Lembar Praktikum 03

* Mata Kuliah : Pemrograman Berorientasi Obyek
* Pokok Bahasan : Tentang Methods
* Tujuan Pembelajaran :
  + Siswa dapat menjelaskan, membuat, dan menerapkan jenis method berdasarkan jumlah argumen
  + Siswa dapat menjelaskan, membuat, dan menerapkan jenis method berdasarkan sifat (void dan non-void)
  + Siswa dapat menjelaskan, membuat, dan menerapkan jenis method berdasarkan aksesibilitasnya

Misalkan diberikan sebuah studi kasus sebagai berikut:

Suatu rumah terdiri dari beberapa ruang. Selanjutnya rumah tersebut akan dipasang ubin berbentuk persegi panjang dengan ukuran tertentu. Dengan asumsi bahwa setiap ruang berbentuk persegi panjang, akan dihitung banyaknya ubin yang diperlukan. Dalam hal ini, kita diminta untuk membuat program Java nya.

# Kegiatan 1

Secara umum ide penyelesaian dari kasus di atas adalah menghitung terlebih dahulu luas setiap ruangan yang masing-masingnya berbentuk persegi panjang, kemudian ditotal jumlah luasnya. Selanjutnya total luas ruangan dibagi dengan luas ubin yang juga berbentuk persegi panjang.

Berikut ini adalah langkah penyelesaiannya:

1. Buat project baru dengan nama ‘ProjectPengubinan’ sekaligus main class nya
2. Buat class ‘PersegiPanjang’ di dalam package ‘projectpengubinan’. Class ini nantinya akan digunakan untuk menghitung luas ruang dan ubin yang sama-sama berbentuk persegi panjang.
3. Silakan modifikasi class ‘PersegiPanjang ’sebagai berikut

public class PersegiPanjang {

// atribut

int panjang;

int lebar;

// method

int hitungLuas(){

int luas = this.panjang \* this.lebar;

return luas;

}

}

1. Buat class ‘Ruang’ di dalam package ‘projectpengubinan’. Class ini nantinya akan digunakan untuk membuat obyek ruang yang ada di dalam rumah, serta menghitung luasnya
2. Silakan modifikasi class ‘Ruang ’sebagai berikut

public class Ruang {  
 // panjang ruang

int panjang;  
 // lebar ruang

int lebar;

int hitungLuas(){

// menghitung luas dengan memanfaatkan class persegi panjang

PersegiPanjang p = new PersegiPanjang();

// panjang dari persegi panjang diambil dari panjang ruang

p.panjang = this.panjang;

// lebar dari persegi panjang diambil dari lebar ruang

p.lebar = this.lebar;

// hitung luasnya

return p.hitungLuas();

}

}

1. Buat class ‘Ubin’ di dalam package ‘projectpengubinan’. Class ini nantinya akan digunakan untuk membuat obyek ubin yang akan digunakan, serta menghitung luasnya
2. Silakan modifikasi class ‘Ubin’

public class Ubin {

// panjang ubin

int panjang;

// lebar ubin

int lebar;

// menghitung luas sebuah ubin

int hitungLuas(){

...

}

}

Dengan konsep yang sama dengan class ruang, lengkapilah method hitungLuas() pada class Ubin supaya perhitungan luasnya memanfaatkan class PersegiPanjang yang telah dibuat sebelumnya.

1. Selanjutnya dari beberapa class yang telah dibuat, kita akan gunakan untuk menghitung jumlah ubin pada kasus berikut ini.

Sebuah rumah terdapat 3 buah ruang berbentuk persegi panjang, dengan masing-masing ukuran sebagai berikut:

* Ruang A: panjang = 3 meter, lebar = 3 meter
* Ruang B: panjang = 5 meter, lebar = 2 meter
* Ruang C: panjang = 4 meter, lebar = 3 meter

Seluruh ruang tersebut akan dipasangi ubin merk X yang berukuran 40 cm x 40 cm.

Berapakah jumlah ubin yang diperlukan untuk keperluan di atas?

1. Untuk mencari solusi dari permasalahan di atas, kita akan gunakan class-class yang sudah dibuat tadi untuk perhitungannya. Dalam hal ini kita tuliskan kode programnya di main class.

public class ProjectPengubinan {

public static void main(String[] args) {

// hitung luas setiap ruang

Ruang A = new Ruang();

A.panjang = 3;

A.lebar = 3;

Ruang B = new Ruang();

B.panjang = 5;

B.lebar = 2;

Ruang C = new Ruang();

C.panjang = 4;

C.lebar = 3;

// hitung luas total ruang dalam m2

int totalLuasRuang = A.hitungLuas() + B.hitungLuas() + C.hitungLuas();

// hitung luas ubin

Ubin X = new Ubin();

X.panjang = 40;

X.lebar = 40;

// nyatakan luas ubin ke m2

double luasUbin = X.hitungLuas()/100000.;

// hitung jumlah ubin

double jumlahUbin = totalLuasRuang/luasUbin;

System.out.println("Banyaknya ubin yang diperlukan: " + jumlahUbin);

}

}

1. Coba jalankan programnya, dan amati hasilnya.

# Pertanyaan

1. Menurut Anda, jenis method yang didefinisikan di class ‘PersegiPanjang’, ‘Ruang’, dan ‘Ubin’ itu termasuk yang non-void atau void? Jelaskan!

Menurut saya, jenis method diatas termasuk non-void, karena terdapat return pada class ‘PersegiPanjang’, ‘Ruang’, dan ‘Ubin’.

1. Menurut Anda, mengapa dalam kasus di atas method-method tersebut berbentuk non-void? Jelaskan!

Karena terdapat return pada class tersebut. Dan method tersebut menghasilkan sebuah nilai untuk proses perhitungan selanjutnya.

1. Menurut Anda, bisakah dalam kasus di atas method-method tersebut berbentuk void? Jelaskan!

Method void berfungsi tidak mencari nilai atau mengembalikan nilai hasil, sehingga method tidak bisa berhubungan. Dengan hal itu method void tidak bisa digunakan.

# Kegiatan 2

1. Sekarang kita akan lakukan sedikit modifikasi pada class PersegiPanjang sbb:

public class PersegiPanjang {

// method

int hitungLuas(int panjang, int lebar){

int luas = panjang \* lebar;

return luas;

}

}

1. Perubahan tersebut adalah menghapus atributnya, namun sebagai gantinya adalah menambahkan argumen pada method hitungLuas(). Dampak dari pengubahan tersebut, selanjutnya lakukan modifikasi juga pada method hitungLuas() pada class ‘Ruang’

public class Ruang {

int panjang;

int lebar;

int hitungLuas(){

// menghitung luas dengan memanfaatkan class persegi panjang

PersegiPanjang p = new PersegiPanjang();

// hitung luasnya

return p.hitungLuas(this.panjang, this.lebar);

}

}

1. Lakukan hal yang sama untuk method hitungLuas() pada class ‘Ubin’
2. Kemudian coba jalankan programnya dan amati hasilnya!

# Pertanyaan

1. Apakah setelah modifikasi tersebut dihasilkan output yang sama? Jelaskan alasannya!

Iya, output yang dihasilkan sama. Karena jumlah yang dicari sama tapi hanya berbeda pada pengerjaannya yaitu pada saat yang pertama menggunakan jenis method non void dan yang kedua menghapus atribut dengan menambahkan argument.

1. Menurut Anda, apa kelebihan dari perubahan jenis method pada kasus di atas, yaitu dengan memindahkan atribut menjadi argumen dari method?

Menurut saya, jika dengan memindahkan atribut menjadi argument dari method lebih simple, tetapi tergantung dari keinginan programmer lebih nyaman menggunakan jenis method apa.

1. Menurut Anda, apa kekurangan dari perubahan jenis method pada kasus di atas, yaitu dengan memindahkan atribut menjadi argumen dari method?

Menurut saya, memang jika dengan memindahkan atribut menjadi argument dari method terlihat simple, tetapi dari saya sendiri merasakan bahwa lebih nyaman dengan menggunakan jenis method non void. Karena lebih rinci dan jelas letak atribut maupun methodnya.

1. Menurut Anda, sebaiknya bagaimana karakteristik argumen yang akan dipilih untuk sebuah method dalam sebuah class?

Menurut saya, menggunakan jenis method yang sama, tetapi tidak dengan argumen yang sama pada sebuah class.

# Kegiatan 3

1. Selanjutnya kita akan coba memodifikasi lagi class ‘PersegiPanjang’ menjadi seperti berikut ini

public class PersegiPanjang {

// method untuk mengalikan dua bilangan a dan b

static int perkalian(int a, int b){

int hasil = a \* b;

return hasil;

}

// method

int hitungLuas(int panjang, int lebar){

int luas = perkalian(panjang, lebar);

return luas;

}

}

1. Kemudian coba jalankan programnya, dan amati hasilnya!

# Pertanyaan

1. Apakah setelah pengubahan tersebut hasilnya sama?

Iya, hasil progam diatas sama.

1. Selanjutnya cobalah untuk memodifikasi class ‘Ruang’ khususnya pada method hitungLuas() di mana untuk menghitung luasnya perkalian panjang dan lebarnya menggunakan method perkalian() dari class ‘PersegiPanjang’, yaitu

public class Ruang {

int panjang;

int lebar;

int hitungLuas(){

// menghitung luas dengan memanfaatkan class persegi panjang

PersegiPanjang p = new PersegiPanjang();

// hitung luasnya

return p.perkalian(this.panjang, this.lebar);

}

}

1. Perhatikan di Netbeans, apakah ada semacam tanda warning di baris return nya? Jika ada, cobalah klik pesan warningnya tersebut, maka nanti secara otomatis Netbeans akan melakukan koreksi pada baris yang salah tersebut!
2. Amati perubahan pada kode!

# Pertanyaan

1. Apa kesimpulan Anda tentang static method terkait dengan cara mengaksesnya pada class itu sendiri?

Static method dapat diakses pada kelas tanpa harus menulis nama class itu sendiri.

1. Apa kesimpulan Anda tentang static method terkait dengan cara mengaksesnya dari class yang berbeda?

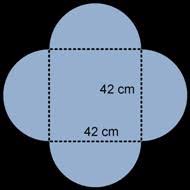
Static method tidak dapat digunakan secara langsung pada class berbeda kecuali static.

1. Apa kesimpulan Anda tentang perbedaan antara static method dan non static method?

Static method tidak membutuhkan referensi dari outer class, sedangkan non static method membutuhkan referensi outer class.

# Project 1

Buatlah program Java untuk menghitung luas bangun datar berikut ini

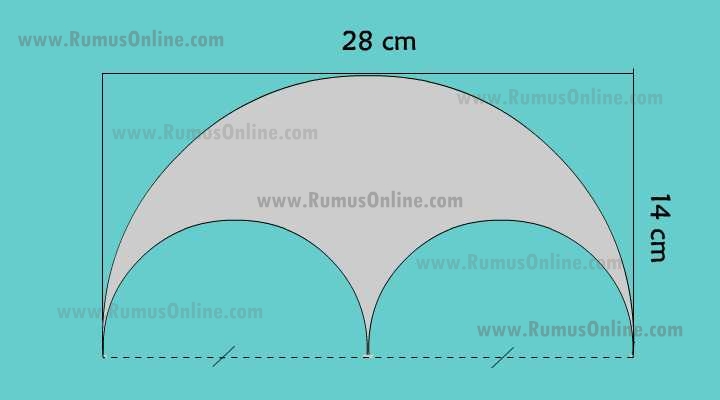


**Petunjuk**:

1. Beri nama projectnya ‘ProjectLuasBangunKompleks’
2. Buatlah class ‘Lingkaran’ yang di dalamnya terdapat method hitungLuas() untuk menghitung luas lingkaran berjari-jari r
3. Buatlah class ‘Persegi’ yang di dalamnya terdapat method hitungLuas() untuk menghitung luas persegi dengan sisi s
4. Silakan tentukan sendiri jenis method yang akan digunakan, apakah dengan argumen atau tanpa argumen, apakah termasuk void atau non-void, dan apakah static atau non-static

# Project 2

Buatlah program Java untuk bisa menghitung luas daerah yang berwarna abu-abu ini



Ketentuan:

1. Gunakan project ‘ProjectLuasBangunKompleks’ dan class-class yang telah dibuat sebelumnya
2. Jika perlu, Anda bisa menambah class lagi yang baru sesuai ide penyelesaian masing-masing
3. Silakan tentukan sendiri jenis method yang akan digunakan, apakah dengan argumen atau tanpa argumen, apakah termasuk void atau non-void, dan apakah static atau non-static