



# DASAR PEMROGRAMAN 1

Disusun oleh  
Mardiyyah Hasnawi

# Topik Bahasan

## *Pemrograman Dasar dalam Bahasa Java*

1. Tipe Data & Variable/Property/Atribut
2. Konstanta dan Casting .
3. Object, Class dan Constructor.
4. Standar I/O.
5. OperatorPercabangan.
6. Perulangan.
7. Fungsi/Prosedur/Method.
8. Array.

# PENDAHULUAN

## Struktur Program Bahasa Java

- Sintak : Deklarasi klas beserta anggotanya

```
_ modifier class Filename {  
    // variables and data types  
    // constructor  
    // methods  
    // inner class  
    // etc...  
}
```

atau

```
[modifier] [class] namaclass {  
    //....  
}
```

- *Contoh :*

```
public class X{  
    public static void main(String [] args) {  
        // Isi  
    }  
}
```

# Aturan Penamaan

- Nama dapat berupa kombinasi huruf, angka, dan underscore
- Nama tidak boleh diawali dengan angka
- Tidak boleh sama keyword
- Pemberian namafile = Nama Kelas
- Ekstensi \*.java

# Keyword di Java

<b>abstract</b>	<b>double</b>	<b>int</b>	<b>super</b>
<b>boolean</b>	<b>else</b>	<b>interface</b>	<b>switch</b>
<b>break</b>	<b>extends</b>	<b>long</b>	<b>synchronized</b>
<b>byte</b>	<b>false</b>	<b>native</b>	<b>this</b>
<b>byvalue</b>	<b>final</b>	<b>new</b>	<b>threadspace</b>
<b>case</b>	<b>finally</b>	<b>null</b>	<b>throw</b>
<b>catch</b>	<b>float</b>	<b>package</b>	<b>transient</b>
<b>char</b>	<b>for</b>	<b>private</b>	<b>true</b>
<b>class</b>	<b>goto</b>	<b>protected</b>	<b>try</b>
<b>const</b>	<b>if</b>	<b>public</b>	<b>void</b>
<b>continue</b>	<b>implements</b>	<b>return</b>	<b>while</b>
<b>default</b>	<b>imports</b>	<b>short</b>	
<b>do</b>	<b>instanceof</b>	<b>static</b>	

# Access Modifier

- public : akses seluruh kelas
- private : dalam kelas saja
- protected dalam kelas pewarisan
- Non-Modifier

# TIPE DATA & VARIABLE



# Tipe Data

## ■ Tipe Data Primitif

Tipe Data	Keterangan
boolean	true atau false
char	Karakter
byte	-128 - 127
short	-32768 - 32767
int	-2147483648 - 2147483647
long	-9223372036854775808 - 9223372036854775807
double	4.9E-324 - 1.7976931348623157E308
float	1.4E-45 - 3.4028235E38

- String adalah objek akan tetapi dapat dipergunakan secara langsung tanpa penciptaan objek



# Tipe Data

## Bilangan bulat (integer)

- byte (8-bit) → -128 s/d 127
- short (16-bit) → -32768 s/d 32767
- char (16-bit) → 0 s/d 65535
- int (32-bit) → -2147483648 s/d  
2147483647
- long (64-bit) → -9223372036854775808  
s/d 9223372036854775807

## Bilangan real

- float (32-bit) → single-precision
- double (64-bit) → double-precision

## Logic

- boolean → true/false

## Literal

- String

# Nilai

## Bilangan bulat (integer)

- Desimal

contoh : 4                      -45                      1230                      -34567

- Oktal

contoh : 0777                      0004                      0345

- Heksadesimal

contoh : 0xFF0x3F4D

## Bilangan real

- Pecahan (untuk tipe float perlu ditambahkan notasi F dan untuk double D, setelah nilainya.

contoh : 0.2345F                      123.453D                      -34.67F

- Eksponen

contoh : 10E05                      1.5e12                      3.45e-5

## Logic

- Boolean

contoh :              true                      false

# Nilai

Karakter → nilai berupa karakter harus diapit dengan tanda petik tunggal (single-quote)

Contoh : 'A' 'g' '.' '@' '5'

String → nilai berbentuk string (lebih dari satu karakter) harus diapit dengan tanda petik ganda (double-quote)

Contoh : "AB"

"Dunia dalam berita"

"345"

# Deklarasi & Inisialisasi Variabel

## Sintak:

**[modifier] tipeData   namaVariabel ;**

**atau**

**[modifier] tipeData   namaVar1, namaVar2, .... ;**

**atau**

**[modifier] tipeData   namaVariabel = nilai;**

**atau**

**[modifier] tipeData   namaVar1 = nilai,...;**

# Contoh

```
byte umur;  
short tahun;  
char nilaiHuruf;  
int tinggi;  
int jarak;  
long berat;  
float ipk;  
double saldo;  
String nama;
```

```
byte umur = 63;  
short tahun = 2008;  
char nilaiHuruf = 'C';  
int tinggi = 175, jarak = 0;  
long berat = 55;  
float ipk = 3.81F;  
double saldo = 0.0;  
String nama = new String();  
String nama = "Ursyalim";
```

# Contoh komunikasi antar 2 Class/Objek

```
/*Namafile : Uji.java
   Hari/tgl : Senin, 19 sep 2017
*/
public class Uji{

    public static void main(String[] args){
        String nama ="Ani";
        int umur = 5;

        System.out.println("Panggil saya : " + nama);
        System.out.println("Saya berumur: " + umur + "tahun");
    }
}
```

# KONSTANTA & CASTING





# Constant/Konstanta di Java

- Deklarasi konstanta di Java menggunakan keyword `final`.

Contoh:

```
final double PI = 3.14;
```

```
final int JUM_MHS = 40;
```

# Type-casting

- Type-casting → mengubah sebuah nilai dengan tipe data tertentu ke dalam tipe data yang lain.

**byte → short, char, int, long, float, double**

**short → int, long, float, double**

**char → int, long, float, double**

**int → long, float, double**

**long → float, double**

**float → double**

- Type-casting dapat juga dilakukan secara eksplisit

Contoh:

```
int x;
```

```
x = (int) (7.0 / 3);
```

ekspresi **7.0 / 3** menghasilkan nilai bertipe real sedangkan **x** bertipe integer. Agar **x** dapat menyimpan hasil perhitungan, maka nilainya harus di-casting terlebih dahulu menjadi bertipe integer, dengan konsekuensi hilangnya tingkat presisi dari nilai tersebut. Pada akhirnya x akan menyimpan nilai 2.

# Tanda komentar

// baris ini merupakan contoh komentar

/\* blok ini, yang dapat terdiri dari beberapa baris, juga merupakan contoh komentar dalam program \*/

CONTOH



# Contoh Program Java

- Transformasi Kasus→ Bahasa Program Java
- Kerangka program :

```
public class HitungLuas{  
    // program utama  
    public static void main (String[] args) {  
        //Langkah 1 : Baca radius  
        //Langkah 2: Hitung Luas  
        //Langkah 3 : Tampilkan Luas  
    }  
}
```

## Program menghitung luas lingkaran

```
public class HitungLuas{  
    // program utama  
    public static void main main (String[] args) {  
        // deklarasi variabel radius & luas  
        double radius;  
        double luas;  
        //Langkah 1 : Baca radius  
        radius = 2;  
        //Langkah 2: Hitung Luas  
        luas = radius*radius*3.14159;  
        // Langkah 3 : Tampilkan Luas  
        System.out.println("Luas : " + luas);  
    }  
}
```

- Penting untuk mengurangi pemakaian memori yang berlebih →  
Gunakan variabel seperlunya
- Program menghitung luas lingkaran

```
public class HitungLuas{  
    // program utama  
    public static void main main (String[] args) {  
        // deklarasi variabel radius  
        double radius;  
        //Langkah 1 : Baca radius  
        radius = 2;  
        //Langkah 2 dan 3 : Hitung Luas & tampilkan  
        System.out.println("Luas : " + radius*radius*3.14159) ;  
    }  
}
```

Alternatif



# Pengujian

- Uji kasus: (*test case*) terhadap program terutama setiap kemungkinan nilai masukan yang ada dsb agar program berjalan sesuai dengan spesifikasinya.
- Contoh :

radius	luas
2/4	0,7853975
-	-
20	1256,636
1/0	Tak terdefinisi

# CLASS & OBJECT



# Class dalam Java

- Sebuah *blueprint* dari object tunggal yang dibuat.
- Class bisa berisi macam-macam variable:
  - ***Local Variable***: variable yang didefinisikan di dalam method, konstruktor, atau dalam blok. Variable dideklarasikan dan diinisialisasi didalam method dan akan dihancurkan ketika method telah dieksekusi.
  - ***Instance Variable***: variable di dalam class akan tetapi di luar method, diinstansiasi ketika class loading. Diakses dari dalam method, konstruktor, atau blok class tertentu.
  - ***Class Variable***: variable di dalam class, di luar method, dan menggunakan kunci static

# Class Member

- Variabel beserta typedata-nya
- Method/fungsi dan fungsi utama
- Constructor
- Inner Class

# Object dalam Java

- Objek mempunyai *state* (kondisi→properti) dan *behavior* (perilaku).
  - Contoh : Seekor kucing memiliki warna, nama sedangkan kebiasaannya atau perilakunya antara lain makan, mengibas ekornya, minum, dll.
- Object software juga memiliki *state* dan *behavior*.
- Object software khususnya *state* disimpan sebagai *field* sedangkan *behavior* disimpan sebagai *method*.

# Object dalam Java- lanjut

- *Objek hasil dari penciptaan Class, ibarat kue adalah objek , maka kelas adalah cetakan kue, dimana kue dibuat menggunakan cetakan tersebut. Sebuah cetakan kue dapat memuat beberapa kue, artinya kelas dapat membuat beberapa kue.*
- *Pengembangan software:*
  - *Method beroperasi pada sebuah state internal object.*
  - *Komunikasi object ke object dilakukan melalui method.*

# Membuat Object

- Menggunakan keyword new
  - Format : ***namaClass instasobjek = new namaClass();***
  - Contoh : `Coba objcoba = new Coba();`
- Pemanggilan anggota class: atribut/method menggunakan ‘.’ (titik)
- Format inisialisasi anggota class/objek:  
`instasobjek.namaAtribut = nilai;` → mengubah nilai atribut  
`instasobjek.namaMethod();` → invokasi method (Fungsi & prosedur)

# Contoh

```
/*Namafile : Coba.java
   Hari/tgl : Senin, 19 sep 2017
*/
public class Coba{
    public static void main(String[] args){
        Coba obj= new Coba(); // obj adalah instans objek
    }
}
```



# Contoh komunikasi antar 2 Class/Objek

```
/*Namafile : Coba.java
   Hari/tgl : Senin, 19 sep 2017
*/
public class Coba{
    String nama;
}
```

```
/*Namafile : Uji.java
   Hari/tgl : Senin, 19 sep 2017
*/
public class Uji{
    String nama ="test1";
    public static void main(String[]
args) {

        Coba obj= new Coba() ;
        obj.nama = "test2";
        System.out.println(obj.nama) ;

    }
}
```

# CONSTRUCTOR



# Constructor /Konstruktor

- Otomatis Dipanggil/Dijalankan Pada Saat Sebuah Class Diinstansi.
- Pertama Kali Dijalankan Pada Saat Sebuah Objek Pertama Kali Diciptakan.
- Jika Dalam Sebuah Class Tidak Terdapat Konstruktor Maka Secara Otomatis Java Akan Membuatkan Sebuah Default Konstruktor.
- Nama Constructor Harus Sama Dengan Nama Class Dan Tidak Boleh Memiliki Tipe Return Value.
- Sama Halnya Dengan Method, Konstruktor Dapat Memiliki Satu Atau Banyak Parameter Maupun Tanpa Parameter.

# Membuat Constructor

## Sintak:

```
[ _modifier] namaConst(param1,param2,...,paramn){  
    //inisialisasi  
}
```

## Keterangan:

*Param1,param2,...paramn* adalah variable yang memiliki tipe data dan digunakan untuk memberikan nilai pada *variable class*

# Jenis konstruktor

1. Konstruktor Default/non parameter
2. Konstruktor berparameter

# Contoh:

// nama kelas manusia= nama konstruktor

```
public class Manusia {
```

```
    String nama;
```

```
    public Manusia(){
```

```
        nama = "Sisi"
```

```
    }
```

```
    //atau
```

```
    //public Manusia(String param1){ //berparameter
```

```
    // nama = param1
```

```
    //}
```

```
}
```

Class name= Constructor name

//class variable

//default atau tanpa berparameter

// inisialisasi variable nama = sisi

//berparameter

// variabel nama diisi dengan nilai param


# Overloading Konstruktor

- Mekanisme dimana dapat membuat Konstruktor lebih dari satu dalam satu Class, tapi dengan ketentuan setiap Konstruktor memiliki Parameter yang berbeda, bisa berbeda jumlah Parameternya ataupun berbeda Type Data parameternya.

- contoh

Overloading

```
public class Orang {  
    String nama;  
  
    // Konstruktor 1  
    public Orang() {  
        nama = "sisi";  
    }  
    // Konstruktor 2  
    public Orang(String nama) {  
        this.nama = nama ;  
    }  
}
```



```
public class OrangAksi {  
  
    public static void main(String[] args){  
  
        Orang orang1 = new Orang() ;  
        Orang orang2 = new Orang("Idriez  
        Gienanjair") ;  
        System.out.println(orang1.nama);  
        System.out.println(orang2.nama);  
  
    }  
}
```

# Tugas 1

1. Buat aplikasi bahasa Java menginisialisasi dan menampilkan nim, nama, jurusan dan fakultas anda!
2. Buat Program menggunakan bahasa java untuk Konversi Waktu (Jam:Menit:Detik) dari masukan detik! (lihat kerangka program pada slide berikutnya)



# Kasus 2

- Menampilkan Waktu dalam format jam:menit:detik.
- Contoh : Masukkan total detik: 1203183086 (satuan detik)
- Spesifikasi :
  1. *mendapatkan total detik melalui masukan keyboard (misalnya. 1203183086 )*
  2. *mendapatkan detik saat ini dari totalDetik % 60 (misal 1203183086 detik % 60 = 26)*
  3. *mendapatkan detik ssat ini dari totalDetik dengan membagi totalDetik dengan 60 (misal 1203183086 detik /60 = 20053051 menit)*

# lanjutan

4. mendapatkan menit saat ini dari totalMenit % 60  
(misalnya  $20053051 \text{ menit} \% 60 = 31 \text{ menit saat ini}$ )
5. mendapatkan total jam totalJam dengan membagi totalMenit dengan 60 (misal  $20053051 \text{ menit} / 60 = 334217 \text{ jam}$ )
6. mendapatkan jam saat ini dari totalJam % 24 (misal  $334217 \text{ jam} \% 24 = 17 \text{ jam saa ini}$ )

# Kerangka Program soal no.2

1. Masukkan total detik
2. Hitung detikSekarang = totalDetik % 60
3. Hitung totalMenit = totalDetik / 60
4. Hitung menitSekarang = totalMenit % 60
5. Hitung totalJam = totalMenit / 60
6. Hitung jamSekarang = totalJam % 24
7. Tampil waktu (Jam:Menit:Detik)

STANDAR I/O



# Membaca Masukan dari Konsol

## 1. Menggunakan kelas **Scanner**

- *Import kelas Scanner ke dalam File Java*
- *Deklarasi instans/objek representasi dari kelas **Scanner**:*

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

- *System.out → keluaran standar → ke layar monitor*
- *System.in → masukan standar → dari keyborad*

➤ Contoh : `import java.util.Scanner;`

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

```
char symbol = input.next().charAt(0);
```

```
// memasukkan data bertipe karakter
```

# Scanner Class

Metode	Keterangan
<code>nextByte()</code>	Membaca angka bertipe byte
<code>nextShort()</code>	Membaca angka bertipe short
<code>nextInt()</code>	Membaca angka bertipe int
<code>nextLong()</code>	Membaca angka bertipe long
<code>nextFloat()</code>	Membaca angka bertipe float
<code>nextDouble()</code>	Membaca angka bertipe double
<code>next()</code>	Membaca suatu string yang berakhir dengan karakter spasi
<code>nextLine()</code>	Membaca sebaris teks (suatu string yang berakhir dengan ENTER)

# Contoh : Scanner Class

```
import java.util.Scanner;

public class HitungLuas{
    // program utama
    public static void main main (String[] args) {
        // objek//instans
        Scanner masukan = new Scanner(System.in);
        //deklarasi var radius & masukan dari kboard
        //Langkah 1 : Baca radius
        double radius = masukan.nextDouble();
        //Langkah 2 dan 3 : Hitung Luas & tampilkan
        System.out.println("Luas : " +
            radius*radius*3.14159);
    }
}
```

# Membaca Masukan berbasis GUI

2. Menggunakan metode **`JOptionPane.showInputDialog("")`** ;
  - *Import kelas JOptionPane : `import javax.swing.JOptionPane;`*
  - *menggunakan statement **`JOptionPane.showInputDialog(x)`** ; dimana x adalah suatu string untuk pesan*
  - *menggunakan statement **`String intString = JOptionPane.showInputDialog (null, x,y, JOptionPane.QUESTION_MESSAGE)`** ;*
    - *dimana x adalah suatu string untuk pesan dan y adalah suatu string untuk judul kotak dialog*



# lanjutan

2. Menggunakan metode **JOptionPane.showInputDialog("")** ;
  - Konversi String menjadi Angka menggunakan metode **parseInt** dalam kelas **Integer**, contoh :

```
int intAngka = Integer.parseInt(intString);
```
  - metode **parseDouble** dalam kelas **Double** contoh :

```
double doubleAngka = Double.parseDouble(intString);
```

# Contoh : JOptionPane Class

```
import javax.swing.JOptionPane;

public class HitungLuas{
    // program utama
    public static void main main (String[] args) {
        //Langkah 1 : Baca radius
        // masukan radius menggunakan kotak dialog
        String inputRadius=
            JOptionPane.showInputDialog("Radius : ");
        //konversi var inputRadius tipe string ->radius tipe double
        double radius = Double.parseDouble(inputRadius);
        //Langkah 2 dan 3 : Hitung Luas & //tampilkan
        System.out.println("Luas : " +
            radius*radius*3.14159) ;
        }//end fungsi main
    }//end class
```

# Membaca masukan Menggunakan InputStream

## 3. Menggunakan metode `readLine()` ;

### ■ Cara untuk memanggil metode:

1. Import paket `java.io.*`; , **Gunakan Class** `BufferedReader` dan `InputStreamReader`
2. Membuat objek dari kelas *BufferedReader* dan memanggil metode `readLine()`;

*BufferedReader input = new BufferedReader (new InputStreamReader(System.in));*

3. `System.in` → masukan standar → dari keyboard

### □ Contoh :

```
import java.io.*;
BufferedReader input = new BufferedReader (new InputStreamReader(System.in));
String statement = input.readLine();
```

# Contoh : BufferedReader Class

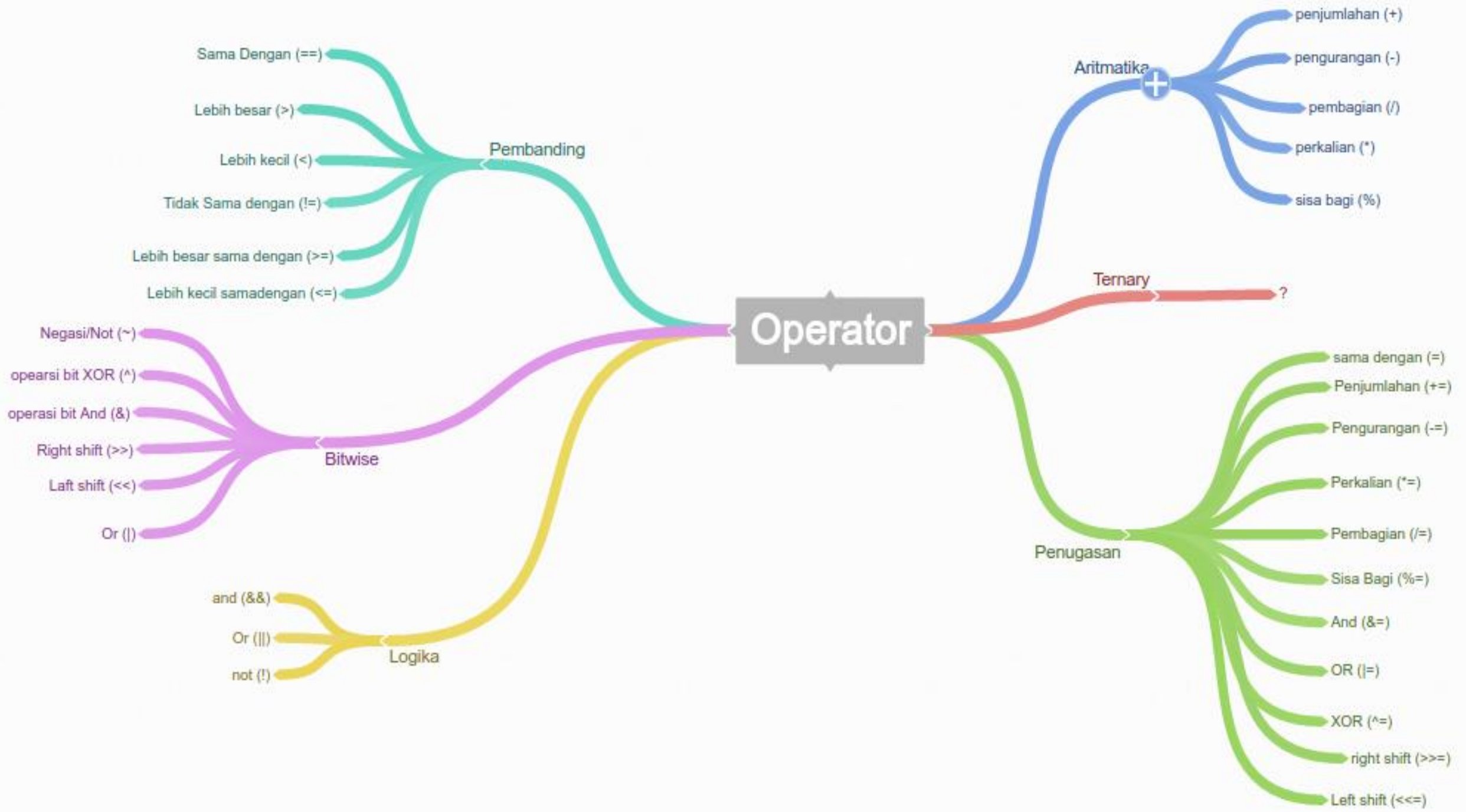
```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class BacaString {
    public static void main(String[] args) throws IOException {
        /* Kamus */
        String str;
        BufferedReader datAIn = new BufferedReader(new
                                                    InputStreamReader(System.in));

        /* Program */
        System.out.print ("\nBaca string dan Integer: \n");
        System.out.print("masukkan sebuah string: ");
        str= datAIn.readLine();
        System.out.print ("String yang dibaca : "+ str);
    }
}
```

# OPERATOR





# Macam-macam Operator

- Arithmetic
- Assignment
- Bitwise
- Conditional
- Equality
- Logical
- Relational

# Sifat Operator

Operator memiliki beberapa sifat:

- Unary → operator dengan sifat ini hanya dapat melibatkan sebuah operand

**-29**

- Binary → operator ini melibatkan dua buah operand

**3 + 8**

- Ternary → operator ini melibatkan tiga buah operand

**(a > b) ? a : b**



# Operator Arithmetic

## ■ Unary

+ (*tanda positif*)                      +34 atau 34

- (*tanda negatif*)                      -67

### Operator increment & decrement

++*ekspresi* (*pre-increment*)

*ekspresi*++ (*post-increment*)

*int a = 5;*

*a++; sama saja dengan a = a + 1;*

--*ekspresi* (*pre-decrement*)

*ekspresi*-- (*post-decrement*)

*Bila letak operator di depan ekspresi, maka operasi inc/dec akan dilakukan lebih dulu, barulah kemudian ekspresi akan dievaluasi. Begitu pula sebaliknya.*

# Operator Arithmetic

## ■ Binary

+ (penjumlahan)

– (pengurangan)

\* (perkalian)

/ (pembagian)

% (modulus)

$5 / 2$  menghasilkan nilai 2, sedangkan

$5 \% 2$  menghasilkan nilai 1, yaitu sisa hasil bagi dari  $5 / 2$ .

# Operator Assignment (binary)

=      \*=      /=      %=      +=      -=  
<<=      >>=      &=      ^=      |=

Contoh:

A = 23

A \*= 5 sama saja dengan A = A \* 5

# Operator Bitwise

Operand-nya harus bertipe bilangan bulat.

## Unary

~ bitwise complement, membalik (invert) nilai setiap bit

## Binary

& bitwise AND, membandingkan 2 bit dan menghasilkan nilai 1 hanya jika kedua bit bernilai 1

| bitwise inclusive OR, membandingkan 2 bit dan menghasilkan nilai 0 hanya jika kedua bit bernilai 0

^ bitwise exclusive OR, membandingkan 2 bit dan menghasilkan nilai 1 hanya jika kedua bit berbeda nilainya

>> bitwise shift right, memindahkan susunan bit ke kanan dan membuang bit paling kanan serta memberikan nilai 0 pada bit paling kiri

<< bitwise shift left; memindahkan susunan bit ke kiri dan membuang bit paling kiri serta memberikan nilai 0 pada bit paling kanan

# Operator Conditional (ternary)

Operator → `?:`

## Contoh:

```
int a = 3;
```

```
String hasil = a > 5 ? "baik" : "buruk";
```

```
System.out.print(hasil);
```

## Output:

buruk

# Operator Equality (binary)

`==` (perbandingan sama dengan)

`!=` (perbandingan tidak sama dengan)

Contoh:

`76 == 54`

`98 != 43`

`"kata" == "KaTa"`

ekspresi ini tidak memberikan hasil yang benar.  
Untuk membandingkan dua string dapat menggunakan function `compareTo()` atau `compareToIgnoreCase()`.

# Operator Logical

&& logical AND, memberikan nilai true jika kedua operand bernilai true.

| | logical OR, memberikan nilai false jika kedua operand bernilai false.

! logical negation (unary), memberikan nilai kebalikan (negasi) dari operand.

# Operator Relational (binary)

> (lebih dari)

< (kurang dari)

>= (lebih dari atau sama dengan)

<= (kurang dari atau sama dengan)

Contoh:

67 < 98

'a' >= 'B';

- Nilai selain true akan dikembalikan bila ekspresinya bernilai benar, dan false bila ekspresinya salah



# Operator

Precedence	Operator	Associativity
1	( ), [ ]	non-associative
2	<b>new</b>	non-associative
3	.	left-associative
4	<b>++</b> , <b>--</b>	non-associative
5	<b>- (unary)</b> , <b>+</b> (unary), <b>!</b> , <b>~</b> , <b>++</b> , <b>--</b> , ( <i>type</i> )	right-associative
6	<b>*</b> , <b>/</b> , <b>%</b>	left-associative
7	<b>+</b> , <b>-</b>	left-associative

8	<<, >>, >>>	left-associative
9	<, >, <=, >=, instanceof	non-associative
10	==, !=	left-associative
11	&	left-associative
12	^	left-associative
13		left-associative
14	&&	left-associative
15		left-associative
16	?:	right-associative
17	=, *=, /=, %=, -=, <<=, >>=, >>>=, &=, ^=,  =	right-associative

# Contoh

```
public class belajarOperator{  
  
    public static void main(String[]args){  
  
        int a = 234; //nilai variabel a = 234  
        int b = 3; // nilai variabel b = 3  
        int c = a +b; //234 + 3  
  
        System.out.println(c);  
  
    }  
}
```

PACKAGE

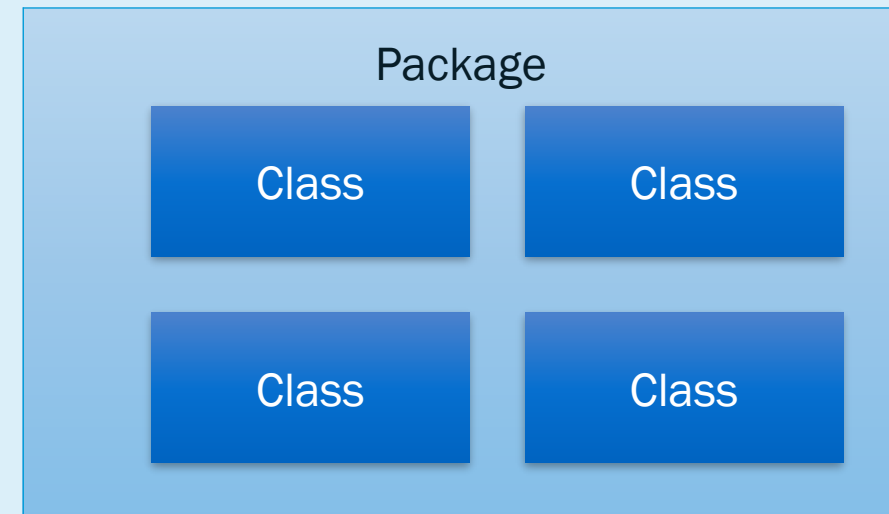


# Sistem package di Java

- JDK (Java Development Kit) bukan hanya berisi compiler & interpreter Java saja, namun juga berbagai macam package.
- Paket (package) dapat di-analogikan seperti unit pada compiler Pascal atau file header pada compiler C/C++.
- Di dalam package terdapat beberapa class
- Pada compiler Pascal dikenal adanya unit. Unit berisi kumpulan perintah dalam bentuk procedure dan function.

Contoh di Pascal:

```
Uses CRT, DOS;  
Begin  
  clrscr;  
  .....  
  getdate (...);  
End.
```



- Pada compiler C/C++ dikenal adanya file header. Contoh di C++:

```
#include <iostream.h>  
#include <conio.h>  
int main() {  
    clrscr();  
    cout << "halo teman";  
    return 0;  
}
```

- Setiap package berisi perintah-perintah tambahan di Java dalam bentuk class dan objek yang memiliki property maupun function (method) yang dapat digunakan untuk proses tertentu.
- Beberapa package yang disediakan oleh Java adalah sebagai berikut:
  - *package java.lang*
  - *package java.io*
  - *package java.swing*
  - *package java.applet*
  - *package java.awt*
  - *package java.net*
  - *dan lain-lain*

# Deklarasi dan Penggunaan Paket/Package

- Di Java, untuk menyertakan sebuah package dapat menggunakan kata kunci “import”.
- Berikut adalah syntax penulisannya :
- Sintak : Deklarasi

```
package namaPackage;
```

atau

```
package namaPackage.subPackage1.subPackage2.subPackagen;
```

## Penggunaan

```
import namaPackage.*; atau
```

```
import namaPackage>NamaClass;
```

atau

```
import namaPackage.namasubPackage>NamaClass;
```



- Bentuk pertama digunakan untuk meng-import hanya satu buah class saja. Contoh:

```
import java.io.DataInputStream;  
import java.applet.Applet;
```

- Sedangkan bentuk kedua digunakan untuk meng-import seluruh class pada package tersebut. Contoh:

```
import java.io.*;  
import java.applet.*;
```

***Khusus untuk package java.lang, kita tidak perlu untuk melakukan import terhadap package ini karena secara otomatis package ini akan di-import oleh compiler Java.***

# Contoh

```
package kendaraan;

public class Sepeda{
    public Sepeda () {
        System.out.println("Ini kelas Sepeda");
    }
}
```

```
package kendaraan;

public class Mobil{
    public Mobil() {
        System.out.println("Ini kelas Mobil");
    }
}
```

# lanjut

```
import kendaraan.Sepeda;  
import kendaraan.Mobil;  
  
public class AlatTransportasi{  
    public static void main(String[] args){  
        Sepeda sepedaku = new Sepeda();  
        Mobil mobilku = new Mobil();  
    }  
}
```

# Tugas 2

## Tugas Mandiri : Praktek Program Java

### Petunjuk tugas

- Tulis Output setiap program ke dalam dokumen dalam bentuk file \*.pdf
- Berikan penjelasan setiap program terutama tujuan program, keyword yang digunakan dari var/tipedata/kelas/method, dll.
- Bagi yang mengerjakan tugas masing-masing, **sertakan NIM, Nama, Hari/Tanggal dan waktu (sebagai komentar baris pertama kode program)** pada setiap source code
- Tugas dikumpul dalam bentuk arsip .rar atau .zip dengan ketentuan:  
*Nama arsip : Tugas2\_NIM1\_NIM2\_NIM3\_NMKelompok.zip atau .rar*
  1. Folder name : **Tugas2\_SourceCode** (berisi kode program)
  2. Filename : **Tugas2\_NIM1\_NIM2\_NIM.pdf** (berisi penjelasan dan output program)