Modul 10 PBO Teori\_Ridhatul Nafsi

Membuat Class Sendiri

Dalam pendefinisian class, dituliskan :

<modifier> class <name> {

<attributeDeclaration>\*

<constructorDeclaration>\*

<methodDeclaration>\*

}

dimana : adalah sebuah access modifier, yang dapat dikombinasikan dengan tipe modifier lain.

Instance Variable

Jika kita menginginkan bahwa atribut – atribut tersebut adalah unik untuk setiap object (dalam hal ini untuk setiap siswa), maka kita harus mendeklarasikannya sebagai instance variable : Sebagai contoh :

public class StudentRecord

{

private String name;

private String address;

private int age;

private double mathGrade;

private double englishGrade;

private double scienceGrade;

private double average;

//area penulisan kode selanjutnya

}

dimana, private disini menjelaskan bahwa variabel tersebut hanya dapat diakses oleh class itu sendiri. Object lain tidak dapat menggunakan variabel tersebut secara langsung.

Contoh dari penggunaan accessor method adalah getAverage,

public class StudentRecord

{

private String name;

:

:

public double getAverage(){

double result = 0;

result = ( mathGrade+englishGrade+scienceGrade )/3;

return result;

}

}

Method getAverage() menghitung rata – rata dari 3 nilai siswa dan menghasilkan nilai return value dengan nama result.

Contoh Kode Program dari class StudentRecord

Berikut ini adalah kode untuk class StudentRecord :

public class StudentRecord

{

private String name;

private String address;

private int age;

private double mathGrade;

private double englishGrade;

private double scienceGrade;

private double average;

private static int studentCount;

/\*\*

\* Menghasilkan nama dari Siswa

\*/

public String getName(){

return name;

}

/\*\*

\* Mengubah nama siswa

\*/

public void setName( String temp ){

name = temp;

}

// area penulisan kode lain

/\*\*

\* Menghitung rata – rata nilai Matematik, Bahasa Inggris, \* \* Ilmu

Pasti

\*/

public double getAverage(){

double result = 0;

result = ( mathGrade+englishGrade+scienceGrade )/3;

return result;

}

/\*\*

\* Menghasilkan jumlah instance StudentRecord

\*/

public static int getStudentCount(){

return studentCount;

}

}

contoh kode dari class yang mengimplementasikan class StudentRecord :

public class StudentRecordExample

{

public static void main( String[] args ){

//membuat 3 object StudentRecord

StudentRecord annaRecord = new StudentRecord();

StudentRecord beahRecord = new StudentRecord();

StudentRecord crisRecord = new StudentRecord();

//Memberi nama siswa

annaRecord.setName("Anna");

beahRecord.setName("Beah");

crisRecord.setName("Cris");

//Menampilkan nama siswa “Anna”

System.out.println( annaRecord.getName() );

//Menampilkan jumlah siswa

System.out.println("Count="+StudentRecord.getStudentCount()

);

}

}

Output dari program adalah sebagai berikut : Anna Student Count = 0

Overloading method di dalam deklarasi

class StudentRecord.

public void print( String temp ){

System.out.println("Name:" + name);

System.out.println("Address:" + address);

System.out.println("Age:" + age);

}

public void print(double eGrade, double mGrade, double sGrade)

System.out.println("Name:" + name);

System.out.println("Math Grade:" + mGrade);

System.out.println("English Grade:" + eGrade);

System.out.println("Science Grade:" + sGrade);

}

Jika kita panggil pada method utama (main) :

public static void main( String[] args )

{

StudentRecord annaRecord = new StudentRecord();

annaRecord.setName("Anna");

annaRecord.setAddress("Philippines");

annaRecord.setAge(15);

annaRecord.setMathGrade(80);

annaRecord.setEnglishGrade(95.5);

annaRecord.setScienceGrade(100);

//overloaded methods

annaRecord.print( annaRecord.getName() );

annaRecord.print( annaRecord.getEnglishGrade(),

annaRecord.getMathGrade(),

annaRecord.getScienceGrade());

}

Kita akan mendapatkan output pada panggilan pertama sebagai berikut :

Name:Anna

Address:Philippines

Age:15

Kemudian akan dihasilkan output sebagai berikut pada panggilan kedua :

Name:Anna

Math Grade:80.0

English Grade:95.5

Science Grade:100.0

overloaded method memiliki property sebagai berikut :

Nama yang sama

Parameter yang berbeda

Nilai kembalian (return) bisa sama ataupun berbeda

Thank you !

Pembuat : Ridhatul Nafsi