

PEMROGRAMAN BERORIENTASI OBJEK

PENDAHULUAN

Rosa Ariani Sukamto

ROSA ARIANI SUKAMTO

Blog: <http://hariiniadalahhadiah.wordpress.com>

Facebook: <https://www.facebook.com/rosa.ariani.sukamto>

Email: rosa_if_itb_01@yahoo.com



TERSTRUKTUR VS OBJEK

dalam satu file

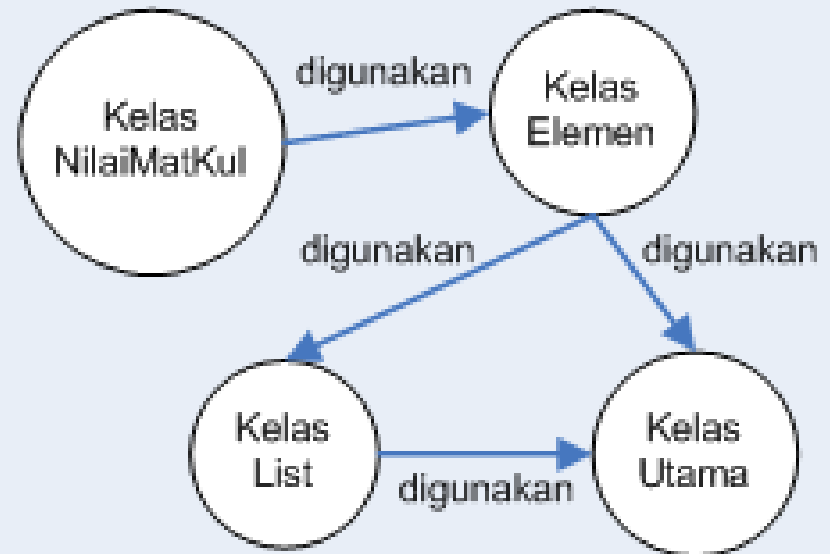
Struktur NilaiMatKul

Struktur Elemen

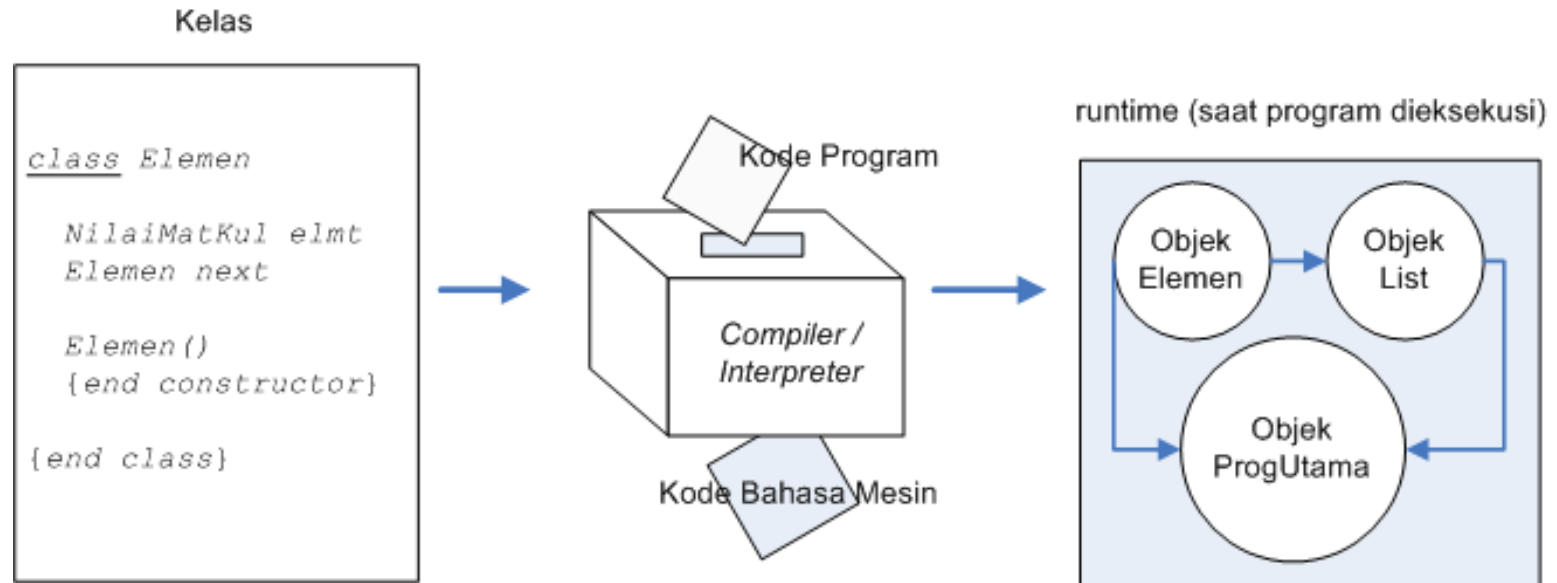
Struktur List

Program Utama

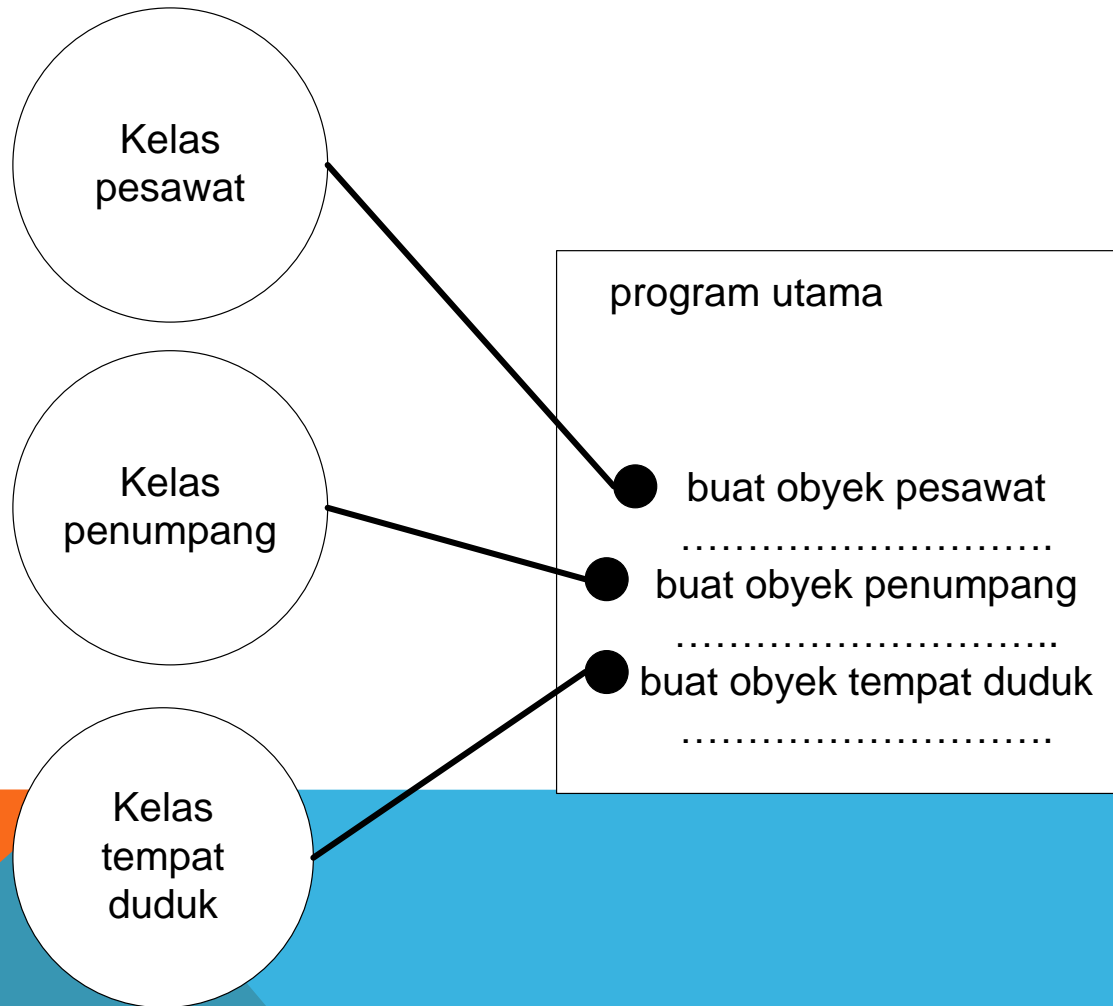
dalam satu direktori



PEMROGRAMAN BERORIENTASI OBJEK (1)



PEMROGRAMAN BERORIENTASI OBJEK (2)



MENGENAL KELAS PADA BAHASA C++ (1)

```
#include <iostream>
#include <string>

using namespace std;

class Halo{
    private:
        string kata;

    public:
        Halo(){
            kata = "Halo Dunia";
        }
        void tulis(){
            cout << kata << endl;
        }
        ~Halo(){
        }

};
```

```
#include "Halo.cpp"

int main(){
    Halo objekHalo;
    objekHalo.tulis();

    return 0;
}
```

MENGENAL KELAS PADA BAHASA C++ (2)

File kelas dan main.cpp berada di dalam satu direktori

Dalam satu direktori, hanya boleh ada 1 main

Kompilasi pada MinGW

```
g++ -c *.cpp
```

Eksekusi pada MinGW

```
g++ main.cpp -o halo.exe
```

MENGENAL KELAS PADA BAHASA PHP 4

```
<?php

class Halo{

    var $kata = "";

    function Halo(){
        $this->kata = "Halo
Dunia";
    }

    function tulis(){
        echo $this-
>kata."<br/>";
    }
}

?>
```

```
<?php

    include "Halo.php";

$halo = new Halo();
$halo->tulis();

?>
```


MENGENAL KELAS PADA BAHASA JAVA (1)

```
class Halo{  
  
    public static void main(String[] args){  
  
        String kata;  
  
        kata = "Halo Dunia";  
        System.out.println(kata);  
  
    }  
}
```

MENGENAL KELAS PADA BAHASA JAVA (2)

Path pada JDK

```
Path="C:\Program Files\Java\jdk\bin";%path%
```

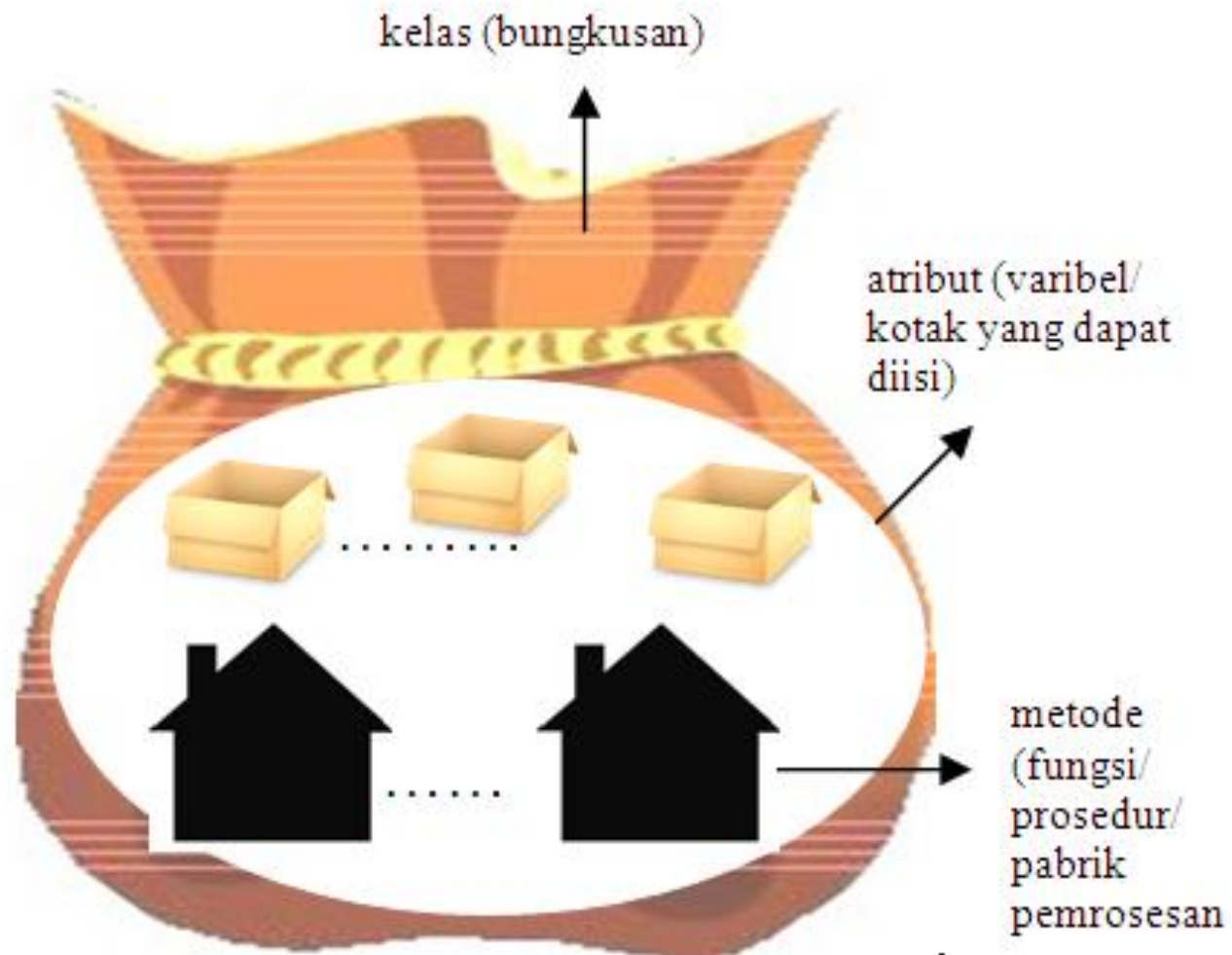
Kompilasi

```
javac *.java
```

Eksekusi

```
java Halo
```

ENKAPSULASI



ATRIBUT

```
#include <iostream>
#include <string>

using namespace std;

class Halo{
    private:
        string kata;

    public:
        .....
        ~Halo(){
        }

};
```

```
<?php

class Halo{

    var $kata = "";

    function Halo(){

        $this->kata = "Halo
        Dunia";

    }

    function tulis(){

        echo $this->kata."<br/>";

    }

}

?>
```

AKSES ATRIBUT PRIVATE (1)

Atribut tidak dapat diakses langsung oleh kelas lain yang menggunakan kelas pemilik atribut



ATRIBUT PRIVATE (2) - C++

```
#include <iostream>
#include <string>
using namespace std;

class Buku{
    private:
        string judul;
        string pengarang;

    public:
        Buku() {
        }

        Buku(string j, string p){
            judul = j;
            pengarang = p;
        }
};
```

```
void setJudul(string j){
    judul = j;
}

void setPengarang (string p){
    pengarang = p;
}

string getJudul(){
    return judul;
}

string getPengarang(){
    return pengarang;
}

~Buku() {
}

};
```

ATRIBUT PRIVATE (3) - C++

```
#include "Buku.cpp"

int main() {
    Buku b1;
    b1.setJudul("J2ME");
    b1.setPengarang("Orang_1");

    cout << b1.getJudul() << endl;
    cout << b1.getPengarang () << endl;

    Buku b2("J2ME", "Orang_2");
    cout << b2.getJudul() << endl;
    cout << b2.getPengarang () << endl;

    return 0;
}
```

ATRIBUT PRIVATE (4) - PHP 4

```
<?php

class Buku{

    var $judul = "";
    var $pengarang = "";

    function Buku($j="",
        $p="") {
        $this->judul = $j;
        $this->pengarang = $p;
    }

    function setJudul($j="") {
        $this->judul = $j;
    }
}
```

```
function setPengarang($p="") {
    $this->pengarang = $p;
}

function getJudul() {
    return $this->judul;
}

function getPengarang() {
    return $this->pengarang;
}

?>
```


ATRIBUT PRIVATE (5) - PHP 4

```
<?php
    include "Buku.php" ;

    $b2 = new Buku("J2EE", "Orang_1") ;
    echo $b2->getJudul() . "<br/>" ;
    echo $b2->getPengarang() . "<br/>" ;

?>
```

ATRIBUT PRIVATE (6) - JAVA

```
class Buku{  
    private String judul;  
    private String  
        pengarang;  
  
    Buku() {  
    }  
  
    Buku(String j, String p){  
        this.judul = j;  
        this.pengarang = p;  
    }  
  
    void setJudul(String j){  
        this.judul = j;  
    }  
}
```

```
    void setPengarang(String p){  
        this.pengarang = p;  
    }  
  
    String getJudul(){  
        return this.judul;  
    }  
  
    String getPengarang(){  
        return this.pengarang;  
    }  
}
```

ATRIBUT PRIVATE (7) - JAVA

```
class Main{  
    public static void main(String[] args){  
        Buku b1;  
        Buku b2;  
  
        b1 = new Buku();  
        b1.setJudul("J2ME");  
        b1.setPengarang("Orang_1");  
        System.out.println(b1.getJudul());  
        System.out.println(b1.getPengarang());  
  
        b2 = new Buku("J2EE", "Orang_2");  
        System.out.println(b2.getJudul());  
        System.out.println(b2.getPengarang());  
    }  
}
```

AKSES ATRIBUT PUBLIC

**Atribut dapat diakses langsung oleh kelas lain
yang menggunakan kelas pemilik atribut**



METODE (METHOD)

Metode atau *method* pada sebuah kelas hampir sama dengan fungsi atau prosedur pada pemrograman prosedural

Pada sebuah metode di dalam sebuah kelas juga memiliki izin akses seperti halnya atribut pada kelas, izin akses itu antara lain private, public, dan protected yang memiliki arti sama pada izin akses atribut yang telah dibahas sebelumnya.

KONSTRUKTOR

Sebuah kelas harus memiliki sebuah metode yang disebut sebagai konstruktor.

Nama sebuah konstruktor harus sama dengan nama dari sebuah kelas, misalkan kelas Buku maka konstruktornya adalah Buku().

Sebuah konstruktor juga dapat menerima sebuah masukan seperti halnya prosedur pada pemrograman prosedural. Fungsi dari sebuah konstruktor adalah :

- mengalokasikan sebuah objek saat program dieksekusi (memerintahkan dibuatnya alokasi objek di memori saat program dijalankan)
- memberikan nilai awal sebagai inisialisasi dari semua atribut yang perlu diinisialisasi
- mengerjakan proses-proses yang diperlukan saat sebuah objek dibuat


namun pada kenyataannya sebuah konstruktor dapat tidak berisi apa-apa, hal ini jika memang tidak diperlukan adanya inisialisasi atau proses yang dikerjakan ketika sebuah objek dibuat.

Konstruktor harus bersifat public karena sebuah konstruktor akan diakses oleh kelas lain untuk membuat objek suatu kelas.

DESTRUKTOR

Destruktor adalah metode yang dipanggil secara otomatis ketika objek dihancurkan.

Sebuah destruktur tidak harus ada pada kode program sebuah kelas jika *compiler* atau *interpreter* tidak memiliki *garbage collection* (mekanisme membersihkan alokasi objek-objek yang sudah tidak terpakai dari memori).



KELAS (1) - C++

```
class Titik{
    /*kelas yang digunakan untuk
       mengimplementasikan sebuah tipe
       titik*/

private:
    int x; /*koordinat x*/
    int y; /*koordinat y*/

public:
    Titik() {
        /*konstruktor*/
        x = 0;
        y = 0;
    }

    Titik(int xp, int yp){
        /*konstruktor*/
        x = xp;
        y = yp;
    }

    void setX(int xp){
        /*mengeset nilai koordinat x*/
        x = xp;
    }
```

```
        int getX(){
            /*mengembalikan nilai
            koordinat x*/
            return x;
        }

        void setY(int yp){
            /*mengeset nilai
            koordinat y*/
            y = yp;
        }

        int getY(){
            /*mengembalikan nilai
            koordinat y*/
            return y;
        }

        ~Titik(){
            /*destruktor*/
        }
```

```
};
```


KELAS (2) - C++

```
#include <iostream>
#include <Titik.cpp>

using namespace std;
/*fungsi main untuk mengetes kelas Titik*/

int main(){

    Titik t1;
    Titik t2(11, 9);

    t1.setX(18);
    t1.setY(28);

    cout << "t1 : nilai X :" << t1.getX() << endl;
    cout << "t1 : nilai Y :" << t1.getY() << endl;

    cout << "t2 : nilai X :" << t2.getX() << endl;
    cout << "t2 : nilai Y :" << t2.getY() << endl;

    return 0;

}
```

KELAS (3) - PHP 5

```
<?php
class Titik{
    /*kelas yang digunakan untuk
    mengimplementasikan sebuah
    tipe titik*/
    private $x; /*koordinat x*/
    private $y; /*koordinat y*/
    public function __construct(){
        /*konstruktor*/
        $this->x = 0;
        $this->y = 0;
    }
    public function setX($xp){
        /*mengeset nilai koordinat
        x*/
        $this->x = $xp;
    }
}
```

```
public function getX(){
    /*mengembalikan nilai
    koordinat x*/
    return $this->x;
}
public function setY($yp){
    /*mengeset nilai
    koordinat y*/
    $this->y = $yp;
}
public function getY(){
    /*mengembalikan nilai
    koordinat y*/
    return $this->y;
}
function __destruct(){
    /*destruktor*/
}
}
?>
```

KELAS (4) - PHP 5

```
<?php
    include "Titik.php";

    /*main untuk mengetes kelas Titik*/
    $t1 = new Titik();

    $t1->setX(18);
    $t1->setY(28);

    echo "t1 : nilai X : "
        . $t1->getX() . "<br/>";
    echo "t1 : nilai Y : "
        . $t1->getY() . "<br/>";

?>
```

KELAS (5) - JAVA

```
class Titik{
/*kelas yang digunakan untuk
mengimplementasikan sebuah
tipe titik*/

private int x; /*koordinat x*/
private int y; /*koordinat
y*/

Titik(){
/*konstruktor*/
x = 0;
y = 0;
}

Titik(int xp, int yp){
/*konstruktor*/
x = xp;
y = yp;
}

public void setX(int xp){
/*mengeset nilai koordinat
x*/
x = xp;
}
```

```
public int getX(){
/*mengembalikan nilai
koordinat x*/
return x;
}

public void setY(int yp){

/*mengeset nilai koordinat
y*/
y = yp;
}

public int getY(){
/*mengembalikan nilai
koordinat y*/
return y;
}

public void finalize(){
/*destruktor*/

}
}
```

KELAS (6) - JAVA

```
class Main{  
    /*metode main untuk mengetes kelas Titik*/  
  
    public static void main(String[] args) {  
        Titik t1 = new Titik();  
        Titik t2 = new Titik(11, 9);  
  
        t1.setX(18);  
        t1.setY(28);  
  
        System.out.println("t1 : nilai X : " + t1.getX());  
        System.out.println("t1 : nilai Y : " + t1.getY());  
  
        System.out.println("t2 : nilai X : " + t2.getX());  
        System.out.println("t2 : nilai Y : " + t2.getY());  
  
    }  
}
```

DAFTAR PUSTAKA

S, Rosa A. dan M. Shalahuddin. 2011. Modul Pembelajaran: Pemrograman Berorientasi Objek. Modula: Bandung.

