PEMROGRAMAN BERORIENTASI OBYEK

By : Sendi Novianto, S.Kom, M.T
Email : to_sendi@yahoo.com

Materi Diskusi

1. Konsep Pemrograman dan Paradigmanya
   Tingkat Bahasa Pemrograman, Paradigma Pemrograman, Tool Pemrograman
2. Konsep Dasar Pemrograman Berorientasi Obyek:
   Kelas (Class), Obyek (Object), Pengkapsulan (Encapsulation), Pewarisan (Inheritance), Polymorphisme
3. Konsep Lanjut Pemrograman Berorientasi Obyek:
   Constructor, Destructor, Overloading, Overriding, Keyword This, Multithreading, Package, Interface, Exception

Tujuan Pembelajaran

1. Pemahaman tentang paradigma pemrograman
2. Pemanfaatan paradigma pemrograman untuk dapat menyelesaikan permasalahan dalam program
3. Penentuan tools yang tepat sesuai dengan paradigma yang dipakai

Konsep Pemrograman dan Paradigmanya

Bahasa Pemrograman?

- Komputer bekerja seperti switching dan hanya mengenali 0 dan 1
- Manusia tidak (paham) berbicara dengan bahasa 0 dan 1
- Perlu bahasa pemrograman yang dapat menjadi perantara percakapan antara komputer dan manusia
- Bahasa pemrograman diubah ke dalam bahasa yang dipahami oleh komputer dengan menggunakan interpreter atau kompiler

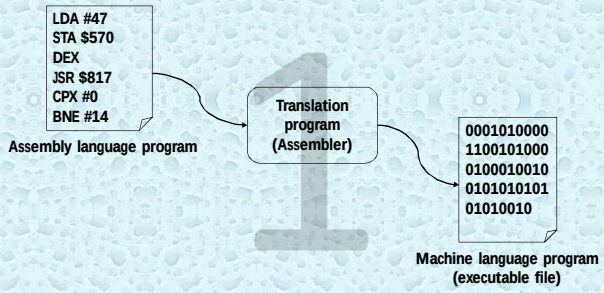
Compiler or Interpreter?

- Compiler:
  Mengkompilasi source code menjadi bentuk file yang bisa dieksekusi
- Interpreter:
  Mengkompilasi dan menjalankan source code secara langsung

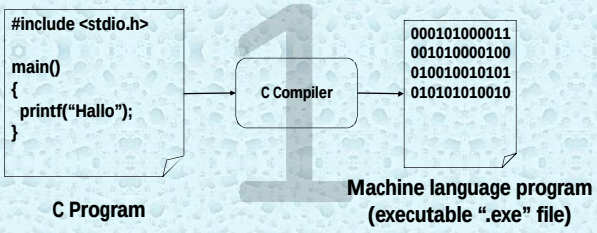
Tingkat Bahasa Pemrograman

- Bahasa Pemrograman Tingkat Rendah (Assembler)
- Bahasa Pemrograman Tingkat Sedang (C, Pascal, Fortran)
- Bahasa Pemrograman Tingkat Tinggi (Java, C++, C#)

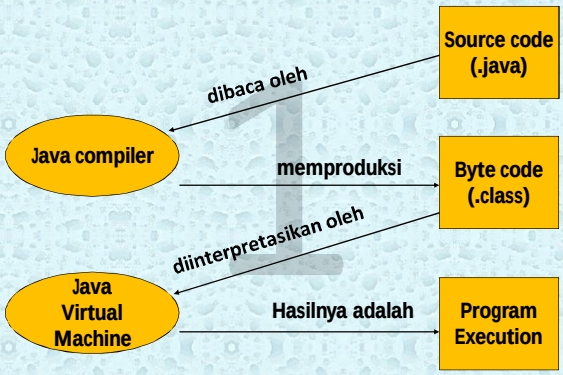
Assembly Language



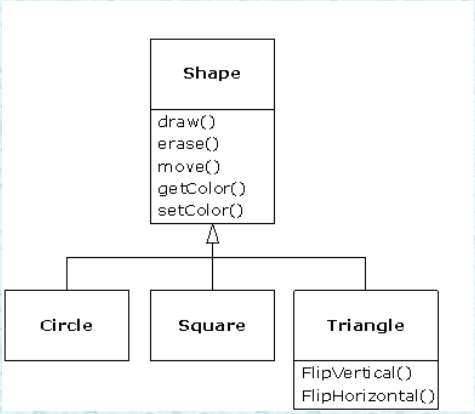
C Language



Java Language



Java Language



Paradigma Pemrograman

Sudut pandang dan style pemrograman berhubungan dengan bagaimana sebuah masalah diformulasikan dalam bahasa pemrograman

- Functional Programming: Urutan fungsi secara sekuensial (Scheme, Lisp)
- Procedural Programming: Pemecahan masalah berdasarkan prosedural kerja. Fungsi adalah unit programming (C, Pascal, Fortran)
- Object-Oriented Programming: Koleksi object yang saling berinteraksi . Class adalah unit programming (Java, C#, C++)

Functional Programming

```
Terminal
Window  Edit  Options                    Help
Yes Master~xlispstat convert.ls
XLISP-PLUS version 3.0
Portions Copyright (c) 1988, by David Betz.
Modified by Thomas Almy and others.
XLISP-STAT Release 3.50 (Beta).
Copyright (c) 1989-1994, by Luke Tierney.

; loading convert.ls
> (convert)
Enter Fahrenheit 200
(CELSIUS IS . 93.333333333333334)
> (convert)
Enter Fahrenheit -200
(CELSIUS IS . -128.88888888888889)
> (convert)
Enter Fahrenheit 98
(CELSIUS IS . 36.66666666666667)
>
```


Procedural Programming

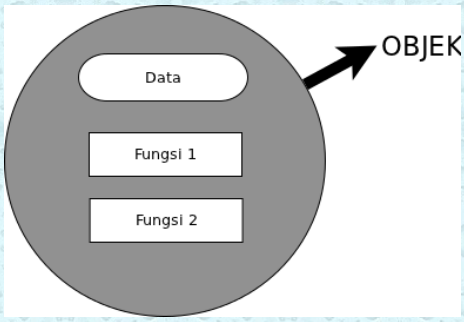
```
abstract class Customer {
    def plan

    def Customer() {
        plan = □
    }

    def abstract checkCredit()
    def abstract checkInventory()
    def abstract ship()

    def process() {
        checkCredit()
        checkInventory()
        ship()
    }
}
```


OOP



Konsep Dasar Pemrograman Berorientasi Obyek

Class & Object