': ' Hitam', 'tahun': 2010}
ini adalah atribut :  {'merk': 'Honda', 'warna': 'Merah', 'tahun': 2020}
ini adalah atribut :  {'merk': 'Kijang', 'warna': 'Putih', 'tahun': 2000}
==================================
```

OOP – Dalam Python

Method/ Behaviour/ Function

Setiap *method* harus memiliki parameter **self**

```
class Sepeda:
  def __init__(self, speed):
    self.gear = 5
    self.speed = speed

  def melaju(self):
    print("Kecepatan sepeda saat ini: ", self.speed)

sepeda = Sepeda(50)
sepeda.gear = 10
print(sepeda.gear)
print(sepeda.speed)
sepeda.melaju()
```

```
10
50
Kecepatan sepeda saat ini:  50
```

5) Referensi

Referensi (1)

- Pemrograman berorientasi Object, Jogiyanto HM, Andi Offset, 1998
- Kadir, Abdul. Dasar Pemrograman Java TM 2. Andi Offset. Yogyakarta. 2004.
- Java for Dummies, Barry Burd, Wiley Publishing, 2007
- Java 6 in 21 Days, Rogers Cadenhead, SAMS, 2007
- Object Oriented Programming in 21 Days, Tony Sintes, SAMS, 2002
- Head First Java, Kathy Sierra & Bert Bates, O'Reilly, 2005
- belajaroracle.com
- academy.oracle.com