

Nama: Arditya Adjie Rosandi  
NIM : 20230801274  
Pemrograman Berorientasi Objek



## 1. Program MhsTester

```
java Salin kode  
  
package Pertemuan7;  
  
class Mahasiswa {  
    // Atribut dan metode akan dijelaskan di bawah ini  
}
```

**Package Declaration:** package Pertemuan7; menunjukkan bahwa kelas Mahasiswa berada dalam package Pertemuan7.

**Class Declaration:** class Mahasiswa mendefinisikan sebuah kelas bernama Mahasiswa. Tidak ada modifier akses yang diberikan, sehingga secara default kelas ini memiliki akses *package-private*, artinya hanya bisa diakses oleh kelas lain dalam package yang sama

```
java Salin kode  
  
public String nama;           // Public: bisa diakses dari mana saja  
protected int usia;          // Protected: bisa diakses dari dalam package dan subclass  
private String jurusan;      // Private: hanya bisa diakses dalam kelas ini
```

**public String nama:**

- **Modifier Akses:** public
- **Deskripsi:** Atribut nama dapat diakses dari mana saja, baik dari dalam kelas Mahasiswa, kelas lain dalam package Pertemuan7, maupun dari luar package.

**protected int usia:**

- **Modifier Akses:** protected
- **Deskripsi:** Atribut usia dapat diakses dari dalam package Pertemuan7 dan oleh subclass yang mungkin meng-extend kelas Mahasiswa di package lain.

**private String jurusan:**

- **Modifier Akses:** private
- **Deskripsi:** Atribut jurusan hanya dapat diakses dari dalam kelas Mahasiswa itu sendiri. Tidak bisa diakses langsung dari luar kelas, termasuk kelas dalam package yang sama.

java

Salin kode

```
// Constructor
public Mahasiswa(String nama, int usia, String jurusan) {
    this.nama = nama;
    this.usia = usia;
    this.jurusan = jurusan;
}
```

**Fungsi:** Konstruktor ini digunakan untuk menginisialisasi objek Mahasiswa dengan nilai-nilai awal untuk atribut nama, usia, dan jurusan.

**Parameter:**

- String nama: Nama mahasiswa.
- int usia: Usia mahasiswa.
- String jurusan: Jurusan mahasiswa.

**Penggunaan this:** Kata kunci this digunakan untuk merujuk pada atribut kelas saat terjadi konflik nama antara parameter dan atribut.

java

Salin kode

```
// Getter untuk atribut private jurusan
public String getJurusan() {
    return jurusan;
}

// Setter untuk atribut private jurusan
public void setJurusan(String jurusanBaru) {
    this.jurusan = jurusanBaru;
}
```

**Getter (getJurusan):**

- **Fungsi:** Mengembalikan nilai atribut jurusan.
- **Aksesibilitas:** public, sehingga dapat diakses dari luar kelas untuk mendapatkan nilai jurusan.

**Setter (setJurusan):**

- **Fungsi:** Mengubah nilai atribut jurusan.
- **Parameter:** String jurusanBaru adalah nilai baru yang akan diassign ke atribut jurusan.
- **Aksesibilitas:** public, memungkinkan perubahan nilai jurusan dari luar kelas.

java

Salin kode

```
// Metode untuk menampilkan informasi
public void tampilkanInfo() {
    System.out.println("NAMA : " + nama);
    System.out.println("USIA : " + usia);
    System.out.println("JURUSAN : " + jurusan);
}
```

**Fungsi:** Menampilkan semua informasi mahasiswa (nama, usia, dan jurusan) ke konsol.

**Aksesibilitas:** public, sehingga dapat dipanggil dari luar kelas untuk menampilkan informasi mahasiswa.

java

Salin kode

```
package Pertemuan7;

public class MhsTester {
    public static void main(String[] args) {
        // Isi metode main akan dijelaskan di bawah
    }
}
```

**Package Declaration:** package Pertemuan7; menunjukkan bahwa kelas MhsTester juga berada dalam package Pertemuan7.

**Class Declaration:** public class MhsTester mendefinisikan kelas MhsTester yang bersifat public, sehingga dapat diakses dari luar package jika diperlukan.

**Main Method:** public static void main(String[] args) adalah titik masuk program yang akan dieksekusi saat program dijalankan.

java

Salin kode

```
Mahasiswa mahasiswa1 = new Mahasiswa("ANDI", 21, "TEKNIK INFORMATIKA");
```

**Fungsi:** Membuat instansi (objek) Mahasiswa dengan nama "ANDI", usia 21, dan jurusan "TEKNIK INFORMATIKA".

**Proses:**

- Memanggil konstruktor Mahasiswa dengan parameter yang diberikan.
- Atribut nama, usia, dan jurusan diinisialisasi sesuai dengan parameter.

java

Salin kode

```
System.out.println("NAMA MAHASISWA : " + mahasiswa1.nama); // Output: ANDI
```

**Fungsi:** Mencetak nilai atribut nama dari objek mahasiswa1.

**Aksesibilitas:** Atribut nama bersifat public, sehingga dapat diakses langsung dari luar kelas.

java

Salin kode

```
System.out.println("USIA MAHASISWA : " + mahasiswa1.usia); // Output: 21
```

**Fungsi:** Mencetak nilai atribut usia dari objek mahasiswa1.

**Aksesibilitas:** Atribut usia bersifat protected dan dapat diakses karena kelas MhsTester berada dalam package yang sama (Pertemuan7).

java

Salin kode

```
System.out.println("JURUSAN MAHASISWA : " + mahasiswa1.getJurusan()); // Output: TEKNIK INFORMATIKA
```

**Fungsi:** Mencetak nilai atribut jurusan menggunakan metode getJurusan().

**Aksesibilitas:** Atribut jurusan bersifat private, sehingga tidak dapat diakses langsung. Metode getJurusan() (yang bersifat public) digunakan untuk mendapatkan nilai tersebut.

java

Salin kode

```
mahasiswa1.setJurusan("SISTEM INFORMASI");  
System.out.println("JURUSAN MAHASISWA SETELAH DIUBAH : " + mahasiswa1.getJurusan()); // Output: SISTEM INFORMASI
```

**Fungsi:**

- Mengubah nilai atribut jurusan dari "TEKNIK INFORMATIKA" menjadi "SISTEM INFORMASI" menggunakan metode setJurusan().
- Mencetak nilai jurusan yang baru untuk memastikan perubahan berhasil.

**Aksesibilitas:** Metode setJurusan() bersifat public, memungkinkan modifikasi atribut jurusan dari luar kelas.

java

Salin kode

```
mahasiswa1.tampilkanInfo();
```

**Fungsi:** Memanggil metode `tampilkanInfo()` yang mencetak semua informasi mahasiswa (nama, usia, dan jurusan) ke konsol.

## 2. Program Mobil

java

Salin kode

```
package Pertemuan7;

public class Mobil {
    // Atribut dan metode akan dijelaskan di bawah ini
}
```

**Package Declaration:** `package Pertemuan7;` menunjukkan bahwa kelas `Mobil` berada dalam package `Pertemuan7`.

**Class Declaration:** `public class Mobil` mendefinisikan kelas `Mobil` sebagai kelas publik yang dapat diakses dari luar package.

java

Salin kode

```
public String merk;
protected int tahunProduksi;
private double harga;
```

**public String merk:** Atribut `merk` dapat diakses dari mana saja karena bersifat `public`.

**protected int tahunProduksi:** Atribut `tahunProduksi` dapat diakses dalam package yang sama dan oleh subclass di package lain karena bersifat `protected`.

**private double harga:** Atribut `harga` bersifat `private`, hanya bisa diakses dari dalam kelas `Mobil`.

java

Salin kode

```
public Mobil(String merk, int tahunProduksi, double harga) {  
    this.merk = merk;  
    this.tahunProduksi = tahunProduksi;  
    this.harga = harga;  
}
```

**Fungsi:** Konstruktor ini menginisialisasi objek Mobil dengan merk, tahunProduksi, dan harga.

**Parameter:**

- String merk: Merek mobil.
- int tahunProduksi: Tahun produksi mobil.
- double harga: Harga mobil.

java

Salin kode

```
public double getHarga() {  
    return harga;  
}
```

**Fungsi:** Mengembalikan nilai harga.

**Aksesibilitas:** Bersifat public, sehingga dapat diakses dari luar kelas Mobil.


java

Salin kode

```
public void setHarga(double hargaBaru) {  
    if (hargaBaru > 0) {  
        this.harga = hargaBaru;  
    } else {  
        System.out.println("HARGA HARUS LEBIH BESAR DARI 0");  
    }  
}
```

- **Fungsi:** Mengubah nilai harga jika hargaBaru lebih besar dari 0. Jika tidak, pesan kesalahan ditampilkan.
- **Parameter:**
  - double hargaBaru: Harga baru yang ingin diset.
- **Aksesibilitas:** Bersifat public.

java


 Salin kode

```
public void tampilkanInfoMobil() {  
    System.out.println("MERK : " + merk);  
    System.out.println("TAHUN PRODUKSI : " + tahunProduksi);  
    System.out.println("HARGA : " + harga);  
}
```

**Fungsi:** Menampilkan semua informasi tentang mobil (merk, tahunProduksi, dan harga) ke konsol.

**Aksesibilitas:** Bersifat public.


java

 Salin kode

```
Mobil mobil1 = new Mobil("TOYOTA", 2022, 30000000);
```

Membuat instansi Mobil dengan merk "TOYOTA", tahunProduksi 2022, dan harga 30,000,000.


java

 Salin kode

```
System.out.println("MERK MOBIL : " + mobil1.merk);  
System.out.println("TAHUN PRODUKSI MOBIL : " + mobil1.tahunProduksi);  
System.out.println("HARGA MOBIL : " + mobil1.getHarga());
```

Mengakses merk dan tahunProduksi secara langsung karena bersifat public dan protected, dan mengakses harga melalui metode getHarga() karena harga bersifat private.


java

 Salin kode

```
mobil1.setHarga(35000000);  
System.out.println("HARGA MOBIL SETELAH DIUBAH : " + mobil1.getHarga());
```

Mengubah harga menjadi 35,000,000 menggunakan setHarga().

java

 Salin kode

```
mobil1.tampilkanInfoMobil();
```

Memanggil metode tampilkanInfoMobil() untuk menampilkan semua informasi mobil.

### 3. Program Nilai 1

```
java Salin kode

static class Nilai {
    // Atribut
    private double quis;
    private double uts;
    private double uas;
}
```

Kelas Nilai adalah inner class yang bersifat static, dengan atribut quis, uts, dan uas, masing-masing bertipe double dan bersifat private. Ini berarti atribut-atribut ini hanya bisa diakses melalui metode getter dan setter yang disediakan.

```
Salin kode

public void setQuis(double quis) {
    this.quis = quis;
}

public void setUTS(double uts) {
    this.uts = uts;
}

public void setUAS(double uas) {
    this.uas = uas;
}

// Getter untuk atribut
public double getQuis() {
    return quis;
}

public double getUTS() {
    return uts;
}

public double getUAS() {
    return uas;
}
```

**Setter:** Mengatur nilai untuk atribut quis, uts, dan uas.

**Getter:** Mengambil nilai dari masing-masing atribut.



java

Salin kode

```
public double getNA() {  
    return (0.2 * quis) + (0.3 * uts) + (0.5 * uas);  
}
```

**Fungsi:** Menghitung nilai akhir (NA) berdasarkan bobot:

- quis: 20%
- uts: 30%
- uas: 50%

**Return Value:** Mengembalikan hasil perhitungan NA sebagai double.

java

Salin kode


```
public char getIndex() {  
    double na = getNA();  
    if (na >= 80 && na <= 100) {  
        return 'A';  
    } else if (na >= 68 && na < 80) {  
        return 'B';  
    } else if (na >= 56 && na < 68) {  
        return 'C';  
    } else if (na >= 45 && na < 56) {  
        return 'D';  
    } else {  
        return 'E';  
    }  
}
```

**Fungsi:** Menentukan indeks nilai berdasarkan nilai akhir (NA).

- A: 80 - 100
- B: 68 - 79
- C: 56 - 67
- D: 45 - 55
- E: Di bawah 45

**Return Value:** Mengembalikan karakter yang merepresentasikan indeks nilai.

java

 Salin kode


```
public String getKeterangan() {  
    switch (getIndex()) {  
        case 'A':  
            return "SANGAT BAIK";  
        case 'B':  
            return "BAIK";  
        case 'C':  
            return "CUKUP";  
        case 'D':  
            return "KURANG";  
        case 'E':  
            return "SANGAT KURANG";  
        default:  
            return "TIDAK ADA";  
    }  
}
```

**Fungsi:** Memberikan keterangan berdasarkan indeks nilai:

- A -> "SANGAT BAIK"
- B -> "BAIK"
- C -> "CUKUP"
- D -> "KURANG"
- E -> "SANGAT KURANG"

**Return Value:** Mengembalikan string keterangan yang sesuai.

java

 Salin kode

```
public static void main(String[] args) {

    System.out.print("\033[H\033[2J"); // Membersihkan layar terminal

    Nilai n = new Nilai();           // Membuat objek Nilai
    n.setQuis(60);                    // Mengatur nilai quis
    n.setUTS(80);                     // Mengatur nilai UTS
    n.setUAS(75);                     // Mengatur nilai UAS


    System.out.println("QUIS        : " + n.getQuis());
    System.out.println("UTS         : " + n.getUTS());
    System.out.println("UAS         : " + n.getUAS());
    System.out.println("NILAI AKHIR : " + n.getNA());
    System.out.println("INDEX       : " + n.getIndex());
    System.out.println("KETERANGAN  : " + n.getKeterangan());
}
```

**Fungsi:** Metode main adalah titik masuk untuk menjalankan program. Kode ini:

1. Menghapus layar terminal (hanya berfungsi di terminal tertentu).
2. Membuat objek Nilai.
3. Menetapkan nilai quis, uts, dan uas menggunakan metode setter.
4. Menampilkan nilai quis, uts, uas, nilai akhir, indeks, dan keterangan ke layar.

## 4. Program Nilai 2

java

 Salin kode

```
private double quiz;
private double UTS;
private double UAS;
```

Atribut quiz, UTS, dan UAS disimpan sebagai variabel private bertipe double, yang berarti nilai-nilai ini hanya bisa diakses melalui metode getter dan setter di dalam kelas ini.

```

public void setQuiz(double quiz) {
    if (quiz >= 0 && quiz <= 100) {
        this.quiz = quiz;
    } else {
        System.out.println("NILAI QUIZ HARUS ANTARA 0 DAN 100");
    }
}

public void setUTS(double UTS) {
    if (UTS >= 0 && UTS <= 100) {
        this.UTS = UTS;
    } else {
        System.out.println("NILAI UTS HARUS ANTARA 0 DAN 100");
    }
}

public void setUAS(double UAS) {
    if (UAS >= 0 && UAS <= 100) {
        this.UAS = UAS;
    } else {
        System.out.println("NILAI UAS HARUS ANTARA 0 DAN 100");
    }
}

```

**Fungsi:** Metode ini digunakan untuk menetapkan nilai quiz, UTS, dan UAS dengan validasi.

**Validasi:** Jika nilai yang diberikan berada di luar rentang 0 - 100, maka pesan error akan ditampilkan, dan nilai tidak akan disimpan.

java

```

public double getQuiz() {
    return quiz;
}

public double getUTS() {
    return UTS;
}

public double getUAS() {
    return UAS;
}

```

**Fungsi:** Metode ini digunakan untuk mengakses nilai quiz, UTS, dan UAS dari luar kelas.

**Return Value:** Mengembalikan nilai dari masing-masing atribut.

java

Salin kode

```
public double getNA() {  
    return 0.20 * quiz + 0.30 * UTS + 0.50 * UAS;  
}
```

- **Fungsi:** Menghitung nilai akhir berdasarkan bobot nilai:
  - quiz: 20%
  - UTS: 30%
  - UAS: 50%
- **Return Value:** Mengembalikan hasil perhitungan nilai akhir (NA) sebagai double.

java

Salin kode

```
public class NilaiTester2 {  
    public static void main(String[] args) {  
  
        System.out.print("\033[H\033[2J"); // Membersihkan layar terminal  
  
        Nilai n = new Nilai(); // Membuat objek Nilai  
  
        n.setQuiz(90);           // Menetapkan nilai quiz menjadi 90  
        n.setUTS(70);           // Menetapkan nilai UTS menjadi 70  
        n.setUAS(150);          // Menetapkan nilai UAS menjadi 150 (akan ditolak karena me  
  
        System.out.println("NILAI QUIZ : " + n.getQuiz());  
        System.out.println("NILAI UTS  : " + n.getUTS());  
        System.out.println("NILAI UAS  : " + n.getUAS());  
        System.out.println("NILAI AKHIR : " + n.getNA()); // Menampilkan Nilai Akhir (NA)  
    }  
}
```

**Membuat Objek Nilai:** Nilai n = new Nilai(); membuat objek Nilai bernama n.

**Set Nilai dengan Validasi:**

- n.setQuiz(90); menetapkan nilai kuis menjadi 90 (dalam rentang yang valid).
- n.setUTS(70); menetapkan nilai UTS menjadi 70 (dalam rentang yang valid).

- `n.setUAS(150);` mencoba menetapkan nilai UAS menjadi 150. Karena nilai ini melebihi 100, setter akan menampilkan pesan "NILAI UAS HARUS ANTARA 0 DAN 100" dan nilai tidak akan disimpan.

**Menampilkan Nilai dan Nilai Akhir:** Menampilkan nilai quiz, UTS, UAS, dan nilai akhir yang dihitung berdasarkan bobot.

## 5. Program Siswa

```
java Salin kode  
  
class Siswa {  
    private String nama;           // Private: hanya bisa diakses dalam kelas ini  
    private int nilaiUjian;        // Private: hanya bisa diakses dalam kelas ini
```

`nama` dan `nilaiUjian` adalah atribut private dari kelas `Siswa`, yang berarti mereka tidak dapat diakses langsung dari luar kelas ini.

`nama` menyimpan nama siswa, sedangkan `nilaiUjian` menyimpan nilai ujian siswa.


```
java Salin kode  
  
public Siswa(String nama, int nilaiUjian) {  
    this.nama = nama;  
    setNilaiUjian(nilaiUjian); // Gunakan setter untuk validasi nilai awal  
}
```

Constructor ini digunakan untuk membuat objek `Siswa` dengan `nama` dan `nilaiUjian` tertentu.

`this.nama = nama;` menginisialisasi nama siswa.

`setNilaiUjian(nilaiUjian);` menggunakan setter untuk mengatur nilai ujian dan memvalidasi nilai awal yang diberikan.

java

 Salin kode

```
public String getNama() {  
    return nama;  
}  
  
public void setNama(String nama) {  
    this.nama = nama;  
}  
  
public int getNilaiUjian() {  
    return nilaiUjian;  
}  
  
public void setNilaiUjian(int nilaiUjian) {  
    if (nilaiUjian >= 0 && nilaiUjian <= 100) {  
        this.nilaiUjian = nilaiUjian;  
    } else {  
        System.out.println("\nNILAI HARUS ANTARA 0 DAN 100");  
    }  
}
```


getNama() mengembalikan nilai nama.

setNama(String nama) menetapkan nilai baru untuk nama.

getNilaiUjian() mengembalikan nilai nilaiUjian.

setNilaiUjian(int nilaiUjian) mengatur nilai nilaiUjian dengan validasi. Jika nilai berada di luar rentang 0-100, pesan kesalahan akan ditampilkan dan nilai tidak diubah.


java

 Salin kode

```
public void tampilkanInfo() {  
    System.out.println("NAMA SISWA : " + nama);  
    System.out.println("NILAI UJIAN : " + nilaiUjian);  
}
```

Metode ini menampilkan nama dan nilaiUjian dari siswa.

java


 Salin kode

```
public class SiswaTester {  
    public static void main(String[] args) {  
  
        System.out.print("\033[H\033[2J");
```

SiswaTester adalah kelas utama untuk menjalankan dan menguji fungsi kelas Siswa.

System.out.print("\033[H\033[2J"); membersihkan layar konsol (ini hanya berfungsi di beberapa terminal).

java


 Salin kode

```
Siswa siswa1 = new Siswa("ANDI", 85);  
siswa1.tampilkanInfo();
```

siswa1 adalah objek dari kelas Siswa yang diinisialisasi dengan nama "ANDI" dan nilaiUjian 85.

tampilkanInfo() menampilkan informasi awal siswa1.

java

 Salin kode

```
siswa1.setNama("BUDI");  
siswa1.setNilaiUjian(95);  
  
System.out.println("\nSETELAH DIUBAH : ");  
siswa1.tampilkanInfo();
```


setNama("BUDI") mengubah nama menjadi "BUDI".

setNilaiUjian(95) mengubah nilaiUjian menjadi 95.

Setelah atribut diubah, tampilkanInfo() dipanggil untuk menampilkan informasi terbaru.

## 6. Program Waktu

java

 Salin kode

```
public class Waktu {  
    private int menitWaktu;
```

menitWaktu adalah atribut private yang menyimpan jumlah waktu dalam satuan menit.



java

Salin kode

```
public Waktu(int menitWaktu) {  
    this.menitWaktu = menitWaktu;  
}
```

- Konstruktor ini menerima parameter menitWaktu dan menentukannya ke atribut kelas menitWaktu.
- Misalnya, jika objek dibuat dengan `new Waktu(125)`, menitWaktu akan diinisialisasi dengan nilai 125 menit.

java

Salin kode

```
public int getJam() {  
    return menitWaktu / 60;  
}
```

`getJam()` mengembalikan jumlah jam dengan membagi menitWaktu dengan 60.

Misalnya, jika menitWaktu adalah 125, hasilnya adalah 2 (jam).

java

Salin kode

```
public int getMenit() {  
    return menitWaktu % 60;  
}
```

`getMenit()` mengembalikan sisa menit setelah dihitung jam.

Misalnya, jika menitWaktu adalah 125, hasilnya adalah 5 (menit).

java


Salin kode

```
public void setJam(int j) {  
    menitWaktu = (j * 60) + (menitWaktu % 60);  
}
```

`setJam(int j)` mengatur bagian jam dengan mengalikan `j` dengan 60, kemudian menambahkan sisa menit dari menitWaktu.

Jika kita memanggil `setJam(3)` ketika menitWaktu adalah 125, hasilnya adalah  $(3 * 60) + 5 = 185$ .

java


 Salin kode

```
public void setMenit(int m) {  
    menitWaktu = ((menitWaktu / 60) * 60) + m;  
}
```

setMenit(int m) mengatur bagian menit dengan mengganti bagian menit dari menitWaktu tanpa mempengaruhi bagian jam.

Jika kita memanggil setMenit(30) ketika menitWaktu adalah 185, hasilnya adalah  $180 + 30 = 210$ .


java

 Salin kode

```
public int getTotalMenit() {  
    return menitWaktu;  
}
```

getTotalMenit() mengembalikan nilai menitWaktu secara keseluruhan.


java

 Salin kode

```
public static void main(String[] args) {  
  
    System.out.print("\033[H\033[2J");  
  
    Waktu waktu = new Waktu(125); // 125 menit dari jam 00:00
```

Kode ini membersihkan layar konsol dan membuat objek Waktu dengan nilai 125 menit, atau 2 jam dan 5 menit dari jam 00:00.


java

 Salin kode

```
System.out.println("JAM : " + waktu.getJam()); // Output: 2  
System.out.println("MENIT : " + waktu.getMenit()); // Output: 5
```

Menampilkan jam (2) dan menit (5) dari menitWaktu (125 menit).

java


 Salin kode

```
waktu.setJam(3); // Mengatur jam menjadi 3, menit tetap 5
System.out.println("SETELAH SET JAM(3) - TOTAL MENIT : " + waktu.getTotalMenit());
```

setJam(3) mengubah menitWaktu menjadi 185 (3 jam 5 menit).

getTotalMenit() kemudian menampilkan nilai 185.

java

 Salin kode

```
waktu.setMenit(30); // Mengatur menit menjadi 30, jam tetap 3
System.out.println("SETELAH SET MENIT(30) - TOTAL MENIT : " + waktu.getTotalMenit(
    }
}
```

setMenit(30) mengubah menitWaktu menjadi 210 (3 jam 30 menit).

getTotalMenit() kemudian menampilkan nilai 210.