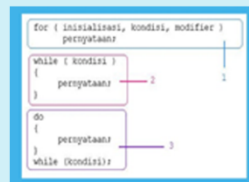




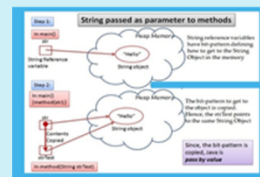
Struktur Dasar



Pengkondisian



Perulangan & Array



Method



Implementasi

# MODUL PRAKTIKUM

## Algoritma Pemrograman

### Tim Penyusun :

Basworo Bawiso M (BBM)	1106120053	Agung Candra Dutiya Purwanta (ANG)	1106124202
Pratiwi Galuh Putri (PGP)	1106124197	Mochamad Thariq Januar (JNR)	1106120083
Nurrida Aini Zuhroh (RID)	1106124204	Slamet Mamat Rachmat (SLR)	1106120077
Anisatun Nafi'ah (ANS)	1106124198	Reza Harli Saputra (RHS)	1106122133
I Komang Jaka Aksara W (JEK)	1106120068	Kevin Rohni Goklas Sinaga (KVN)	1106120110
Jan Fandro Siahaan (JFS)	1106120078	Timbul Prawira Gultom (TPG)	1106121188
Nana Ramadhewi (RDN)	1106120064	Adnan Baharrudin Fanani (ABF)	1106122109
Cahya Nofandiyan Putra (CNP)	1106122108	Nur Intan Paramanisa (INT)	1106122050

### Dosen Pembina :

Mardiyanto Wiyogo, ST



LABORATORIUM PROGRAMMING DAN DATABASE  
PROGRAM STUDI SISTEM INFORMASI  
TELKOM ENGINEERING SCHOOL  
TELKOM UNIVERSITY

# Tata Tertib, Kelengkapan dan Persyaratan Praktikum Algoritma dan Pemrograman

## I. Kelengkapan dan persyaratan praktikum

### 1. Kelengkapan Praktikum

#### Kartu Praktikum

- Kartu praktikum harus segera dilengkapi dengan data – data praktikan berikut foto dan cap laboratorium. Pemberian cap dilakukan pada saat praktikum pertama.
- Praktikan harus membawa kartu praktikum setiap mengikuti kegiatan praktikum.
- Apabila kartu praktikum hilang, praktikan harus mengganti sesuai dengan aslinya kemudian meminta cap laboratorium kepada asisten untuk legalisir , sebelum praktikum selanjutnya diadakan.

#### Pre Test / TP

- Pre test diadakan sebelum pelaksanaan praktikum.
- Pre test merupakan **tugas wajib**.
- Pengumuman mengenai Pre test akan diumumkan kemudian.
- Pengerjaan modul dikerjakan pada kertas A4, margin 3222, tulis tangan rapi, serta nomor berurut.
- Pre test wajib dikerjakan oleh praktikan, apabila tidak mengerjakan pre test maka nilai modul akan dikurangi.
- Apabila melanggar aturan pengerjaan, akan dikenakan sanksi pengurangan nilai modul.
- **Plagiat = nilai 0**

#### Tes Awal

- Tes awal dilaksanakan sebelum pelaksanaan praktikum.
- Pengumuman mengenai tes awal akan diumumkan kemudian.

- Setiap keterlambatan, praktikan diberi kesempatan untuk mengikuti tes awal tanpa diberi perpanjangan waktu.

### **Praktikum**

- Setiap praktikan wajib mengikuti praktikum dengan persyaratan praktikum yang telah ditentukan. Apabila salah satu atau lebih dari syarat tersebut tidak terpenuhi maka praktikan tidak diperkenankan mengikuti praktikum dan asisten berhak mengeluarkan praktikan.
- Praktikan wajib mematuhi tata tertib yang ada pada Common Lab.

### **Tes Akhir**

- Tes akhir merupakan tes yang dilakukan pada akhir praktikum.
- Pengumuman mengenai tes akhir akan diumumkan kemudian.

### **2.Persyaratan Praktikum**

- Memenuhi seluruh kelengkapan praktikum yang tercantum pada poin sebelumnya.

## **II. Tata Tertib Praktikum**

1. Praktikan wajib memenuhi seluruh kelengkapan dan persyaratan Praktikum termasuk membawa kartu Praktikum.
2. Asisten dapat memberi teguran bahkan mengeluarkan praktikan yang tidak dapat menjaga ketenangan, ketertiban, kebersihan, dan kerapian laboratorium pada saat praktikum dilaksanakan.
3. Setiap praktikan wajib bertutur kata baik dan sopan dalam bersikap kepada asisten laboratorium.
4. Setiap barang yang dipinjam wajib dikembalikan ke tempat semula.
5. Tidak mengikuti praktikum **dua modul** atau lebih tanpa alasan yang jelas (sakit dengan keterangan dokter, berita duka, acara keagamaan, dan acara kampus disertai surat keterangan resmi minimal fakultas) dan tidak dapat dipertanggungjawabkan maka nilai praktikum algoritma pemrograman sama dengan E.
6. Tukar jadwal :
  - a. Tukar jadwal dilakukan oleh masing-masing praktikan melalui form tukar jadwal.
  - b. Tukar jadwal dapat disetujui oleh asisten Lab apabila alasannya jelas.

- c. Penyerahan form tukar jadwal, maksimal H - 1x24 jam dari jadwal praktikum terdekat dari modul yang bersangkutan.
7. Pengumpulan Tugas yang berkaitan dengan praktikum hanya dilakukan di laboratorium Prodase (C200).
8. Seragam :
- Untuk setiap kegiatan praktikum, praktikan diwajibkan untuk memakai seragam rapi dan sopan berupa kemeja putih dan memakai celana bahan warna biru gelap atau rok bahan panjang semata kaki warna biru gelap (seragam Telkom University Fak.Teknik).
9. Sanksi Keterlambatan praktikum :
- ❖ Terlambat 1 – 15 menit : boleh mengikuti praktikum tanpa adanya penambahan waktu.
  - ❖ Terlambat > 15 menit : tidak diperbolehkan mengikuti praktikum.
  - ❖ Referensi waktu berdasarkan jam digital WITU
10. Selama praktikum, praktikan tidak diperkenankan meninggalkan ruangan praktikum tanpa seizin asisten jaga.
11. Alat komunikasi dinyalakan dalam mode silent atau dimatikan.
12. Pengumuman mengenai Praktikum Algoritma Pemrograman HANYA akan diinformasikan melalui media yang ditentukan.
13. Segala bentuk kecurangan akan dikenai sanksi nilai “E”.
14. Kepentingan praktikum akan dilayani pada jam kerja 08.00 – 20.00 WITU
15. Praktikum susulan:
- Merupakan praktikum yang diperuntukkan bagi praktikan yang tidak mengikuti praktikum pada modul tertentu dengan alasan yang jelas dan dapat dipertanggungjawabkan.
- Syarat dan ketentuan praktikum susulan:
1. Praktikan tidak mengikuti maksimal dua (2) praktikum dengan alasan yang dapat dipertanggungjawabkan.
  2. Dilakukan pada akhir semua praktikum (setelah modul VI).
  3. Nilai yang diberikan pada praktikum susulan maksimal 90% dari nilai pada praktikum normal.
16. Hal – hal yang belum tercantum dalam peraturan ini akan ditentukan kemudian.

### III. Tabel Waktu Praktikum dan Penilaian

Segi penilaian	Nilai	Waktu
Pre test / TP	10%	----
Tes awal	20%	10 menit
Praktikum	40%	60 menit
Tes akhir	30%	20 menit
Tugas Bonus	Opsional	Opsional

### IV. Bentuk Penilaian

Setiap modul masing-masing memiliki point / nilai maksimal yang berbeda. Berikut tingkatan point/nilai yang terdapat dalam masing-masing modul :

Modul	Point / nilai maksimal
Pengenalan java	80
Pengkondisian	85
Perulangan	90
Array	100
Method	115
File	130

Dengan demikian, point maksimum praktikum Algoritma dan Pemrograman adalah 600 point dari keseluruhan modul awal hingga akhir.

## Pengantar JAVA

### TUJUAN PRAKTIKUM :

Praktikan dapat mengenal bentuk dasar code Java Console Application dengan sederhana, mengenal operasi dan tipe dasar.

### JAVA SEKILAS PANDANG

Java merupakan bahasa pemrograman tingkat tinggi yang diciptakan berdasarkan turunan dari C++. Target utama dari penggunaan bahasa Java adalah pengkodean berarah objek yang simpel (tidak memerlukan header), menghindari manipulasi pointer secara manual (otomatis), dan lainnya. Kini, penggunaan Java sudah sangat banyak di perusahaan mengingat Java adalah *cross-platform* dan bahkan *cross-device*.

Bentuk dasar (Anatomi) Console Application :

```
public class Main {  
  
    /**  
     * @param args  
     */  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println("Give It a Try!");  
    }  
}
```

Penjelasan :

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

Awal mula kode bermula dari sini. Fungsi main() ini menandakan bahwa baris kode yang dibawah ini akan dijalankan pertama kali secara otomatis ketika program dijalankan.

```
System.out.println("Give It a Try!");
```

Salah satu contoh baris kode yang mana ini digunakan untuk menampilkan ke console / layar. Perlu diperhatikan bahwa di bagian akhir perlu diberikan semicolon (;) untuk menandakan akhir dari satu baris kode. Serta, penulisan kode di Java membedakan huruf besar dan kecil (case-sensitive) sehingga perlu berhati-hati akan huruf besar dan kecil ketika menulis baris kode.

```
public static void main(String[] args) {  
    System.out.println("Give It a Try!");  
}
```

Sebenarnya main() di atas adalah sebuah fungsi (penggunaannya akan dijelaskan nanti di bab selanjutnya) yang mana perlu diawali dan diakhiri oleh kurung kurawal ( { ... } )

#### A. KOMENTAR

Komentar dalam Java berupa sebuah blok untuk menerangkan/memberikan informasi pada suatu kelas, field, method, dan lainnya agar orang lain dapat mengerti apa maksud dari kode tersebut.

Bentuk umum komentar dalam Java ada dua :

```
/* ... */
```

```
// ...
```

```
public class Komentar {  
    public static void main(String[] args) {  
        /**  
         * Ini adalah contoh komentar.  
         * Komentar tidak akan dieksekusi pada program  
         */  
        System.out.print("Penggunaan komentar");  
    }  
}
```

Komentar **TIDAK AKAN** ikut dikompilasi menjadi program sehingga programmer tidak perlu khawatir programnya akan membengkak karena terlalu banyak menuliskan penjelasan (komentar).



## B. INPUT / OUTPUT

Dalam java dikenal juga dengan proses input dan proses output. Berikut adalah contoh program untuk input dan output :

```
/* Program untuk proses input dan output*/
import java.util.Scanner;
public class Praktikum {
    public static void main(String[] args) {
        Scanner input= new Scanner(System.in);
        System.out.println("Nilai a adalah");
        //input
        int a=input.nextInt();
        //output
        System.out.println(a);
    }
}
```

Penjelasan :

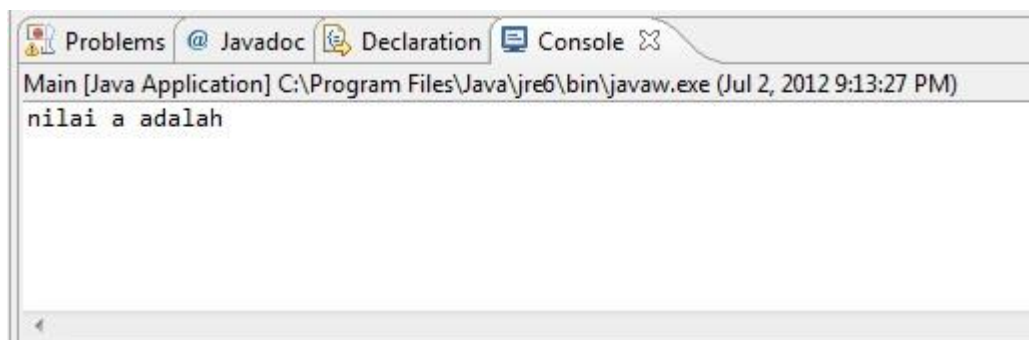
Baris kode berikut adalah baris kode yang digunakan untuk proses input atau proses pemberian nilai pada sebuah variabel (akan dijelaskan pada pembahasan berikutnya).

```
Scanner input= new Scanner(System.in);
```

Perlu diperhatikan bahwa, ketika Scanner digunakan. Pada class yang sama juga harus terdapat baris kode

```
import java.util.Scanner;
```

Baris tersebut berfungsi agar Scanner dapat didefinisikan pada class tersebut. Maka ketika program dijalankan, program akan menampilkan sebagai berikut



Ketika kursor berada pada console/layar, maka kita sudah dapat mengetikkan suatu nilai, dan nilai tersebut akan menjadi nilai dari variabel a.

### C. TYPE DATA

Tipe data adalah klasifikasi antar data. Tujuannya adalah mencegah tercampurnya data lain yang memiliki “bentuk” yang berbeda. Tipe data tersebut dapat disimpan dalam sebuah wadah yang disebut **variable**. Contoh pembuatan variabel :

```
public class Praktikum {  
  
    public static void main(String[] args) {  
        String h= "Hello!";  
        System.out.println(h);  
    }  
}
```

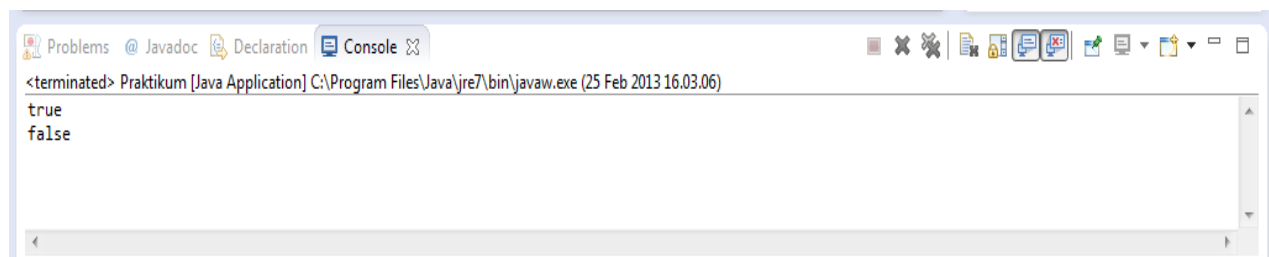
Berikut adalah tipe data yang dikenal di Java :

#### 1. Boolean

Boolean adalah tipe data yang hanya menyatakan kondisi **true** (benar) dan **false** (salah). Boolean pada dasarnya adalah representasi dari 1 (true) dan 0 (false).

Contoh :

```
public class Praktikum {  
  
    public static void main(String[] args) {  
        boolean b=true;  
        System.out.println(b);  
        boolean c=false;  
        System.out.println(c);  
    }  
    // Maka di layar akan menuliskan true false  
}
```

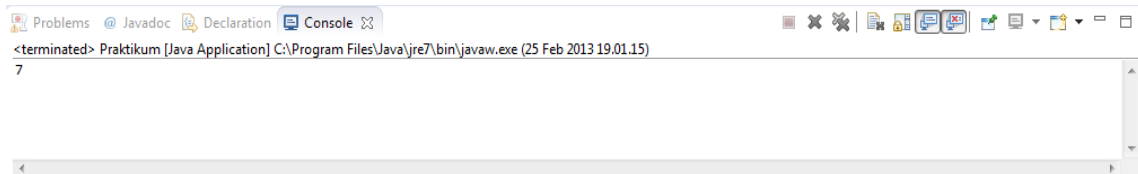


## 2. Integer

Integer merupakan tipe data numerik yang bulat dan dapat dilakukan proses aritmatika. Adapun tipe data yang sejenis adalah Byte, Long, Short. Perbedaannya adalah besaran bit yang dipakai.

Contoh :

```
public class Praktikum {  
    //Melakukan operasi penjumlahan  
    public static void main(String[] args) {  
        int a=3+4;  
        System.out.println(a);  
    }  
    // Maka di layar akan menuliskan angka 7  
}
```

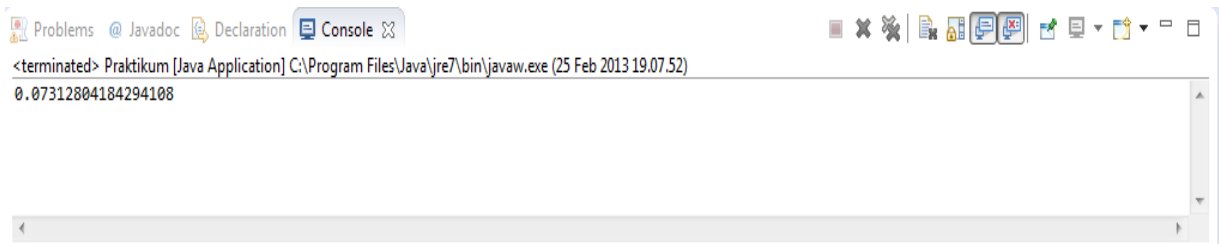


## 3. Float

Type float digunakan untuk menangani bilangan pecahan. Tipe data jenis adalah Double untuk angka yang lebih besar dan presisi lebih tinggi.

Contoh :

```
public class Praktikum {  
    //Math.random() adalah fungsi untuk mengeluarkan nilai acak antara 0  
    hingga 1  
    public static void main(String[] args) {  
        double d= Math.random();  
        System.out.println(d);  
    }  
    // Maka di layar akan menuliskan angka random antara 0-1  
}
```



#### 4. Karakter

Char adalah tipe data untuk karakter. Pada dasarnya karakter memiliki nomor indeks yang biasa disebut ASCII code yang direpresentasikan sebagai angka. Sebagai contoh huruf 'd' memiliki nilai 100. Char dapat dilakukan operasi matematika seperti layaknya integer.

Contoh :

```
public class Praktikum {
    //Math.random() adalah fungsi untuk mengeluarkan nilai acak antara
    0 hingga 1
    public static void main(String[] args) {
        char b='d'+1;
        System.out.println(b);
    }
    // Maka di layar akan menuliskan huruf e. BUKAN d+1 atau 5
}
```

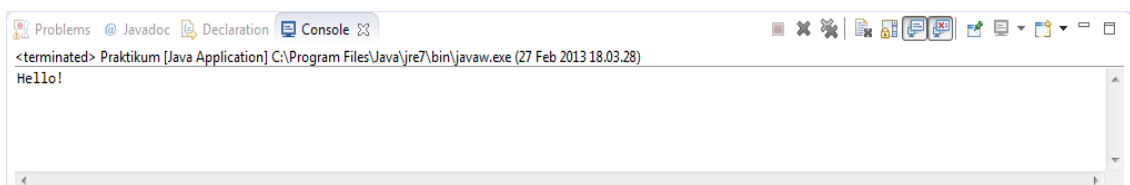


#### 5. String

String adalah tipe data yang dapat menyimpan sederet karakter menjadi satu seperti layaknya kalimat atau kata. Pada dasarnya string adalah sejenis array (sekumpulan) dari char yang dimanipulasi sehingga menjadi tipe data baru. Dibandingkan tipe data yang lain, tipe data string memiliki fungsi manipulasi paling banyak.

Contoh :

```
public class Praktikum {
    //Tes membuat variable
    public static void main(String[] args) {
        String h= "Hello!";
        System.out.println(h);
    }
    // Maka di layar akan menuliskan tulisan Hello!
}
```



## D. OPERATOR PADA JAVA

Operator merupakan tanda yang digunakan untuk melakukan suatu operasi.

Ada beberapa jenis operator yang dapat digunakan, yaitu :

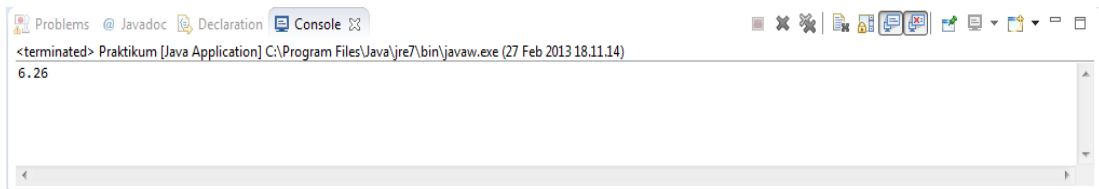
- Operator assignment
- Operator aritmatik
- Operator perbandingan
- Operator logika

### 1. OPERATOR ASSIGNMENT

Operator assignment adalah operator yang melakukan pengisian nilai kepada suatu variabel sehingga variabel yang telah dibuat jadi menyimpan suatu nilai.

Contoh :

```
public class Praktikum {  
    //Melakukan operasi +1 pada karakter  
    public static void main(String[] args) {  
        double d;  
        d=3.14;  
        d=d+3.12; // Menambah 3.14 dengan 3.12 lalu dimasukan  
        sebagai nilai d  
        System.out.println(d);  
    }  
    // Maka di layar akan menuliskan angka 6.26  
}
```



### 2. OPERATOR ARITMETIK ()

Ada beberapa operator aritmetik yang sudah kita kenal, yaitu

- Penjumlahan (+)
- Pengurangan (-)
- Pembagian (/)
- Perkalian (\*)
- Modulus (sisa pembagian %)

Berikut ini adalah contoh penggunaan operator aritmetik.

```

public class aritmatik {

    public static void main(String[] args) {
        System.out.println("penambahan 3+5 = "+(3+5));
        System.out.println("pengurangan 5-3= "+(5-3));
        System.out.println("perkalian 5*5= "+(5*5));
        System.out.println("pembagian 90/5= "+(90/5));
        System.out.println("modulus 52%6= "+(52%6));
    }

}

```

```

<terminated> aritmatik [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (27 Feb 2013 18.19.21)
penjumlahan 3+5 = 8
pengurangan 5-3= 2
perkalian 5*5= 25
pembagian 90/5= 18
modulus 52%6= 4

```

Untuk beberapa kasus seperti tipe data lain, operator bisa jadi bermakna lain. Seperti pada String, dapat dilakukan operator + untuk menggabungkan string pertama dan kedua dan selanjutnya, namun tidak dapat dioperasikan -, /, \* atau %.

Contoh :

```

public class Praktikum {
    //Menulis String
    public static void main(String[] args) {
        String h;
        h="Wow"+ "Wohoo";
        System.out.println(h);//
        //Maka dilayar akan menuliskan tulisan WowWohoo
    }

}

```

```

<terminated> Praktikum [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (27 Feb 2013 18.25.50)
WowWohoo

```

### 3. OPERATOR LOGIKA

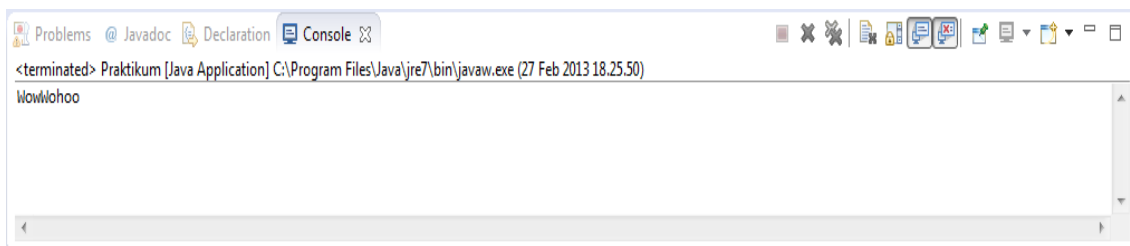
Operator logika digunakan untuk melakukan operasi dan komparasi dalam nilai Boolean. Ada beberapa operator yang digunakan untuk operasi boolean, yaitu,

- **Operator == (EQUAL)**

Operator == digunakan untuk menyatakan apakah nilai di ruas kiri sama dengan ruas kanan. Mengembalikan nilai true apabila ya dan false apabila tidak.

Contoh :

```
public class Praktikum {  
    //Operator EQUAL  
    public static void main(String[] args) {  
        boolean h=false;  
        System.out.println(h==false);  
        //Maka dilayar akan menuliskan tulisan true  
    }  
}
```

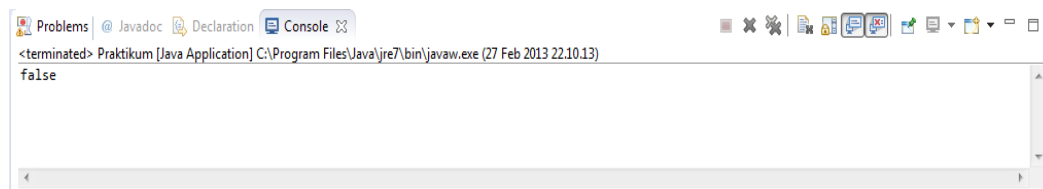


- **Operator && (AND)**

Operator && (AND) sifatnya adalah konjungsi (dan), dimana mengembalikan nilai true apabila ruas kiri dan kanan sama-sama memiliki nilai true, selain itu akan dianggap salah.

Contoh :

```
public class Praktikum {  
    //Operator AND  
    public static void main(String[] args) {  
        boolean h=false==false;  
        boolean b=true==false;  
        System.out.println(h && b);  
        //Maka dilayar akan menuliskan tulisan false  
    }  
}
```

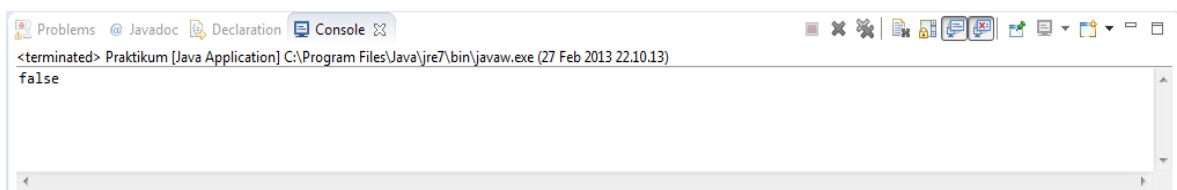


- **Operator || (OR)**

Operator || (OR) sifatnya adalah disjungsi (atau), dimana mengembalikan nilai true apabila antara ruas kiri atau kanan memiliki nilai true, salah satu atau keduanya. Mengembalikan nilai false apabila keduanya memiliki nilai false.

Contoh :

```
public class Praktikum {  
    //Operator OR  
    public static void main(String[] args) {  
        boolean h=false==false;  
        boolean b=true==true;  
        System.out.println(h || b);  
        //Maka dilayar akan menuliskan tulisan true  
    }  
}
```





- **Operator! (NOT)**

Operator ! (NOT) akan membalikkan boolean yang dijadikan operan. Apabila nilai operan adalah true, maka hasilnya akan menjadi false.

Contoh :

```
public class Praktikum {  
    public static void main(String[] args) {  
        boolean h=true==true;  
        System.out.println(!h); //  
        //Maka dilayar akan menuliskan tulisan false  
    }  
}
```



#### 4. **OPERATOR LOGIKA (NUMERIC)**

Untuk angka, dapat juga dilakukan hal-hal yang seperti kita lakukan dulu di logika matematika SMA. Yaitu (==) untuk sama dengan, (<) untuk kurang dari, (>) untuk lebih dari, (<=) untuk kurang dari atau sama dengan, (>=) untuk lebih dari atau sama dengan.

Contoh :

```
public class Praktikum {  
    public static void main(String[] args) {  
        int g=5;  
        System.out.println(3<=g);  
        //Maka dilayar akan menuliskan tulisan true  
    }  
}
```



## E. TOOLS YANG DIGUNAKAN

### 1. ECLIPSE

Dalam praktikum ini, tools yang digunakan adalah ECLIPSE. Eclipse merupakan sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan disemua platform.

### 2. PENGUNAAN ECLIPSE

Dalam menggunakan eclipse, ada beberapa langkah yang harus dilakukan. Berikut adalah langkah – langkah yang harus dilakukan, yaitu

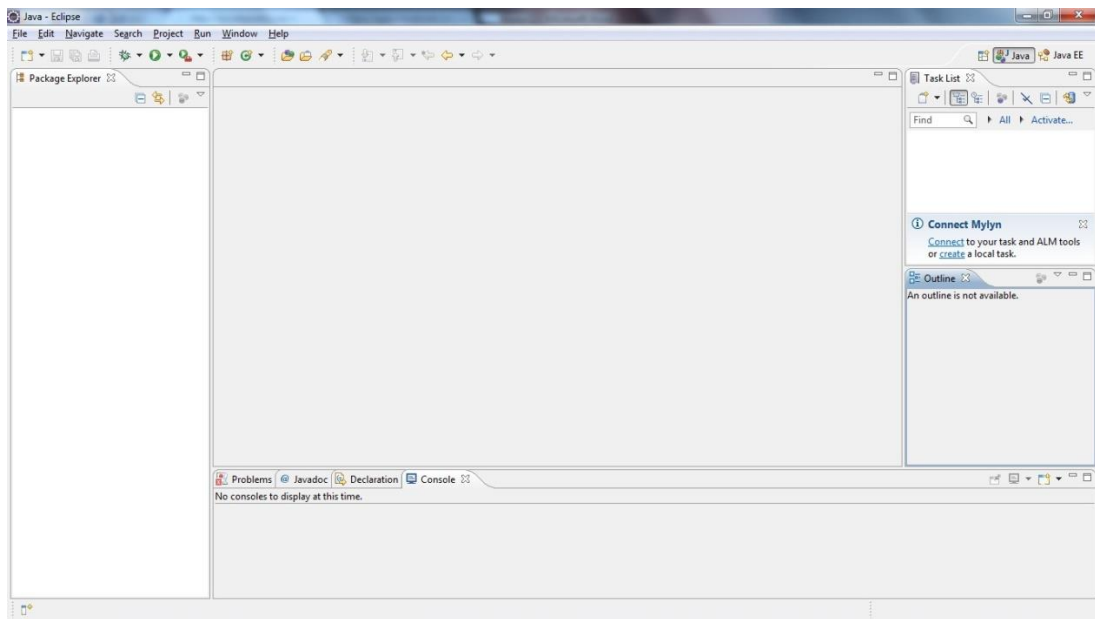
- a. Membuka eclipse, klik icon seperti gambar dibawah ini untuk membuka eclipse



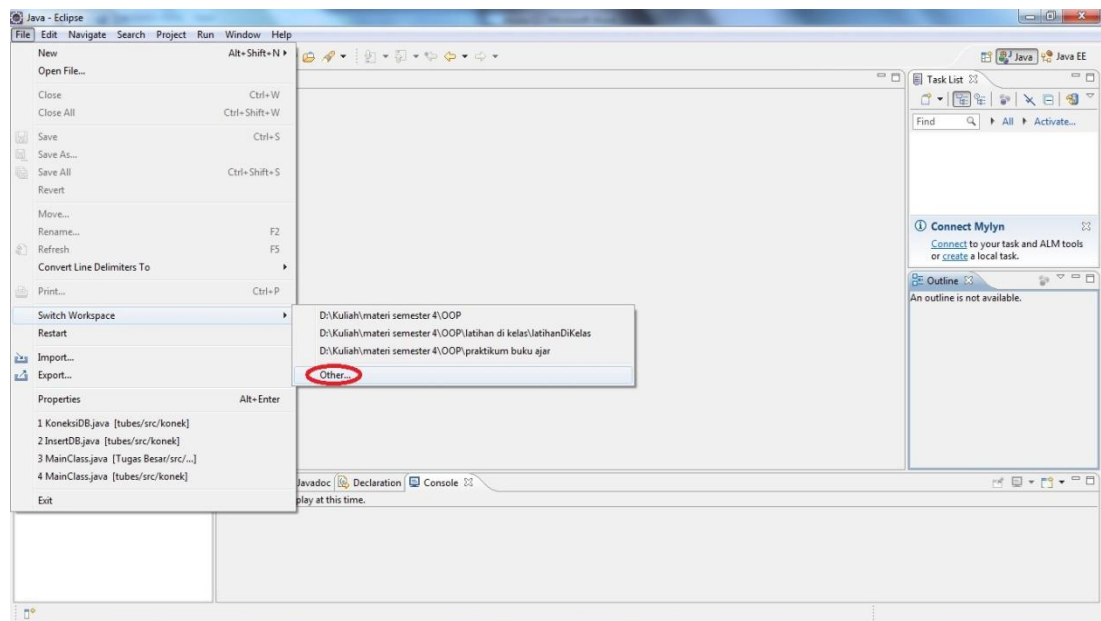
- b. Setelah klik gambar seperti itu, eclipse akan membuka workspace yang ditandai dengan proses seperti gambar dibawah ini



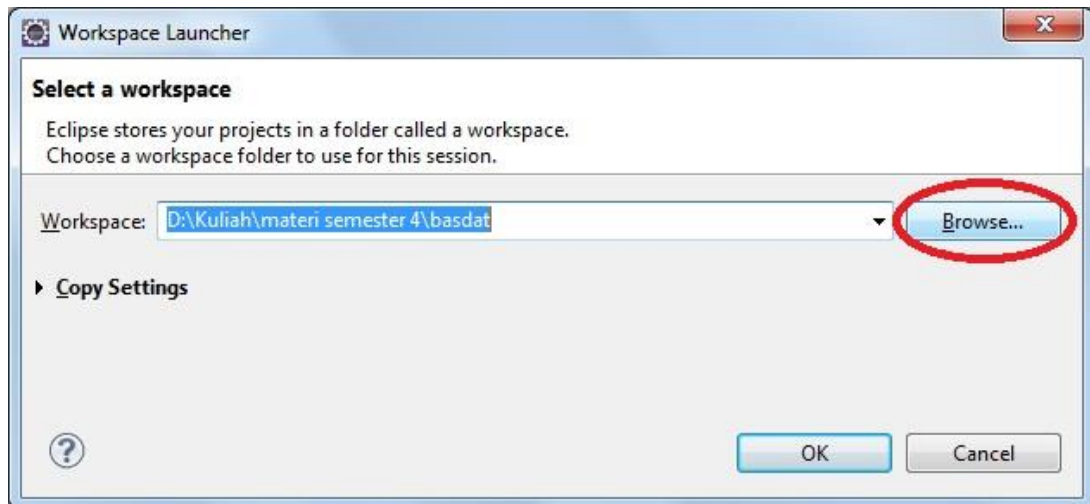
c. setelah itu akan muncul tampilan seperti dibawah ini



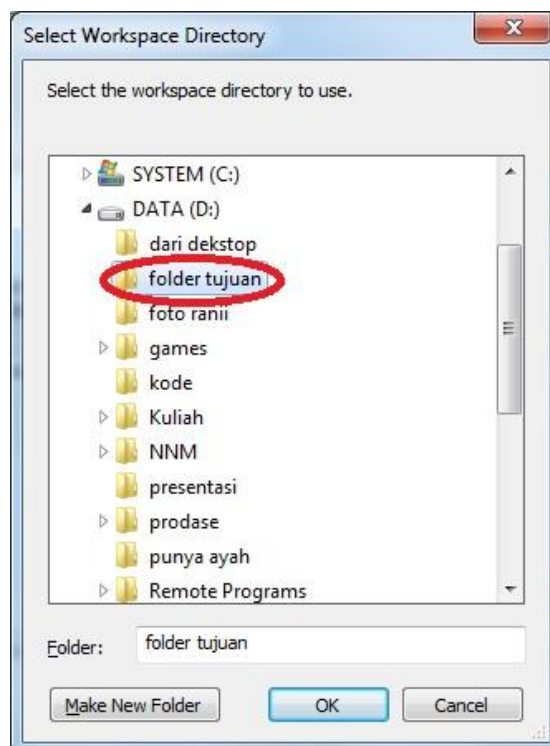
d. Untuk menyimpan project yang akan dibuat di folder yang diinginkan, maka pilih **FILE -> Switch Workspace -> other**



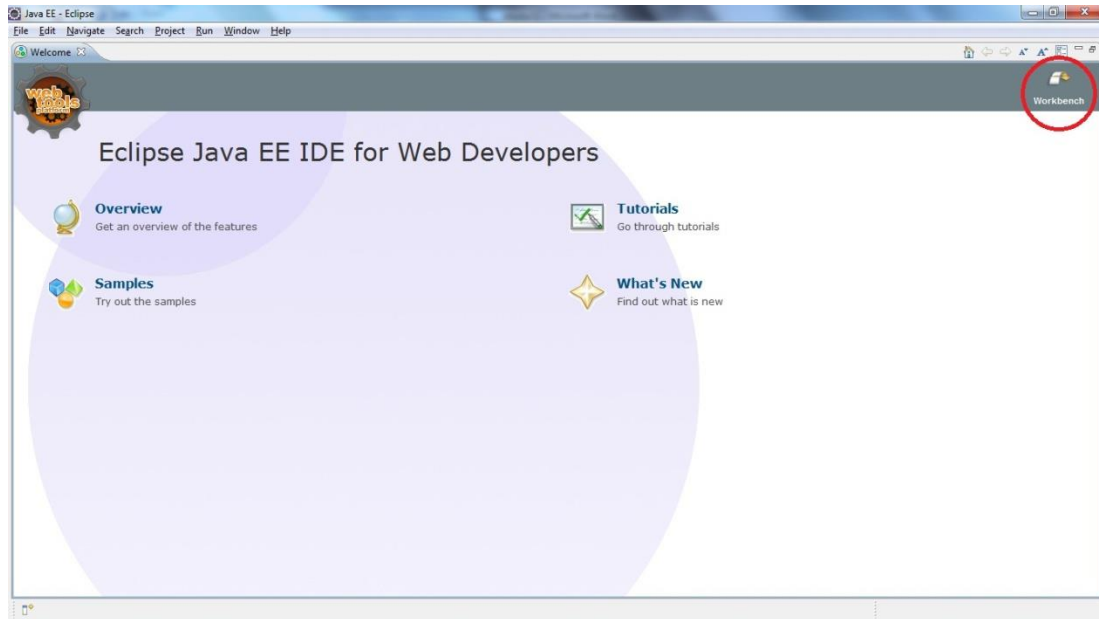
- e. Lalu akan muncul tabel dialog seperti gambar dibawah ini, pilih browse untuk memilih folder yang diinginkan.



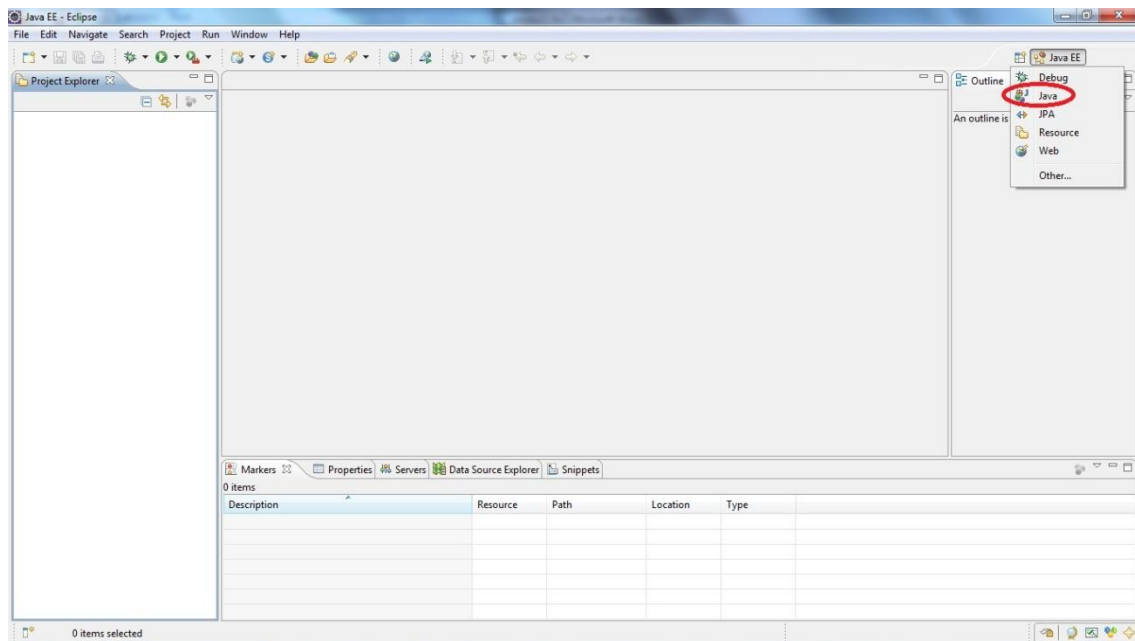
- f. lalu ketika tabel dialog seperti berikut muncul pilih folder yang anda inginkan



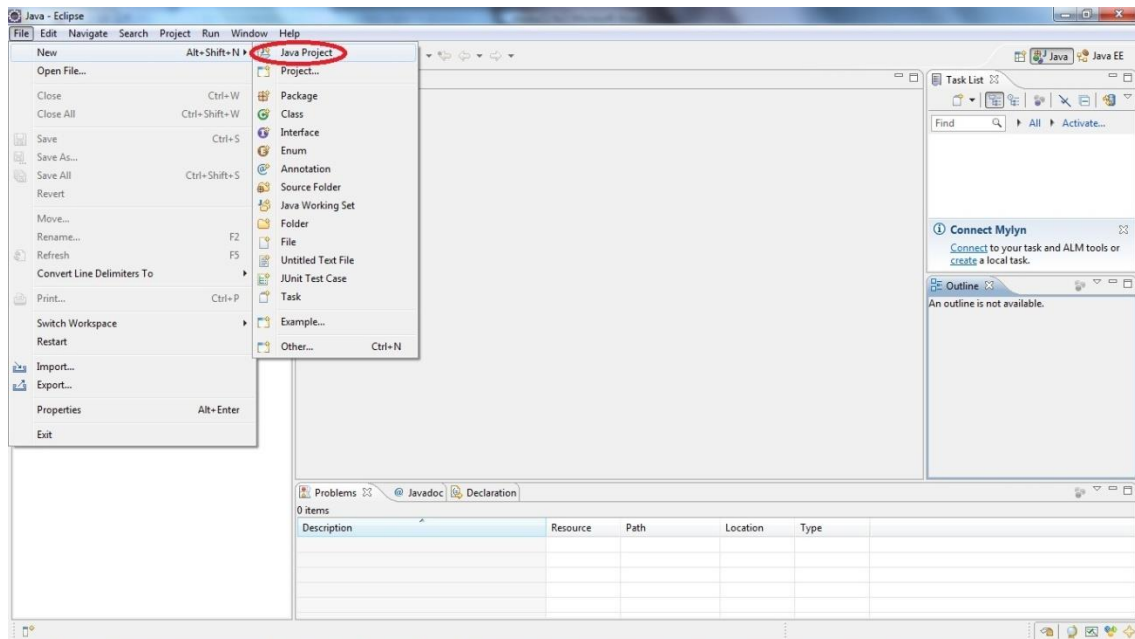
- g. setelah itu eclipse akan melakukan restart, lalu mncul tampilan sebagai berikut, pilih workbench



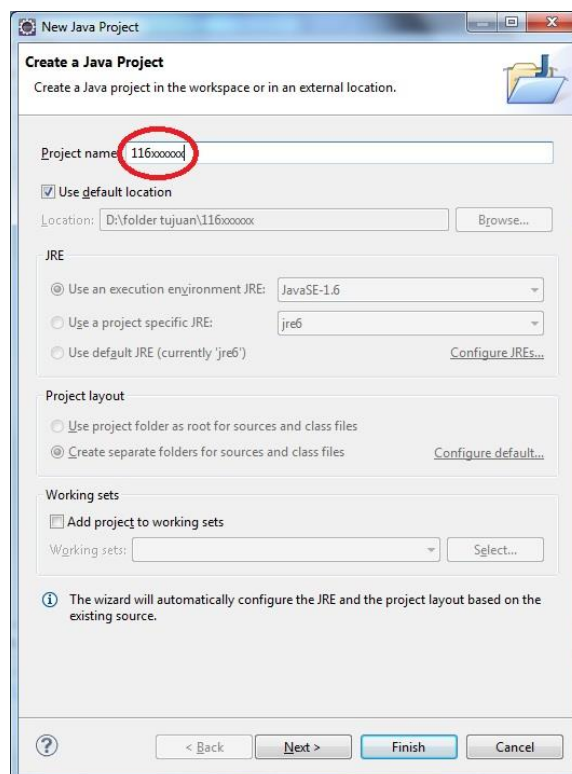
- h. lalu ubah dari java EE menjadi java pada pilihan sebagai berikut



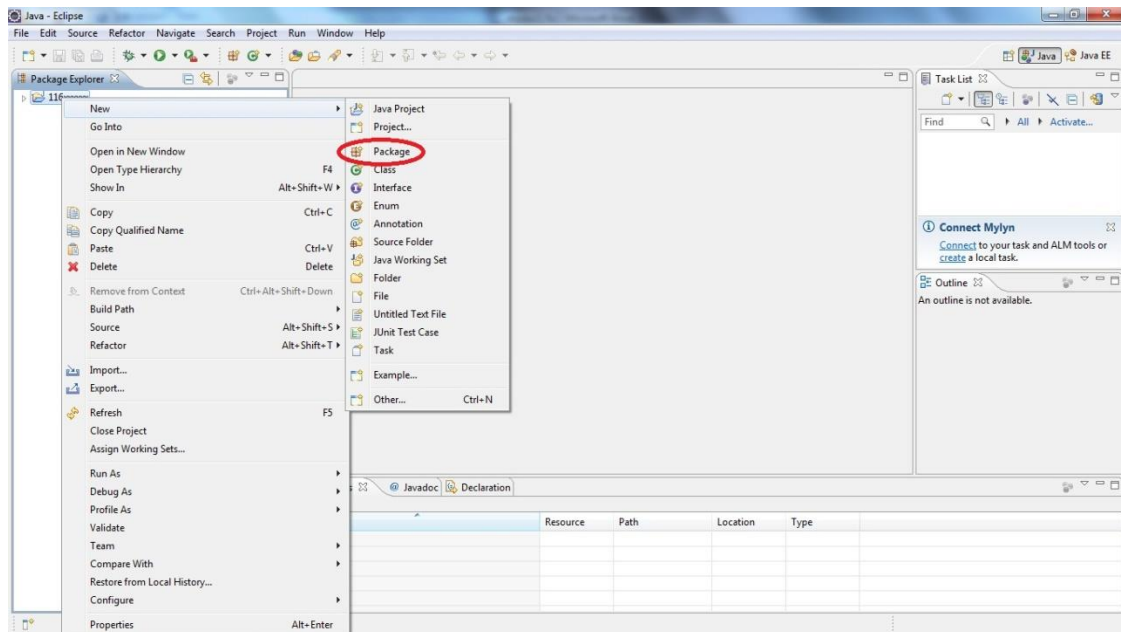
- i. Setelah itu, kita akan memulai pekerjaan kita dengan membuat project, pilih **FILE -> New -> Java Project**.



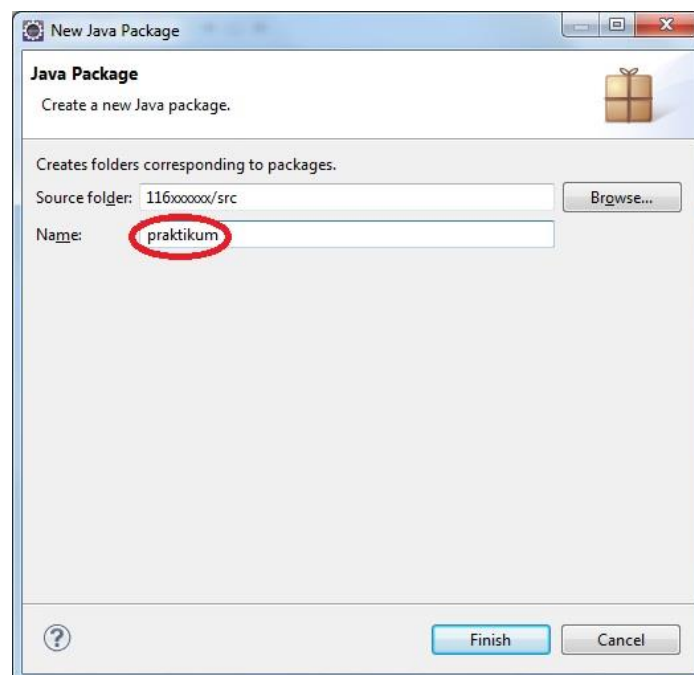
- j. Pada tabel dialog berikut, ketik nama project yang diinginkan, misal nim anda lalu klik finish.



- k. Setelah itu project yang anda buat akan terlihat pada package explorer. Lalu kita buat package, klik kanan pada project yang dibuat lalu pilih **New -> Package**.

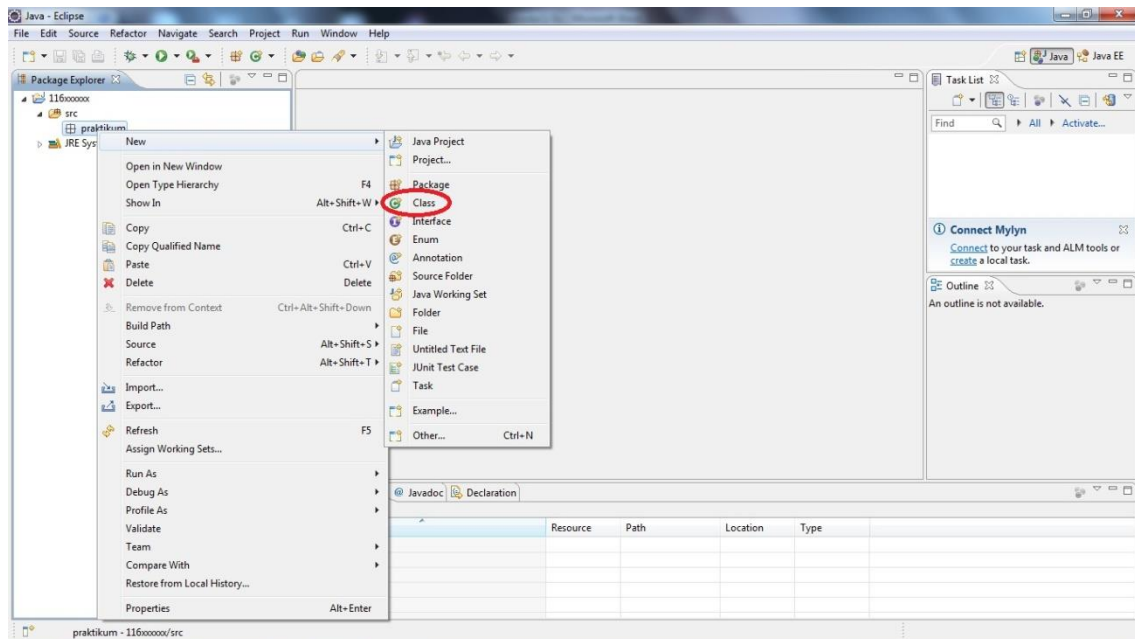


- l. Ketikkan nama package yang akan dibuat pada tabel dialog berikut lalu klik finish.



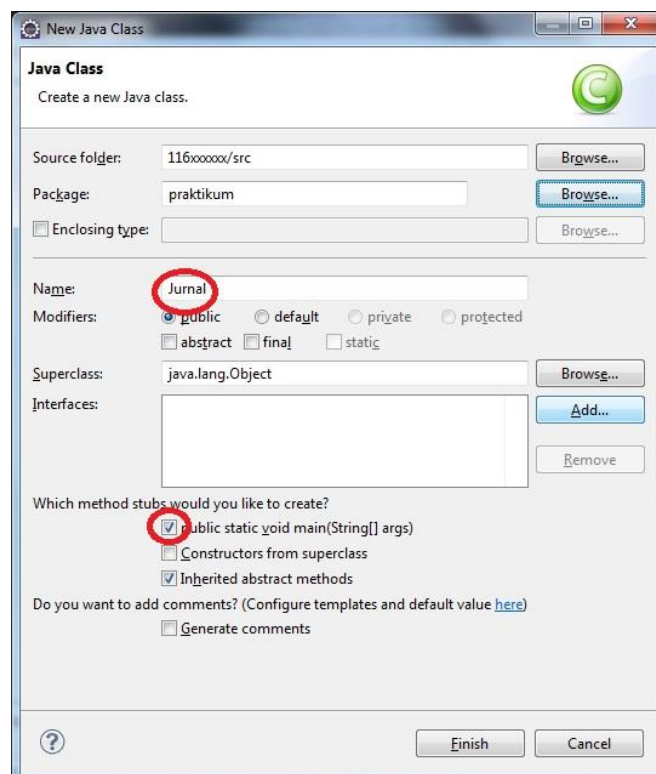
**Catatan : package bersifat optional (tidak harus dibuat)**

- m. Setelah itu, kita akan membuat class. Klik kanan pada package (jika ada) atau project, pilih **New -> Class**.



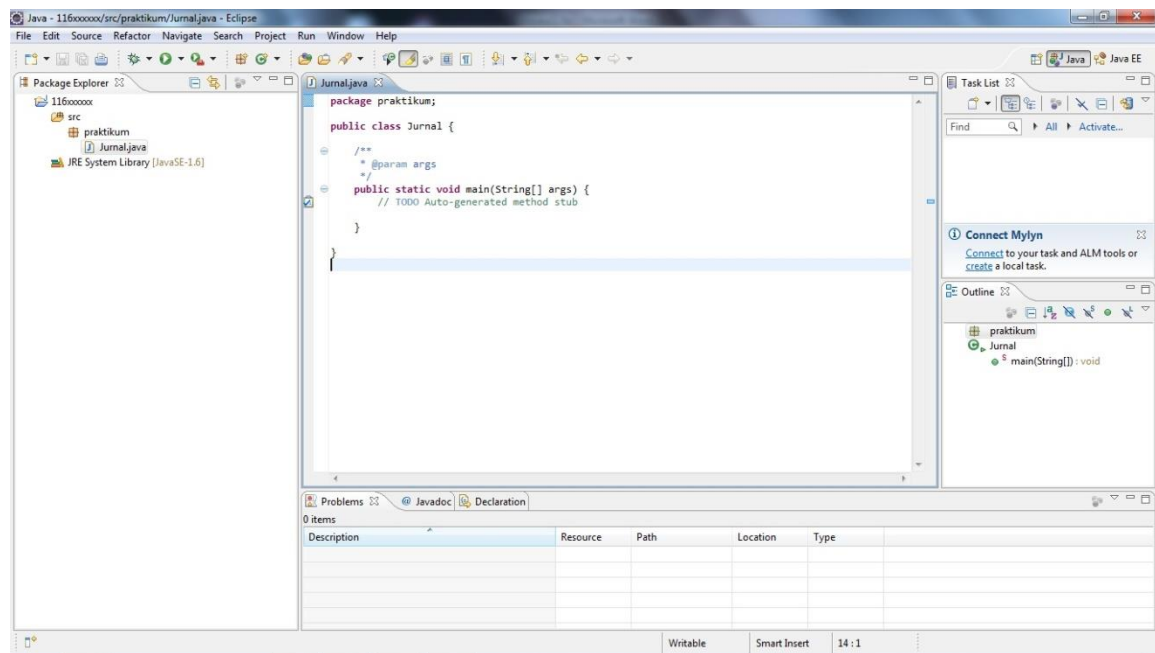
**Catatan : jika kita membuat class pada project atau tidak terdapat package pada project anda maka, class berada pada package default.**

- n. pada tabel dialog berikut, isikan nama dari kelas yang kita buat, kita ketikkan nama class yang diinginkan, misal **Jurnal**. Kemudian klik **public static void main** lalu klik **Finish**.



- o. Selanjutnya akan muncul tampilan seperti berikut.





p. Setelah itu, ketikkan kode program pada class yang baru dibuat.

## Pengkondisian

### **TUJUAN PRAKTIKUM :**

1. Praktikan memahami bentuk umum serta logika pengkondisian dalam Java
2. Praktikan dapat mengimplementasikan pengkondisian dalam program Java
3. Praktikan mampu memecahkan masalah sederhana dengan menggunakan analisa kasus dan mengimplementasikannya ke dalam bahasa pemrograman Java

### **A. PENGKONDISIAN**

#### **1. Konstruksi pengambilan keputusan**

Konstruksi pengambilan keputusan adalah konstruksi yang memungkinkan program melakukan evaluasi terhadap variable / kondisi kemudian menjalankan alur program yang sesuai dengan kondisi. Dalam hal ini program dikatakan mengambil keputusan berdasarkan hasil evaluasi variable atau kondisi.

Konstruksi pengambilan keputusan :

- a. Konstruksi if
- b. Konstruksi if...else
- c. Konstruksi ...else if ...
- d. Konstruksi switch

a. **Konstruksi if**

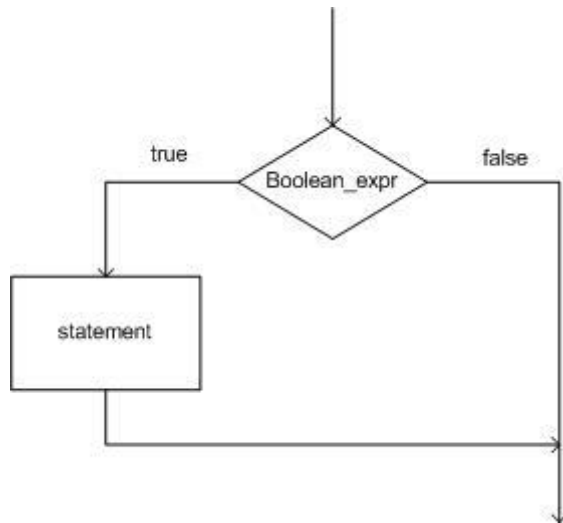


Diagram Alir untuk Konstruksi if

Aksi akan dieksekusi bila kondisi bernilai true.

Tabel 2.1 : notasi struktur IF
<b>Dalam Java</b>
<pre>If (kondisi) {     AKSI; }</pre>
<b>Contoh:</b> <pre>int a = 3; if (a&gt;0) {     System.out.println( a + “ bernilai positif” ); }</pre>

**b. Konstruksi if – else**

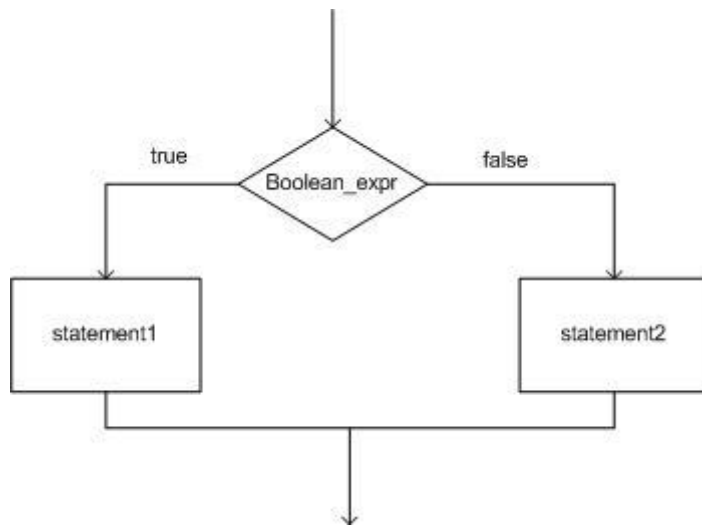


Diagram Alir untuk Konstruksi if-else

Konstruksi if-else dipakai untuk mengeksekusi salah satu dari 2 pernyataan dari syarat tertentu yang pada pada if yang dapat bernilai benar atau salah.

Tabel 2.2 : notasi struktur IF – ELSE
<b>Dalam Java</b>
<pre>If (kondisi) {     AKSI1; } else {     AKSI2; }</pre>
<b>Contoh:</b> <pre>int a = 3; if (a&gt;0) {     System.out.println( a + " bernilai positif"); } else {     System.out.println( a + " bernilai negatif"); }</pre>

AKSI1 akan dieksekusi bila kondisi bernilai true. Kalau kondisi bernilai false, maka AKSI2 akan dieksekusi.

c. Konstruksi - else if –

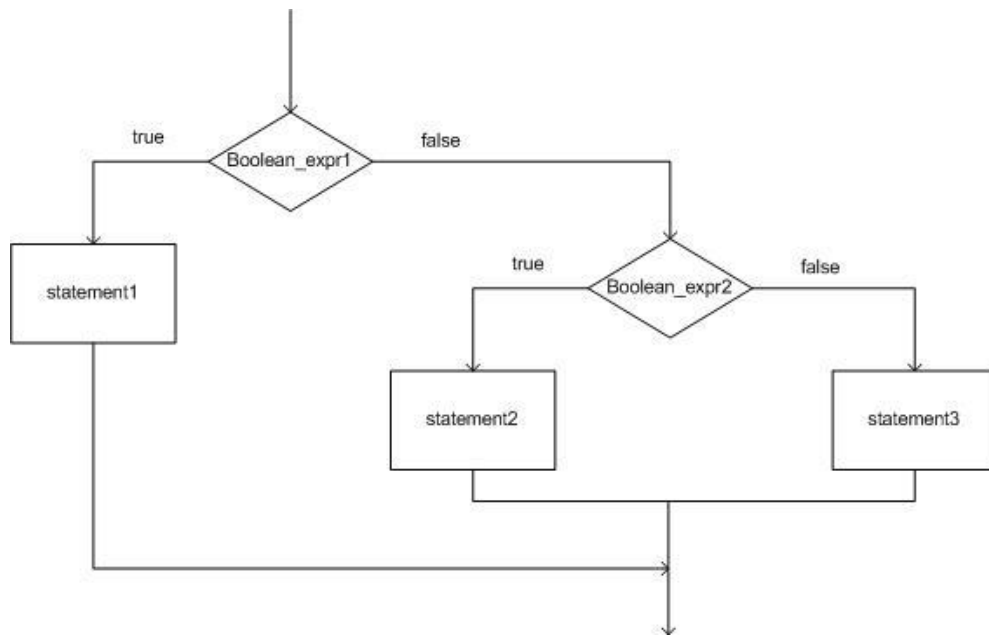


Diagram Alir untuk Konstruksi else-if

Konstruksi else-if dipakai untuk memberikan kondisi tertentu pada bagian else.

<i>Tabel 2.3 : notasi struktur - ELSE IF -</i>
<b>Dalam Java</b>
<pre> If (kondisi1) {     AKSI1; } Else If (kondisi2) {     AKSI2; } </pre>
<p><b>Contoh:</b></p> <pre> int a = 3; if (a % 2 == 0) {     System.out.println( a + " bilangan genap"); }else if ( a % 2 != 0){     System.out.println( a + " bilangan ganjil"); } </pre>

Ketika kondisi1 bernilai false, maka alur program akan menuju ke bagian else.

Selanjutnya AKSI2 diatas akan dikerjakan kalau kondisi2 bernilai true.

Silahkan anda ketik contoh program analisa kasus di bawah ini sebagai bahan latihan dan pahami kode programnya.

Contoh 2.1
Dalam Java
<pre>public class Test {     public static void main(String[] args) {         int A = 3;         int B = 6;         if (A&gt;B){             System.out.println(A + "&gt;" + B);         } else if (A &lt; B) {             System.out.println(A + "&lt;" + B);         } else {             System.out.println(A + "=" + B);         }     } }</pre>

#### d. Struktur switch

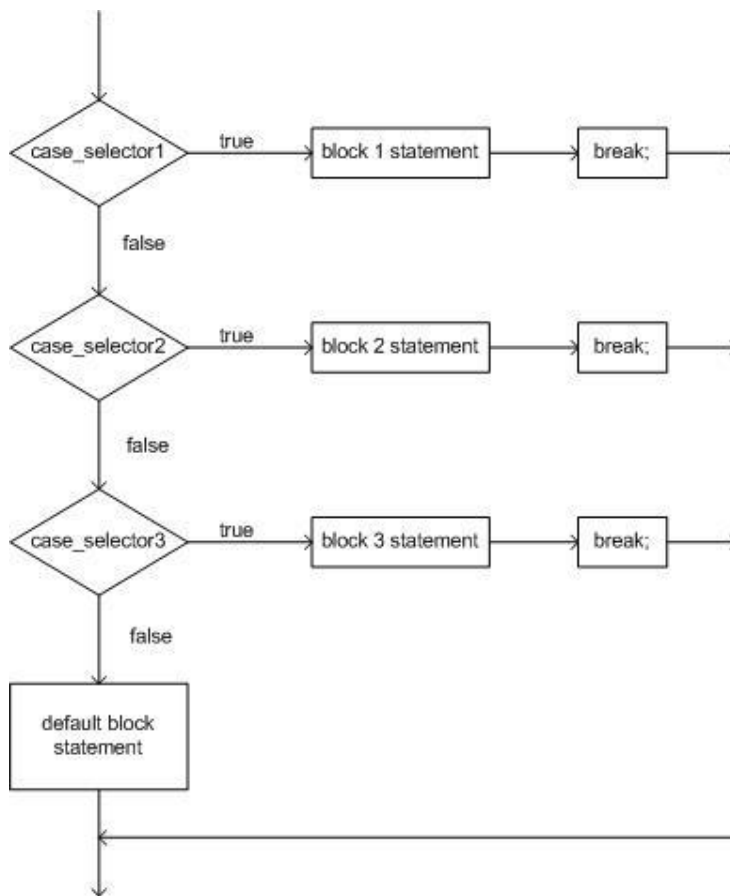


Diagram Alir untuk Konstruksi switch

Struktur switch digunakan untuk menangani banyak kemungkinan. Struktur switch melakukan evaluasi dan membandingkan ekspresi untuk semua konstanta case dan mengendalikan eksekusi program ke pernyataan case yang cocok

Tabel 2.4 : notasi struktur SWITCH
<b>Dalam Java</b>
<pre>switch ( variabel ) {     case kondisi1 : AKSI1;         case kondisi2 : AKSI2;         case kondisi-n: AKSI-n ;     }</pre>

Keterangan dalam penggunaan switch :

- *case* merupakan kata kunci yang mengindikasikan suatu nilai yang diuji.
- Kondisi tidak dapat berupa variable, ekspresi atau method, tetapi dapat berupa konstanta  $k = \{default, 1, 2, 3, \dots, n\}$

- *break* dapat digunakan dalam switch. *break* adalah pernyataan yang sifatnya optional yang mengakibatkan aliran program keluar dari blok *switch*. Jika setelah aksi tidak disertai *break*, maka aliran program akan masuk ke *case* selanjutnya.
- Default merupakan kata kunci yang mengidikasikan kondisi umum yang akan dieksekusi jika semua case yang diuji tidak sesuai dengan nilai variabel.

Berikut ini adalah contoh pemakaian Struktur **switch** pada program java untuk mencari hari yang dinotasikan dengan angka sesuai dengan inputan user .

Contoh 2.2
<b>Dalam Java</b>
<pre> public class Test {      public static void main(String[] args) {         int grade = 100;         switch(grade)         {             case 100:                 System.out.println("Excellent!");                 break;             case 90:                 System.out.println("Good Job!");                 break;             case 80:                 System.out.println("Study Harder!");                 break;             default:                 System.out.println("Sorry, You Failed!");         }     } } </pre>

**Catatan:** coba contoh 2.2, ubah nilai pilihan menjadi 2 dan perhatikan perbedaan keluarannya.



Latihan soal :

1. Buatlah program kelulusan matakuliah yang Inputanya berupa nilai. Jika nilainya kurang dari 55 maka dinyatakan tidak lulus
2. Buatlah program pembelian. Inputan berupa total pembelian, jika total pembelian paling rendah 50000 mendapat potongan 20%, sehingga jumlah yang dibayar adalah total pembelian dikurang potongan. Output berupa besarnya potongan dan jumlah yang harus dibayar.

## Perulangan

### **TUJUAN PRAKTIKUM :**

1. Praktikan memahami bentuk umum perulangan dalam Java
2. Praktikan dapat mengimplementasikan perulangan dalam program Java
3. Praktikan mampu memecahkan masalah sederhana dengan menggunakan analisa kasus dan mengimplementasikannya ke dalam bahasa pemrograman Java

### **PERULANGAN**

#### **1. Definisi Perulangan (Looping)**

Perulangan dalam algoritma didefinisikan sebagai bentuk algoritma yang berfungsi untuk mengulang perintah-perintah baris program dengan aturan tertentu. Pengulangan bertujuan untuk mengefisienkan penulisan kode program, sehingga tidak perlu dilakukan berulang-ulang kali.

#### **2. Struktur Perulangan (Looping)**

Struktur perulangan secara umum terdiri atas dua bagian, yaitu:

- a. Kondisi perulangan, yaitu berupa ekspresi Boolean yang harus dipenuhi untuk melaksanakan kondisi perulangan. Kondisi ini mengakibatkan suatu kondisi perulangan akan berhenti pada saat kondisi Boolean tersebut terpenuhi.
- b. Badan (body) perulangan, yaitu suatu aksi (bagian algoritma) yang harus diulang selama kondisi yang ditentukan untuk perulangan tersebut masih terpenuhi.

#### **3. Jenis Perulangan dalam Algoritma**

Dalam modul ini akan dibahas beberapa jenis perulangan dalam bahasa pemrograman Java, antara lain :

- a. *While*
- b. *Do-while*
- c. *For*
- d. *Nested loop*

Berikut akan kita bahas bentuk-bentuk tersebut satu per satu:

### a. While

While adalah bentuk perulangan yang memiliki jumlah perulangan sesuai dengan suatu kondisi logika tertentu. *Do-while loop* mirip dengan *while-loop*. Pernyataan di dalam *do-while loop* akan dieksekusi beberapa kali selama kondisi bernilai benar(*true*).

Tabel 3.1.1 : notasi struktur While

Inisialisasi
While ( kondisi ){
// statemen yang akan diulang
....
iterasi
}

Maksud dari bentuk di atas adalah selama kondisi\_perulangan terpenuhi atau bernilai benar (*true*), maka statemen akan terus dilaksanakan sampai kondisi\_perulangan bernilai salah (*false*). Jumlah perulangan ini minimal 0 kali, karena pengecekan kondisi dilakukan di awal.

```
import java.util.Scanner;

public class while1 {

    public static void main(String[] args) {
        int x = 0;
        while (x<5) {
            String nama = "";
            Scanner input = new Scanner (System.in);
            System.out.println ("Nama : "+nama);
            x++;
            nama = input.next();
        }
    }
}
```

### b. Do-while

Sama halnya dengan *while*, *do-while* juga akan menjalankan looping selama kondisi\_perulangan terpenuhi atau bernilai benar (*true*). Berbeda pada perulangan *while*, pada perulangan *do-while* pengecekan kondisi (syarat) perulangan dilakukan setelah

eksekusi statement yang diulang. Sehingga statement dalam blok do-while paling sedikit dieksekusi satu kali. Bentuk umum perulangan do-while :

Tabel 3.2.1 : notasi struktur While
Inisialisasi Do { // statemen yang akan diulang ... Iterasi } while ( kondisi );

Maksud dari bentuk di atas adalah statemen akan dilakukan sebelum ada pemeriksaan kondisi perulangan.

<pre>import java.util.*;  public class dowhile1 {      public static void main(String[] args) {          int x = 0;         do {             String nama="";             Scanner input = new Scanner (System.in);             System.out.println("Nama : "+nama);             x++;             nama = input.next();         }         while (x&lt;5);     } }</pre>
---

### c. For

Bentuk for digunakan untuk perulangan yang memiliki jumlah perulangan yang telah dipastikan sebelumnya. Bentuk umum dari perulangan traversal adalah sebagai berikut:

Tabel 3.3.1 : notasi struktur for
<b>For</b> ( inisialisasi; kondisi; iterasi ) { // statemen yang akan diulang

```
}
```

Maksud dari bentuk di atas adalah akan dilaksanakan AKSI sebanyak N kali, dimana nilai N adalah penyesuaian kondisi perulangan dengan kondisi awal. Perulangan akan berhenti dilaksanakan jika kondisi perulangan bernilai salah (false).

```
public class looping {  
  
    public static void main(String[] args) {  
        int i;  
        for (i=1;i<=5;i++){  
            System.out.println(i+" ");  
        }  
    }  
}
```

#### d. Nested loop

Nested loop merupakan perulangan di dalam perulangan. Pelajari contoh berikut dan cobalah untuk mengetahui hasilnya :

```
public class looping {  
  
    public static void main(String[] args) {  
        int i;  
        int j;  
        for (i=1;i<=5;i++){  
            for (j=1;j<=5;j++){  
                System.out.print("* ");  
            }  
            System.out.println("");  
        }  
    }  
}
```

**SOAL LATIHAN:**

1. Buat program log-in admin lab prodase dengan :

username : prodase

password : 12345

Bila salah memasukan username atau password, akan muncul peringatan "anda tidak berhak mengakses program ini".

2. Buatlah program dengan keluaran seperti berikut:

1

2 1 2

3 2 1 2 3

4 3 2 1 2 3 4

### **TUJUAN PRAKTIKUM :**

1. Mendeklarasikan dan membuat array
2. Mengakses elemen-elemen array
3. Menentukan jumlah element dalam sebuah array
4. Mendeklarasikan dan membuat array multidimensi

### **A. PENGENALAN ARRAY**

Dalam mendeklarasian variable, kita sering menggunakan tipe data yang sama namun dengan nama variable atau identifier yang berbeda – beda. Sebagai contoh, kita memiliki tiga variable dengan tipe data *int* dengan identifier yang berbeda tiap variabelnya.

```
int angka1;
```

```
int angka2;
```

```
int angka3;
```

```
angka1 = 10;
```

```
angka2 = 20;
```

```
angka3 = 30;
```

Pada contoh di atas, kode tersebut kurang efektif karena harus menginisialisasi dan menggunakan tiap variable padahal dalam java atau pemrograman lain terdapat kemampuan lain untuk menampung variable – variable dengan tipe data yang sama dan dapat dimanipulasi dengan efektif.

Tipe variable ini disebut dengan **array**. Sebuah array akan menyimpan beberapa item data dengan tipe data yang sama di dalam sebuah blok memori yang berdekatan yang kemudian dibagi menjadi beberapa slot.

## B. Pendeklarasian Array

Array harus di deklarasikan seperti layaknya sebuah variable, apabila Anda mendeklarasikan array, maka harus membuat sebuah list dari tipe data, yang diikuti oleh tanda kurung siku buka dan kurung siku tutup, yang diikuti oleh nama identifier.

Contoh :

```
//tipe [ ] namaArray;  
Int [ ] nilai;
```

Atau

```
//tipe namaArray[];  
Int nilai[];
```

Setelah pendeklarasian, kita harus membuat array dan menentukan beberapa panjangnya dengan sebuah konstruktor, proses ini didalam java disebut instantiasi (kata dalam java yang berarti membuat). Untuk meng-instantiasi sebuah objek, kita membutuhkan sebuah konstruktor. Kita akan membicarakan lagi meng-instantiasi dan pembuatan konstruktor pada praktikum selanjutnya.

Contoh :

```
//deklarasi objek  
//format penulisan = tipe namaArray[];  
Int nilai[];  
  
//instantiasi objek  
//format penulisan = variableArray = new tipe[jumlahElemen];
```



```
nilai = new int[100];
```

*Atau bisa juga ditulis*

```
//deklarasi dan instantiasi
```

```
//format penulisan = tipe namaArray[] = new tipe[jumlahElemen];
```

```
Int nilai[] = new int [100];
```

Pada contoh diatas, deklarasi akan memberitahukan kepada compiler java, bahwa identifier `ages` akan digunakan sebagai nama array yang berisi data-data integer, dan kemudian untuk membuat atau meng-instantiasi sebuah array baru yang terdiri dari 100 elemen.

Selain menggunakan sebuah keyword baru untuk menginstantiasi array, juga dapat secara otomatis mendeklarasikan array, membangun, kemudian memberitahukan sebuah nilai (value).

Sebagai contoh,

```
//membuat sebuah array yang berisi variabel-variabel boolean pada sebuah identifier.
```

```
//array ini terdiri dari 4 elemen yang diinialisasikan sebagai value {true,false,true,false}
```

```
boolean result[]={true,false,true,false};
```

```
//membuat sebuah array yang terdiri dari penginialisasian 4 variabel double bagi value{100,90,80,75}
```

```
double[4]={100,90,80,75};
```

```
//membuat sebuah array String dengan identifier days. Array ini terdiri dari 7 elemen.
```

```
String days[]={“mon”,“tue”,“wed”,“thu”,“fri”,“sat”,“sun”};
```

### C. MENGAkses ELEMEN ARRAY

Untuk mengakses elemen – elemen yang terdapat dalam array, kita membutuhkan nomor atau disebut dengan index atau subscript. Nomor – nomor index atau subscript sudah diberikan dalam array, sehingga program atau programmer dapat mengaksesnya bila dibutuhkan. Perlu dicatat untuk nomor index array dimulai dari angka nol dan terus bertambah hingga list value array tersebut berakhir. Index array bertipe data *int* dan perlu diingat lagi index di dalam array dimulai dari 0 sampai dengan panjang array dikurangi 1.

Sebagai contoh, perhatikan potongan kode program di bawah ini :

```
int angka [] = {6,7,8,9,10};           //elemen – elemen array

System.out.println ( angka [2] );      //mengakses elemen array
```

Maka akan ditampilkan angka 8 pada saat di run. Perlu diingat kembali bahwa index array dimulai dari 0 sehingga pada kasus di atas akan menampilkan angka 8 bukan angka 7.

Pada saat array dideklarasikan atau dikonstruksi, nilai yang disimpan dalam array akan diinisialisasikan sebagai nol. Sehingga jika kita menggunakan tipe data reference seperti *String*, array tersebut tidak akan diinisialisasikan menjadi string kosong ( "" ). Sehingga untuk array *String* kita harus menginisialisasi valuenya secara eksplisit.

#### ***Petunjuk penulisan program:***

1. Biasanya, lebih baik menginisialisasi atau meng-instantiate array setelah anda mendeklarasikannya

```
int[] arr = new int[100];
lebih disarankan daripada,
int[] arr;
arr=new int[100];
```

2. elemen-elemen dalam n-elemen array memiliki index dari 0 sampai n-1. Perhatikan disini bahwa tidak ada elemen array `arr[n]`. Hal ini akan menyebabkan array-index out-of-bounds exception.

3. Anda tidak dapat mengubah ukuran dari sebuah array.

Berikut ini adalah contoh, bagaimana untuk mencetak seluruh elemen didalam array. Dalam contoh ini digunakanlah loop, sehingga kode kita menjadi lebih pendek.

```
package abc;

public class modul {

    public static void main(String[] args) {

        int[] arr = new int[100];

        for(int i=0;i<20;i++) {

            System.out.print(arr[i]);

        }

    }

}
```

#### Contoh program :

```
Import java.util.Scanner;

public class {

    public static void main (String [ ] args){

        Scanner input = new Scanner (System.in);

        final int angka [] = new int [5];

        System.out.println ("-Masukan nilai elemen array-");

        for (int i=0; i<angka.length; i++){

            System.out.print ("elemen ke-"+(i+1)+" = ");

            angka [i] = new input.nextInt();

        }

        System.out.println ();

        System.out.printf("%s 8s \n", "Index", "Values"); // kolom heading

        //keluaran masing2 nilai elemen array

        for (int i=0;i<angka.length;i++){

            System.out.printf("%5d %8d \n", counter, angka[counter]);

        }

    }

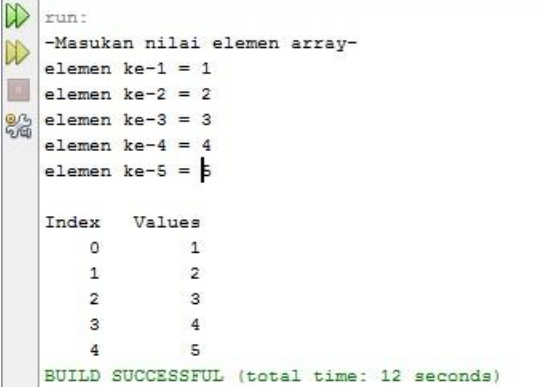
}
```

```

    }
}

```

## Output :



```

run:
-Masukan nilai elemen array-
elemen ke-1 = 1
elemen ke-2 = 2
elemen ke-3 = 3
elemen ke-4 = 4
elemen ke-5 = 5

Index  Values
    0      1
    1      2
    2      3
    3      4
    4      5

BUILD SUCCESSFUL (total time: 12 seconds)

```

`final int angka = new int [5];` menunjukkan bahwa panjang array *angka* adalah 5. Inputan menggunakan *Scanner* sehingga value untuk array tersebut akan dimasukan melalui keyboard. Pada sebelumnya kita menggunakan `System.out.println` untuk mencetak.

pada kasus ini sedikit berbeda ada `System.out.printf`, dalam fungsinya sama yaitu mencetak nilai ke layar, pada `System.out.printf` untuk mengisi nilai atau variable menggunakan % (persen) di ikuti tipe variabel s (String ), d (integer ). Dan ada spasi di beri notasi angka menunjukkan berapa jumlah spasi yang akan di gunakan. ("`%s%8s\n`", "*Index*", "*Values*"), String "*Index*" akan di letakkan di (%s) yg awal kemudian String "*Values*" di letakkan di (%8s) ada delapan spasi dari String "*index*". **(coba diubah-ubah/modifikasi).**

## D. Panjang array

untuk mengetahui berapa banyak elemen didalam sebuah array, anda dapat menggunakan **length (Panjang)** field dalam array. Panjang field dalam array akan mengembalikan ukuran dari array itu sendiri.

Sebagai contoh,

`arrayName. length`

pada contoh sebelumnya, kita dapat menuliskannya kembali seperti berikut ini,

***Petunjuk penulisan program:***

- 1. Pada saat pembuatan loop untuk memproses elemen-elemen dalam array, gunakanlah length field didalam pernyataan pengkondisian dalam loop. Hali ni akan menyebabkan loop secara otomatis menyesuaikan diri terhadap ukuran array yang berbeda-beda*
- 2. Pendeklarasian ukuran array di java, biasanya digunakan constant untuk mempermudah. Sebagai contoh,*

```
final int ARRAY_SIZE = 1000; //pendeklarasian constant
```

**Contoh Program**

```
Import java.util.Scanner;
```

```
public class contoh1 {
```

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

```
        int array_length = 10;
```

```
        //jumlah elemen array
```

```
        int array[] = new int [array_length];
```

```
        //besar array_length = 10
```

```
        Scanner input = new Scanner (System.in);
```

```
        System.out.println ("Masukan nilai elemen array");
```

```
        for (int i=0;i<array.length;i++){
```

```
            System.out.print("Elemen ke "+(i+1)+" = ");
```

```
            array [ i ] = input.nextInt();
```

```
        }
```

```
        //perulangan input data
```

```
        System.out.printf ("%s %s \n", "index", "values");
```

```
        for (int i=0;i<array.length;i++){
```

```
            System.out.printf ("%5d %8d \n", i, array [ i ]);
```

```

    }
}
}

```

Akan muncul pernyataan untuk mengisi nilai elemen array dari elemen ke-1 hingga elemen ke-10. Pada contoh kali ini, akan dimasukan angka sebagai berikut :

```

Masukan nilai elemen array
Elemen ke 1 = 1
Elemen ke 2 = 2
Elemen ke 3 = 3
Elemen ke 4 = 4
Elemen ke 5 = 5
Elemen ke 6 = 6
Elemen ke 7 = 7
Elemen ke 8 = 8
Elemen ke 9 = 9
Elemen ke 10 = 10

```

Maka hasil yang akan muncul adalah :

index	values
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10

Pada contoh kasus di atas :

*int array\_length = 10;* menunjukkan inisialisasi constanta *array\_length* dengan nilai = 10, *int array[] = new int [array\_length];* membuat dan menginisialisasi objek array kosong untuk memasukkan nilai inputan dari perulangan. Dimana di contoh program tersebut jumlah *array\_length = 10*  
**(coba diubah-ubah/modifikasi).**

## E. ARRAY MULTI DIMENSI

Array multi dimensi merupakan array yang ada di dalam array. Array multi dimensi juga dapat diartikan sebagai matrix yang terdiri dari baris dan kolom. Array multi dimensi dideklarasikan dengan menambah tanda kurung siku setelah nama array.

	Column 0	Column 1	Column 2	Column 3
Row 0	a[ 0 ][ 0 ]	a[ 0 ][ 1 ]	a[ 0 ][ 2 ]	a[ 0 ][ 3 ]
Row 1	a[ 1 ][ 0 ]	a[ 1 ][ 1 ]	a[ 1 ][ 2 ]	a[ 1 ][ 3 ]
Row 2	a[ 2 ][ 0 ]	a[ 2 ][ 1 ]	a[ 2 ][ 2 ]	a[ 2 ][ 3 ]

Column index  
Row index  
Array name

### Contoh deklarasi array :

```
//array integer dengan ukuran 100 x 100
```

```
int twoD [ ] [ ] = new int [100] [100];
```

```
//contoh array String dengan ukuran 3 x 3
```

```
String mahasiswa[ ] [ ] = {{ "budi", "andi"},  
                             {"tono", "rudi"},  
                             {"okta", "tasya"}};
```

Untuk mengakses elemen array multi dimensi, caranya hampir sama dengan array satu dimensi. Hanya saja dalam array multi dimensi kita perlu menentukan letak kolom dan baris dari elemen tersebut. misalnya untuk mengakses elemen pertama dari array *mahasiswa* di atas kita tuliskan :

```
System.out.print ( mahasiswa [0] [0] );
```

Kode tersebut akan mencetak string "budi" pada layar.

### Contoh program :

```
import java.util.Scanner;
```

```
class prodase {
```

```

public static void main (String [] args){
    Scanner in = new Scanner (System.in);
    String mahasiswa [ ] [ ] = new String [3][3];
    System.out.println ("Inputkan :");
    for (int i=0;i<3;i++){
        for (int j=0;j<3;j++){
            System.out.print ("baris ke-"+(i+1)+" kolom ke-"+(j+1)+" = ");
            mahasiswa [i][j] = in.next ();
        }
    }

    System.out.println ();
    System.out.println ("Output :");
    for (int i=0;i<3;i++){
        for (int j=0;j<3;j++){
            System.out.print (mahasiswa [i][j] + " ");
        }
    }
}

```

Cobalah kode program di atas dan perhatikan keluaran apa yang dilakukannya. pada program di atas kita perhatikan :

```
String mahasiswa [ ] [ ] = new String [3][3];
```

Merupakan proses pendeklarasian sebuah array yang nantinya akan diinputkan elemen – elemen arraynya.

```

for (int i=0; i<3; i++){

    for (int j=0; j<3; j++){

```



```

        System.out.print ("baris ke-"+(i+1)+" kolom ke-"+(j+1)+" = ");

        mahasiswa [i][j] = in.next ();

    } }

```

Pada potongan kode di atas, merupakan kode yang digunakan untuk proses penginputan elemen – elemen array.

```

for (int i=0; i<3; i++){

    for (int j=0; j<3; j++){

        System.out.print (mahasiswa [i][j] + " ");

    }

}

```

Walaupun kode di atas menggunakan perulangan *for* seperti kode sebelumnya, namun fungsinya berbeda, pada potongan kode di atas digunakan untuk mengakses elemen array dan menampilkannya di layar.

#### **F. Input Array menggunakan fungsi *Random***

Nilai random merupakan nilai bilangan acak, dalam program untuk men-Generate bilangan random harus membuat objek random dulu. Sintak random di java harus menggunakan java import.

```

import java.util.Random;

```

#### **Sintaks :**

```

Random objekRandom = new Random();

```

**Contoh :**

1. Array satu dimensi dengan input random

```
import java.util.Random;

public class arrayRandomSatuDimensi {

    public static void main (String [][] args){
        //instantiasi untuk pembentukan objek random
        Random rdm = new Random ();
        int nilairandom; //deklarasi variable untuk menampung objek random
        int a [] = new int [ 4 ]; //instantiasi objek arrar 1 dimensi
        System.out.println ("-Array satu dimensi dengan input random-\n");
        for (int i=0;i<a.length;i++){
            //assignment nilai random ke variable integer
            //(i+1) salah untk melakukan random setiap kali perulangan
            nilairandom = rdm.nextInt(i+1);
            a[i] = nilairandom;
            System.out.println ("Nilai random array 1 dimensi index ke["+1+"] = "+a[i]);
        }
        System.out.println ();
        System.out.printf("%s %8s \n, "index", "values");
        for (int i=0;i<a.length;i++){
            System.out.printf("%5d %8d \n", i, a[i]);
        }
    }
}
```

2. Array multidimensi dengan input random

```
import java.util.Random;

public class ArrayRandomMultidimensi {

    public static void main (String [] args){
        //instantiasi untuk pembentukan objek random
        Random rdm = new Random ();
        int nilairandom; //deklarasi variable untuk menampung objek random
```

```

int b[] [] = new int [4][4];//initaliasi objek array 2 dimensi
System.out.println ("-Array multidimensi dengan input random-\n");
for (int i=0;i<b.length;i++){
    for (int j=0;j<b.length;j++){
        nilairandom = rdm.nextint (i+1);
        b[i][j] = nilairandom;
        System.out.print (b[i][j]+" ");
    }
    System.out.println;
}
}

```

Coba run beberapa kali, dapat dilihat angka di dalam array tersebut berubah-ubah sesuai dengan fungsi Random.

#### Latihan Soal:

1. Buatlah sebuah array yang menampung nama – nama hari dalam seminggu. Gunakan Scanner untuk inputnya dan perulangan untuk menampilkannya!
2. Buatlah program penjumlahan 2 matriks berdimensi 2! (gunakan array 2 dimensi dan input random)

### TUJUAN PRAKTIKUM :

1. Memahami konsep method pada java.
2. Mengerti esensi penggunaan method dalam Java.
3. Memahami bentuk umum method.

#### A. Class dan Object

Class adalah cetak biru (rancangan) atau prototype atau template dari objek. Kita bisa membuat banyak objek dari satu macam class. Class mendefinisikan sebuah tipe dari objek.

Di dalam class kita dapat mendeklarasikan variabel dan menciptakan objek (instansiasi). Sebuah class mempunyai anggota yang terdiri dari atribut dan method. Atribut adalah semua field identitas yang kita berikan pada suatu class, misal class manusia memiliki field atribut berupa nama dan umur. Method dapat kita artikan sebagai semua fungsi ataupun prosedur yang merupakan perilaku (behaviour) dari suatu class.

Bagian-bagian dari sebuah Class secara umum penulisan class terdiri atas 2 bagian yakni:

#### 1. Class Declaration

Bentuk Umum :

```
[modifier] class <nama_kelas>
{
    ...
    ...
    <class body>
    ...
    ...
}
```

[modifier] adalah pengaturan level akses terhadap kelas tersebut. Dengan kata lain, modifier ini akan menentukan sejauh mana kelas ini dapat digunakan oleh kelas atau package lainnya. Adapun macam-macam modifier ialah :

- *kosong / default / not specified*

Kelas tersebut dapat diakses oleh kelas lain dalam satu package.

- *public*

Kelas tersebut dapat dipakai dimanapun, maupun kelas lain atau package lain.

- *private*

Kelas tersebut tidak dapat diakses oleh kelas manapun.

## 2. Class Body

Class Body merupakan bagian dari kelas yang mendeklarasikan kode program java.

Class Body tersusun atas:

- Konstruktor
- Variable Instance* (Atribut)
- Method* (dikenal juga sebagai function atau def)

Untuk dapat menggunakan kelas yang telah didefinisikan, anda harus membuat sebuah objek dari kelas tersebut (*instance class*), dengan *syntax*:

```
NamaKelas namaObjek = new NamaKelas ( [parameter] );
```

Contoh:

```
Hitungluas segitiga = new Hitungluas();
```

## 3. Instance Variables (Atribut)

Suatu objek dapat dibedakan berdasarkan sifat (behavior) yang berbeda. objek juga dapat dibedakan berdasarkan atributnya. Misalnya burung dapat dibedakan berdasarkan suara kicauan, warna bulu, bentuk tubuh, dsb. . Dalam bahasa lain dikenal juga sebagai *property* yang mana merupakan ciri-ciri dari sebuah objek.

Atribut yang membedakan suatu *instance* objek burung yang satu dengan yang lainnya disebut **instance variable**.

Bentuk Umum :

```
[modifier] <type_data> <nama_variabel> = [nilai_default];
```

Contoh :

```
public double tinggi;  
private int berat = 70;
```

Modifier untuk atribut, yaitu public, protected, private. Penjelasan modifier atribut serupa dengan penjelasan modifier pada kelas.

Adapun perbedaan *local* dan *instance variable* adalah :

1. *Instance variable* dideklarasikan di dalam kelas tetapi tidak di dalam method.

```
class segitiga {  
    double tinggi = 15.2; // ini adalah variabel instance  
    String jenis; // ini adalah variabel instance  
    int tambah() {  
        return 3;  
    }  
}
```

2. *Local variable* dideklarasikan di dalam method.

```
int tambah() {  
    int total = tinggi * 2; // total adalah variabel local  
    return total;  
}
```

Object adalah instance dari class. Jika class secara umum merepresentasikan (template) sebuah object, sebuah instance adalah representasi nyata dari class itu sendiri.

Contoh : Dari class Fruit kita dapat membuat object Mangga, Pisang, Apel dan lain lain.

#### 4. Membuat object

Untuk membuat object, kita menggunakan perintah `new` dengan sebuah nama class yang akan dibuat sebagai instance dari class tersebut. Contohnya:

```
String str = new String();
```

```
Random r = new Random();
```

```
Pegawai p2 = new Pegawai();
```

```
Date hari = new Date();
```

**hari** adalah object reference dari class `Date` yang akan digunakan untuk mengakses class `Date`. Sedangkan operator `new` adalah operator yang akan menghasilkan hari sebagai reference ke instance dari class `Date()`.

#### Konstruktor

Tipe khusus method yang digunakan untuk menginstansiasi atau menciptakan sebuah objek. Nama constructor = nama kelas. Constructor TIDAK BISA mengembalikan nilai. Tanpa membuat constructor secara eksplisit-pun, Java akan menambahkan constructor default secara implisit. Tetapi jika kita sudah mendefinisikan minimal sebuah constructor, maka Java tidak akan menambah constructor default. Constructor default tidak punya parameter. Constructor bisa digunakan untuk membangun suatu objek, langsung mengeset atribut-atributnya. Konstruktor seperti ini harus memiliki parameter masukkan untuk mengeset nilai atribut. Access Modifier constructor selayaknya adalah `public`, karena constructor akan diakses di luar kelasnya. Cara panggil constructor adalah dengan menambah keyword "`new`". Keyword `new` dalam deklarasi ini artinya kita mengalokasikan pada memory sekian blok memory untuk menampung objek yang baru kita buat.

```
[modifier] namaclass (parameter) {  
  
    Body constructor;  
  
}
```

### Contoh kode program

```
package DemoManusia;

public class Lagu {

    private String band;

    private String judul;

    public void IsiParam(String judul,String band) {

        this.judul = judul;

        this.band = band;

    }

    public Lagu() {

        judul = "null";

        band = "null";

    }

    public void cetakKeLayar() {

        System.out.println("Judul : " + judul + "\nBand : " + band);

    }

}

package DemoManusia;

public class DemoLagu {

    public static void main(String[] args) {

        Lagu song = new Lagu();

        song.cetakKeLayar();

    }

}

Output :

Judul : null

Band : null
```



Pada kode program di atas terdapat konstruktor yang akan mengisi atribut nama dan band dengan nilai = "null". Jadi saat class Lagu di instansiasi maka nilai atribut nama dan band diisi dengan "null".

### C. Method

Sebuah method adalah bagian-bagian kode yang dapat dipanggil oleh kelas, badan program atau method lainnya untuk menjalankan fungsi yang spesifik di dalam kelas. Secara umum method dalam java adalah sebuah fungsi.

Berikut adalah karakteristik dari method :

1. Dapat mengembalikan / melaporkan nilai balikkan (*return value*) atau tidak (*void*)
2. Dapat diterima beberapa parameter yang dibutuhkan atau tidak ada parameter sama sekali. Parameter bisa juga disebut sebagai argumen dari fungsi. Parameter berguna sebagai nilai masukkan yang hendak diolah oleh fungsi.
3. Setelah method telah selesai dieksekusi, dia akan kembali pada method yang memanggilnya.

#### 1. Deklarasi sebuah method

Method terdiri atas dua bagian yakni :

1. Method declaration
2. Method Body

Method dapat digambarkan sebagai sifat (*behavior*) dari suatu class. Untuk mendefinisikan method pada dalam *class* digunakan syntax :

```
[modifiers] return_type method_identifer ([parameter]) {  
    method_code_block;  
}
```

di mana :

- a. **[modifiers]** merepresentasikan kata kunci pada teknologi Java yang memodifikasi cara-cara penggunaan method. Contoh : public, protected, private, static, final.
- b. **return\_type** adalah tipe nilai yang akan dikembalikan oleh method yang akan digunakan pada bagian lain dari program. Return\_type pada method sama dengan tipe data pada variabel. Return\_type dapat merupakan tipe data primitif maupuntipe data referensi.
- c. **method\_identifier** adalah nama method.
- d. **([parameter])**, merepresentasikan sebuah daftar variabel yang nilainya dilewatkan / dimasukkan ke method untuk digunakan oleh method. Bagian ini dapat tidak diisi, dan dapat pula diisi dengan banyak variabel.
- e. **method\_code\_block**, adalah rangkaian pernyataan / statements yang dibawa oleh method.

Contoh :

```
public int Perkalian ( int y;int z )  
{  
    return y * z ;  
}
```

## 2. Modifier Pada Method

Modifier menentukan level pengaksesan sebuah method. Hal ini menentukan apakah sebuah method biasa diakses oleh objek lain, objek anak, objek dalam satu paket atau tidak dapat diakses oleh suatu object sama sekali berikut adalah beberapa jenis level access:

### 1. Public

Atribut ini menunjukan bahwa fungsi/method dapat diakses oleh kelas lain.

### 2. Private

Atribut ini menunjukan bahwa fungsi atau method tidak dapat diakses oleh kelas lain

### 3. Protected

Atribut ini menunjukan bahwa fungsi atau method bisa diakses oleh kelas lain dalam satu paket dan hanya kelas lain yang merupakan subclass nya pada paket yang berbeda.

#### 4. Tanpa modifier

Atribut ini menunjukkan bahwa method dapat diakses oleh kelas lain dalam paket yang sama.

```
public class MyClass {  
  
    private void privateMethod() {  
  
        System.out.println("I am private method");  
  
    }  
  
    void packageMethod() {  
  
        System.out.println("I am package method");  
  
    }  
  
    protected void protectedMethod() {  
  
        System.out.println("I am protected method");  
  
    }  
  
    public void publicMethod() {  
  
        System.out.println("I am public method");  
  
    }  
}
```

#### 3. Method Tanpa Nilai Balikan

Method ini merupakan method yang tidak mengembalikan nilai. Maka dari itu, kita harus mengganti tipe kembalian dengan kata kunci void. Berikut ini kode program yang dimaksud:

```
class persegi {  
  
    double sisi;  
  
    // mendefinisikan method void (tidak mengembalikan nilai)  
  
    void cetakluas() {  
  
        System.out.println("Luas Persegi = " + (sisi * sisi));  
  
    }  
  
}
```

```

class persegiApp {

    public static void main(String[] args) {

        // instansiasi objek

        persegi p1 = new persegi();

        persegi p2 = new persegi();

        // mengisi data untuk masing-masing objek

        p1.sisi = 10;

        p2.sisi = 5;

        // memanggil method cetakLuas() untuk masing-masing objek

        p1.cetakluas();

        p2.cetakluas();

    }

}

Output :

Luas Persegi = 100.0

Luas Persegi = 25.0

```

#### 4. Method Dengan Nilai Balikan

Di sini, kita akan memodifikasi program sebelumnya dengan mengganti method cetakLuas() menjadi method hitungLuas () yang akan mengembalikan nilai dengan tipe double. Berikut ini kode program yang dimaksud:

```

class Persegi {

    double sisi;

    // mendefinisikan method yang mengembalikan nilai

    double hitungluas() {

        return (sisi*sisi);
    }
}

```

```

    }

    }

    class PersegiApp {

        public static void main(String[] args) {

            // instansiasi objek

            Persegi p1 = new Persegi();

            Persegi p2 = new Persegi();

            // mengisi data untuk masing-masing objek

            p1.sisi = 10;

            p2.sisi = 5;

            // memanggil method hitungLuas() untuk masing-masing objek

            System.out.println("Luas Persegi = " + p1.hitungluas());

            System.out.println("Luas Persegi = " + p2.hitungluas());

        }

    }

```

Output :

Luas Persegi = 100.0

Luas Persegi = 25.0

## 5. Parameter

Dengan adanya parameter, sebuah method dapat bersifat dinamis dan general. Artinya, method tersebut dapat mengembalikan nilai yang beragam sesuai dengan nilai parameter yang dilewatkan. Terdapat dua istilah yang perlu anda ketahui dalam bekerja dengan method, yaitu parameter dan argumen. Parameter adalah variabel yang didefinisikan pada saat method dibuat, sedangkan argumen adalah nilai yang digunakan pada saat pemanggilan method. Dalam referensi lain, ada juga yang menyebut parameter sebagai parameter formal dan argumen sebagai parameter aktual.

Perhatikan kembali definisi method berikut :

```
int luasPersegiPanjang(int panjang, int lebar) {  
  
    return panjang * lebar;  
  
}
```

Disini, variabel panjang dan lebar disebut parameter.

```
Luas1 = luasPersegiPanjang(10, 5);
```

Adapun pada statemen diatas, nilai 10 dan 5 disebut argumen.

Sekarang, kita akan mengimplementasikan konsep di atas ke dalam kelas Persegi. Di sini data sisi akan kita isikan melalui sebuah method. Berikut ini kode program yang dimaksud:

```
class Persegi {  
  
    double sisi;  
  
    // mendefinisikan method mengembalikan nilai  
  
    double hitungluas() {  
  
        return (sisi*sisi);  
  
    }  
  
    void isiSisi(int s) {  
  
        sisi = s;  
  
    }  
  
}  
  
class PersegiApp {  
  
    public static void main(String[] args) {  
  
        // instansiasi objek  
  
        Persegi p1 = new Persegi();  
  
        Persegi p2 = new Persegi();  
  
  
        // mengisi data untuk masing-masing objek
```

```

p1.isiSisi(10);

p2.isiSisi(5);

// memanggil method hitungLuas() untuk masing-masing objek

System.out.println("Luas Persegi = " + p1.hitungluas());

System.out.println("Luas Persegi = " + p2.hitungluas());

}

}

```

Output :

Luas Persegi = 100.0

Luas Persegi = 25.0

Bagaimana jika bagian lain dari program ingin tahu juga nilai luas itu tetapi tidak ingin menampilkannya (mencetaknya). Apabila terdapat suatu fungsi yang tidak menghasilkan suatu nilai apapun maka bagian return type ini diganti dengan void . Contoh penggunaan return:

```

class Persegi {

    double sisi;

    // mendefinisikan method mengembalikan nilai

    double hitungluas(int sisi) {

        return (sisi*sisi);

    }

}

class PersegiApp {

    public static void main(String[] args) {

        // instansiasi objek

        Persegi p1 = new Persegi();

        Scanner input = new Scanner(System.in);

        //fungsi untuk menginputkan suatu nilai

```

```
System.out.print("Masukan sisi persegi : ");

int a = input.nextInt();

// memanggil method hitungLuas()

System.out.println("Luas Persegi = " + p1.hitungluas(a));

}

}
```

Output :

Masukan sisi persegi : 10

Luas Persegi = 100.0

Soal Latihan:

1. Buatlah kode program dimana di dalamnya terdapat method dengan nilai balikan untuk penjumlahan, pengurangan, pembagian, dan pengkalian. Setelah itu lakukan pemanggilan untuk method-method tersebut.
2. Buatlah kode program dimana di dalamnya terdapat method tanpa nilai balikan untuk penjumlahan, pengurangan, pembagian, dan pengkalian. Setelah itu lakukan pemanggilan untuk method-method tersebut.
3. Buatlah program menampilkan daftar member suatu Laundry menggunakan method dengan parameter sebanyak 6 parameter.

Contoh :

Input Data

Nama Depan : Cecep  
Nama Belakang : Surecep  
ID Member : 13011996  
Alamat : Sukabirus  
Nomor Telepon : 0812213458xx  
Tahun bergabung : 2010

Output Data

Cecep Surecep, 13011996 dengan alamat di Sukabirus dan nomor telepon 0812213458xx bergabung menjadi member di Laundry Trimo Resik sejak tahun 2010



## JAVA READ AND WRITE FILE

### **TUJUAN PRAKTIKUM**

1. Praktikan memahami bagaimana penggunaan sebuah file dalam Java.
2. Praktikan memahami bagaimana proses pembacaan maupun penulisan sebuah file dalam program Java.
3. Praktikan memahami bagaimana format penulisan sebuah file sehingga mudah untuk di baca diprogram yang lain.

Basworo Bawiso Muhammad (BBM)	basworo28@gmail.com
Cahya Nofandiyan Putra (CNP)	cahya.nputra@gmail.com
Adnan Baharrudin Fanani (ABF)	adnanbf@gmail.com
Pratiwi Galuh Putri (PGP)	putrig@yahoo.com
Nur Intan Paramanisa (INT)	intan.paramanisa@gmail.com
Nurrida Aini Zuhroh (RID)	nurrida.aini@gmail.com
Anisatun Nafi'ah (ANS)	aniskunis09@gmail.com
I Komang Jaka Aksara Wiguna (JEK)	xeckond@hotmail.co.id
Jan Fandro Siahaan (JFS)	fandrocommando@gmail.com
Nana Ramadhew (RDN)	nanaramadhewi3@gmail.com
Agung Candra Dutiya Purwanta (ANG)	usr.agung@gmail.com
Mochamad Thariq Januar (JNR)	januar.xp10@gmail.com
Slamet Mamat Rachmat (SLR)	slametmrachmat@gmail.com
Reza Harli Saputra (RHS)	reza.harli@gmail.com
Kevin Rohni Goklas Sinaga (KVN)	kevinrgsinaga@gmail.com
Timbul Prawira Gultom (TPG)	timbulprawira@gmail.com