1. Jelaskan perbedaan metode accessor (getter) dan metode mutator (setter)!

Jawaban :

-Accessor method istilahnya getter (pengambil), yaah dia berarti method yang berfungsi mengambil/memanggil data.

-Mutator method istilahnya setter (pengatur), berarti dia method yang berfungsi memodifikasi/set data.

2. Jelaskan secara rinci perbedaan modifier default, public, private dan protected!

Jawaban :

Modifier atau pengenal adalah sifat yang dimiliki oleh suatu atribut atau method di Java.

Java menyediakan 2 macam tipe modifier, yaitu:

Access Modifier

Non-Access Modifier

\*Java Access Modifier

Mengenal Access Modifier di Java

Ada 4 Macam Acces Modifier di Java sesuai gambar di atas, yaitu:

-Public

Kelas, method maupun atribut yang mempunyai access modifier public dapat diakses dari manapun dan oleh kelas apapun.

Dengan modifier public membuat member tersebut dapat diakses dari mana saja

Contoh:

package example;

class Person {

            public String name;

            public changeName(String newName) {

                        this.name = newName;

            }

}

Pada class Person diatas terdapat dua member, yaitu:

1.         Property name

2.         Method changeName()

Pada kedua member itu diberikan modifier public yang berarti kedua member tersebut dapat diakses dari mana saja. Tetapi, dikarenakan class Person tidak diberikan modifier public maka class Person hanya bisa diakses(import) oleh class yang berada pada package yang sama, dan tidak akan bisa diakses oleh class yang berbeda package kecuali class Person telah diberi modifier public seperti ini:

package example;

public class Person {

            public String name;

            public changeName(String newName) {

                        this.name = newName;

            }

}

 -Protected

Atribut dan method yang mempunyai access modifier protected, artinya bahwa atribut dan method tersebut dapat diakses oleh kelas yang sama, package yang sama, dan kelas turunannya (subclass).

Dengan modifier protected, kita dapat mengakses member dalam sebuah class melalui class itu sendiri, class turunannya, dan class yang berada dalam satu package. Modifier protected, seperti private hanya bisa diberikan pada member dan tidak bisa diberikan pada class.

Contoh:

package example;

public class Person {

            protected String name;

            public setName(String newName) {

                        this.name = newName;

            }

            Public getName() {

                        Return this.name;

            }

}

Dari contoh diatas, kita memberikan modifier protected pada property name. Jika kita mencoba mengakses pada class yang berada pada package yang sama maka tidak akan terjadi error, namun jika diakses dari class yang berbeda package maka akan terjadi error.(Athallah)

-Default / no access

Atribut dan method yang mempunyai access modifier default, artinya bahwa atribut dan method tersebut dapat diakses oleh kelas yang sama dan package yang sama.

-Private

Atribut dan method yang mempunyai access modifier default, artinya bahwa atribut dan method tersebut dapat diakses oleh kelas yang sama dan package yang sama.

Dengan modifier private, member hanya dapat diakses oleh class itu sendiri. Perlu diingat bahwa modifier private tidak bisa diberikan pada class, hanya bisa diberikan pada membernya.

Contoh:

class Person {

            private String name;

            public setName(String newName) {

                        this.name = newName;

            }

            Public getName() {

                        Return this.name;

            }

}

Pada contoh diatas, property name kita berikan modifier private sehingga property tersebut tidak dapat diakses kecuali class itu sendiri. Jadi, untuk dapat mengakses property dengan modifier private dibuatlah method setter dan getter dan diberikan modifier public. Seperti contoh diatas, dibuat sebuah method setter dan juga getter. Pada contoh diatas, method setter-nya adalah setName() dan method getter-nya adalah getName().

Contoh Program

Public class Manusia{

    //bersifat private agar dapat diakses hanya kelas Manusia

    Private String nama;

    //bersifat protected agar dapat diakses oleh subclass nya

    Protected String jenisKelamin;

    Public Manusia(String nama, String jenisKelamin){

        Nama = nama;

        jenisKelamin = jenisKelamin;

    }

    Public void setNama(String nama){

       This.nama = nama;

    }

    Public String getNama(){

      Return nama;

    }

    Public void setJenisKelamin(String jenisKelamin){

      This.jenisKelamin = jenisKelamin;

    }

    Public String tampilkan(){

       Return(“Nama : “+nama +”\nJenis Kelamin : “+jenisKelamin);

    }

}

\*Non Access Modifier

Java menyediakan beberapa macam Non-Access Modifier, yaitu:

-Static

Static adalah salah satu jenis modifier di Java yang digunakan agar suatu atribut atau pun method dapat diakses oleh kelas atau objek tanpa harus melakukan instansiasi terhadap kelas tersebut.

Method main adalah salah satu contoh method yang mempunyai modifier static.

-Final

Final adalah salah satu modifier yang digunakan agar suatu atribut atau method bersifat final atau tidak bisa diubah nilainya. Modifier ini digunakan untuk membuat konstanta di Java.

-Abstract

Abstract adalah modifier yang digunakan untuk membuat kelas dan method abstrak

-Synchronized

Synchronized adalah modifier yang digunakan dalam aplikasi Java berbasis thread. Modifier ini menspesifikasikan bahwa method merupakan thread safe. Artinya bahwa hanya ada satu jalur eksekusi pada method yang menggunakan modifier jenis ini dan memaksa thread thread lain menunggu giliran.

-Native

Modifier Native digunakan untuk spesifikasi method dengan implementasi di bahasa lain, seperti C, C++.

-Transient

Modifier ini digunakan agar suatu variable tidak bisa di serialisasi. Serialization adalah konsep dimana sebuah objek dapat ditransfer dari suatu aplikasi ke aplikasi lainnya atau dari suatu workstation ke workstation lainnya. Konsep ini sangat diperlukan ketika membuat aplikasi client server.

Salah satu tujuan serialization adalah bahwa tidak boleh ada perubahan terhadap atribut pada saat objek di transformasikan menjadi stream.

-Volatile

-Implements

-Extends

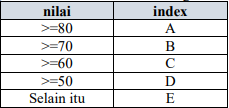
3. Buatlah program dengan ketentuan dibawah ini :

1. Terdapat satu buah kelas dengan modifier default

2. Terdapat atribut sebanyak 4 buah, yang terdiri dari : nim, nama dan kelas yang bertipe data String serta nilai yang bertipe data double.

3. Buah metode accessor dan mutator untuk masing-masing atribut diatas.

4. Buat metode getIndex() untuk mencari index nilai dengan ketentuan seperti ini :

[](https://www.blogger.com/u/1/blog/post/edit/4310209025248791298/341131789979222896)

5. Buat metode main() yang akan memanggil semua metode yang sudah dibuat.

6. Metode main() akan memanggil semua metode setter dengan memberi value pada setiap pemanggilan tersebut.

Contoh :

[](https://www.blogger.com/u/1/blog/post/edit/4310209025248791298/341131789979222896)

Contoh Output:

[](https://www.blogger.com/u/1/blog/post/edit/4310209025248791298/341131789979222896)

Jawaban :

import java.util.Scanner;

public class Ilman10519069\_Soal3{

     public static void main(String[] args) {

          Scanner input = new Scanner(System.in);

          Ilman10519069 data = new Ilman10519069();

          String nim;String nama;String kelas;double nilai;

          /\*

          System.out.print("Input NIM   : ");nim = input.nextLine();

          System.out.print("Input Nama  : ");nama = input.nextLine();

          System.out.print("Input Kelas : ");kelas = input.nextLine();

          System.out.print("Input Nilai : ");nilai = input.nextDouble();

          data.setNim(nim);

          data.setNama(nama);

          data.setKelas(kelas);

          data.setNilai(nilai);

          \*/

          data.setNim("10505050");

          data.setNama("Cantika Anggun");

          data.setKelas("SI-01");

          data.setNilai(75);

          //

          System.out.println("NIM   : "+ data.getNim());

          System.out.println("Nama  : "+ data.getNama());

          System.out.println("Kelas : "+ data.getKelas());

          System.out.println("Nilai : "+ data.getNilai());

          System.out.println("Return "+ data.getIndex());

}

}

public class Ilman10519069{

     private String nim;private String nama;private String kelas;private double nilai;

     public String getNim(){return nim;}public String getNama(){return nama;}public String getKelas(){return kelas;}public double getNilai(){return nilai;}

     public void setNim(String nim){this.nim = nim;}public void setNama(String nama){this.nama = nama;}public void setKelas(String kelas){this.kelas = kelas;}public void setNilai(double nilai){this.nilai = nilai;}

     public double getIndex(){

          if(nilai>=80){System.out.println("Index : A");}

          else if(nilai>=70){System.out.println("Index : B");}

          else if(nilai>=60){System.out.println("Index : C");}

          else if(nilai>=50){System.out.println("Index : D");}

          else {System.out.println("Index : E");}

          System.out.println("");System.out.println("");System.out.println("");

          return(0);

     }

}