

03. Variabel, Tipe Data, Operator dan Input-Output

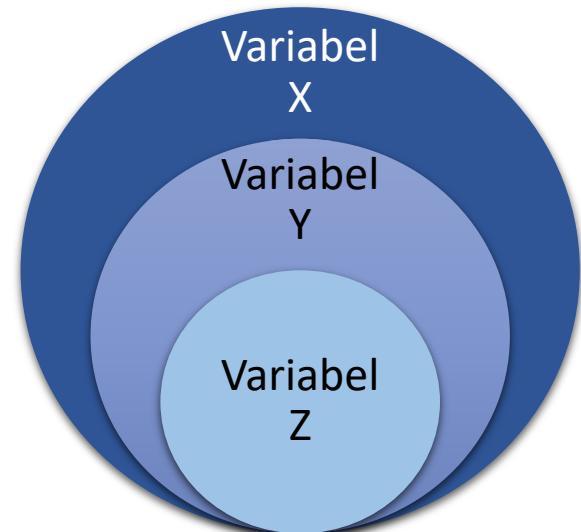
Inggrid Y. R. P. S.S.T., M.Tr.T.

Variabel

- Variable digunakan dalam bahasa pemrograman untuk menyimpan nilai sementara dimana untuk digunakan kembali nantinya.
- Variabel memiliki tipe data dan nama.
- tipe data mengindikasikan tipe dari nilai pada variabel tersebut.

Jenis Variabel

- Variable **lokal** adalah variable yang hanya bisa dikenali pada sub program
- Variabel **global** adalah variable yang dapat dikenali pada keseluruhan program



Aturan Penamaan Variabel

- Nama variable **tidak boleh** menggunakan kata kunci Java
- Nama variable **boleh** menggunakan huruf, angka(0-9), garis bawah(_), dan symbol dolar(\$), namun sebaiknya penggunaan
 - ✓ Nama variable menggunakan diawali huruf kecil
 - ✓ Apabila nama variable lebih dari satu kata maka kata yang setelahnya diawali huruf besar.

❖ Bentuk:

<tipe data> <nama> [=nilai awal]

nilai dalam tanda [] bersifat optional.

contoh:

```
int contVariabel;  
int contohVar = 34;
```



Tipe Data

- Tipe data adalah jenis data yang ingin kita simpan di variabel.
- Tipe data dapat dikategorikan menjadi dua kelompok, yaitu
 1. *tipe data Primitif*
 2. *tipe data Referensi.*

Tipe Data Primitif

Jenis Data	Deskripsi	Ukuran	Minimum	Maksimum
boolean	true / false	1-bit		
char	Karakter Unicode	16-bit		
byte	Bilangan bulat	8-bit	-127	128
short	Bilangan bulat	16-bit	-32768	32767
int	Bilangan bulat	32-bit	-2147483648	2147483647
long	Bilangan bulat	64-bit	- 922337203685477580 75808	922337203685477580 7
float	Bilangan riil	32-bit	1.4012984643248 1707e-45	3.40282346638528860 e+38
double	Bilangan riil	64-bit	4.9406564584124 6544e-324	1.79769313486231570 e+308

Deklarasi

-----Deklarasi-----

```
int nilai;  
double angka;  
float a, b, c;
```

-----Pemberian nilai-----

```
int nilai=75;  
double angka=2.5;
```

Mencetak Variabel

```
System.out.println(nilai);
```

```
System.out.println(a);
```

-----atau-----

```
System.out.println("Nilai anda adalah" +nilai);
```

```
System.out.println("angka adalah" +a);
```

Casting Tipe Data

- **Casting** → ketika kita ingin memberikan nilai dari tipe data primitive ke tipe data primitive yang lain
- Widening casting (otomatis) – mengubah tipe data dari yang ukurannya **lebih kecil** ke tipe data yang **lebih besar**
byte -> short -> char -> int -> long -> float -> double



*Ilustrasi
widening
casting*

Casting Tipe Data

- Narrowing casting (manual) – mengubah tipe data dari yang ukurannya lebih besar ke tipe data yang lebih kecil

double -> float -> long -> int -> char -> short -> byte



Ilustrasi narrowing casting

Contoh Cascading Tipe Data

- Widening casting(otomatis)

```
byte umur = 9;
double myDouble = umur;
System.out.println(umur);      // Outputs 9
System.out.println(myDouble);   // Outputs 9.0
```

- Narrowing casting(manual)

```
double ipk = 3.78;
int myInt = (int) ipk;
System.out.println(ipk);    // Outputs 3.78
System.out.println(myInt);   // Outputs 3
```

Tipe Data Referensi

- Tipe data non-primitive dibuat berdasarkan kebutuhan programmer.
- Nilai bawaan non-primitive adalah null
- Pendeklarasian tipe data ini hampir sama dengan deklarasi pada tipe data primitif.
- Tipe data non-primitive diawali dengan huruf besar

Tipe Data Referensi

- Ciri khas tipe data referensi adalah kemampuannya menampung banyak nilai.
- Pada tipe data primitif , nilai yang bisa ditampung Cuma 1 saja. Perhatikan contoh berikut ini:

A) ***Tipe Primitif :***

- int x = 9; (ada 1 nilai saja, yaitu angka 9)
- char hurufku = "h"; (ada 1 nilai saja, yaitu huruf h)

B) ***Tipe Referensi :***

- String tulisan = "Aku Belajar Java"; (ada 16 nilai, termasuk spasi)
- int[] daftar = { 1, 4, 9, 16, 25, 36, 49 }; (ada 7 nilai bertipe integer)

Operator

- Operator → **simbol** yang biasa digunakan dalam menulis suatu pernyataan (*statement*) dalam bahasa pemrograman apapun.
- Operator akan melakukan suatu operasi terhadap operand sesuai dengan fungsinya.
- Contoh operasi antara lain penjumlahan, pengurangan, pembagian dan sebagainya.

3 + 8 * 4
3 8 4 adalah **operand**
+ * adalah **Operator**

Jenis Operator

1. Operator Aritmatika
2. Operator Increment dan Decrement
3. Operator Assignment
4. Operator Relasi
5. Operator Logika
6. Operator Bitwise

1. Operator Aritmatika

- Arithmatic operator (operator aritmatika) → operator yang berfungsi untuk operasi aritmatika.

Arithmatic Operator	Description
+	plus
-	minus
*	point
/	divide
%	modulus

```
public class operatoraritmatika {
    public static void main(String[] args) {
        int a = 20;
        int b = 10;
        System.out.println("Arithmatic Operator");
        System.out.println("bilangan pertama : "+a);
        System.out.println("bilangan kedua: "+b);
        System.out.println(" a + b = " +(a + b));
        System.out.println(" a -b = " -(a -b));
        System.out.println(" a / b = " +(a / b));
        System.out.println(" a * b = " +(a * b));
        System.out.println(" a % b = " +(a % b));
    }
}
```

ariabeltipedataoperator.operatoraritmatika > main >

it - variabeltipedataoperator (run) ✘

```
run:
Arithmatic Operator
bilangan pertama : 20
bilangan kedua: 10
a + b = 30
a -b = 10
a / b = 2
a * b = 200
a % b = 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Operator Increment & Decrement

- Operator Increment dan Decrement → untuk menaikan / menurunkan nilai integer (bilangan bulat) sebanyak satu satuan, dan hanya dapat digunakan pada variabel.

Operator	Use	Description
++	a++	Menaikan/menambah 1 nilai setelah operasi dilakukan
	++a	Menaikan/menambah 1 nilai sebelum operasi dilakukan
--	a--	Penurunan/mengurangi 1 nilai setelah operasi dilakukan
	--a	Penurunan/mengurangi 1 nilai sebelum operasi dilakukan

```
/*
public class OperatorIncrementdanDecrement {
    public static void main(String[] args) {
        int i = 1;
        //increment
        System.out.println("i : " + i);
        System.out.println("++i : " + ++i);
        System.out.println("i++ : " + i++);
        //decrement
        System.out.println("--i : " + --i);
        System.out.println("i--: " + i--);
        System.out.println("i : " + i);
    }
}
```

variabiltipedataoperator.OperatorIncrementdanDecrement > main >

out - variabiltipedataoperator (run) ❁

```
run:
i : 1
++i : 2
i++ : 2
--i : 2
i--: 2
i : 1
```

3. Operator Assignment

- Operator assignment dalam Java → untuk memberikan sebuah nilai ke sebuah variabel.
- Operator assignment hanya berupa '=',

Operator	Penggunaan	Ekuivalen Dengan
<code>+=</code>	<code>Op1 += Op2</code>	<code>Op1 = Op1 + Op2</code>
<code>-=</code>	<code>Op1 -= Op2</code>	<code>Op1 = Op1 - Op2</code>
<code>*=</code>	<code>Op1 *= Op2</code>	<code>Op1 = Op1 * Op2</code>
<code>/=</code>	<code>Op1 /= Op2</code>	<code>Op1 = Op1 / Op2</code>
<code>%=</code>	<code>Op1 %= Op2</code>	<code>Op1 = Op1 % Op2</code>
<code>&=</code>	<code>Op1 &= Op2</code>	<code>Op1 = Op1 & Op2</code>
<code>!=</code>	<code>Op1 != Op2</code>	<code>Op1 = Op1 ! Op2</code>
<code>^=</code>	<code>Op1 ^= Op2</code>	<code>Op1 = Op1 ^ Op2</code>
<code><<=</code>	<code>Op1 <<= Op2</code>	<code>Op1 = Op1 << Op2</code>
<code>>>=</code>	<code>Op1 >>= Op2</code>	<code>Op1 = Op1 >> Op2</code>
<code>>>>=</code>	<code>Op1 >>>= Op2</code>	<code>Op1 = Op1 >>> Op2</code>

3. Operator Assignment

- $a = a + 5$; bisa dipersingkat menjadi `a += 5;`
- $b = b - 5$; bisa dipersingkat menjadi `b -= 5;`
- $c = c * 5$; bisa dipersingkat menjadi `c *= 5;`
- $d = d / 5$; bisa dipersingkat menjadi `d /= 5;`
- $e = e \% 5$; bisa dipersingkat menjadi `e %= 5;`

```
public class operatorassignment2 {  
    public static void main(String[] args) {  
        int a = 10;  
        // Demo operator assignment  
        a += 5;  
        System.out.println("value a [10] += 5 = " + a);  
  
        int b = 10;  
        b -= 5;  
        System.out.println("value b [10] -= 5 = " + b);  
  
        int c = 10;  
        c *= 5;  
        System.out.println("value c [10] *= 5 = " + c);  
  
        int d = 10;  
        d /= 5;  
        System.out.println("value d [10] /= 5 = " + d);  
  
        int e = 10;  
        e %= 5;  
        System.out.println("value e [10] %= 5 = " + e);  
    }  
}
```

Output - variabiltipedataoperator (run) ✘

	run:
	value a [10] += 5 = 15
	value b [10] -= 5 = 5
	value c [10] *= 5 = 50
	value d [10] /= 5 = 2
	value e [10] %= 5 = 0
	BUILD SUCCESSFUL (total time: 0 seconds)

4. Operator Relasi

- Operator relasi dalam Java → untuk menghasilkan nilai boolean yang sering digunakan untuk mengatur alur jalannya sebuah program.

Operator	Penggunaan	Deskripsi
>	$Op1 > Op2$	Menghasilkan true jika Op1 lebih besar dari Op2
<	$Op1 < Op2$	Menghasilkan true jika Op1 lebih kecil dari Op2
\geq	$Op1 \geq Op2$	Menghasilkan true jika Op1 lebih besar atau sama dengan Op2
\leq	$Op1 \leq Op2$	Menghasilkan true jika Op1 lebih kecil atau sama dengan Op2
\equiv	$Op1 \equiv Op2$	Menghasilkan true jika Op1 sama dengan Op2
\neq	$Op1 \neq Op2$	Menghasilkan true jika Op1 tidak sama dengan Op2

```
public class operatorrelasi {  
    public static void main(String[] args) {  
        int x,y,z;  
        x = 100;  
        y = 99;  
        z = 99;  
        System.out.println("Nilai x = "+x);  
        System.out.println("Nilai y = "+y);  
        System.out.println("Nilai z = "+z);  
        // operator sama dengan  
        if(y == z ){  
            System.out.println("y sama dengan z");  
        }else {  
            System.out.println("y tidak sama dengan z");  
        }  
        // operator tidak sama dengan  
        if(x != y ){  
            System.out.println("x tidak sama dengan y");  
        }else {  
            System.out.println("x sama dengan y");  
        }  
        // operator lebih besar dari  
        if(x > y ){  
            System.out.println("x lebih besar dari y");  
        }  
    }  
}
```

menghasilkan

```
Nilai x = 100  
Nilai y = 99  
Nilai z = 99  
y sama dengan z  
x tidak sama dengan y  
x lebih besar dari y  
y lebih kecil dari x  
x lebih besar dari atau sama dengan y  
y lebih kecil dari atau sama dengan x
```

5. Operator Logika

- Operator ini → ekspresi logika yang menghasilkan nilai boolean.
- Operator-operator yang digunakan adalah AND (`&&`), OR (`||`) dan NOT (`!`).

Operator	Deskripsi	Contoh
<code>&&</code>	and	$x=6$ $y=3$ $(x < 10 \&\& y > 1)$ hasil true
<code> </code>	or	$x=6$ $y=3$ $(x==5 \mid\mid y==5)$ hasil false
<code>!</code>	not	$x=6$ $y=3$ $!(x==y)$ hasil true

```
public class operatorlogika {  
    public static void main(String[] args) {  
        boolean _true = true;  
        boolean _false = false;  
  
        System.out.println("Relation with OR (||)");  
        System.out.println("_true || _true : " +(_true||_true));  
        System.out.println("_true || _false : " +(_true||_false));  
        System.out.println("_false || _true : " +(_false||_true));  
        System.out.println("_false || _false : " +(_false||_false));  
  
        System.out.println("Relation with AND (&&)");  
        System.out.println("_true && _true : " +(_true&&_true));  
        System.out.println("_true && _false : " +(_true&&_false));  
        System.out.println("_false && _true : " +(_false&&_true));  
        System.out.println("_false && _false : " +(_false&&_false));  
        System.out.println("Relation with NOT (!)");  
        System.out.println("inverse of (NOT) _true is: " +!_true);  
        System.out.println("inverse of (NOT) _false is: " + !_false);  
    }  
}
```

```
run:  
Relation with OR (||)  
_true || _true : true  
_true || _false : true  
_false || _true : true  
_false || _false : false  
Relation with AND (&&)  
_true && _true : true  
_true && _false : false  
_false && _true : false  
_false && _false : false  
Relation with NOT (!)  
inverse of (NOT) _true is: false  
inverse of (NOT) _false is: true  
BUILD SUCCESSFUL (total time: 1 second)  
|
```

6. Operator Bitwise

- Operator ini digunakan untuk melakukan manipulasi bit dari sebuah bilangan
- **Bitwise OR (|)**
- Hasil bit bernilai 1 ketika salah satu bit-bit bernilai 1, selain itu bernilai 0. Contoh:

```
int a = 5; //0101
int b = 7; //0111
System.out.println(a|b); //output 7
//0101
//0111
//_____
//0111 -> 7
```

6. Operator Bitwise

- Bitwise AND(&)
- Hasil bit bernilai 1 ketika semua bit-bit bernilai 1, selain itu bernilai 0.
- Contoh:

```
int a = 5;//0101
int b = 7;//0111
System.out.println(a&b);//output 5
//0101
//0111
//_____
//0101 -> 5
```

6. Operator Bitwise

- Bitwise XOR(^)
- Nilai bit bernilai 1 ketika ada bit bernilai 1 dan 0, selain itu bernilai 0.
- Contoh:

```
int a = 5;//0101
int b = 7;//0111
System.out.println(a^b);//output 2
//0101
//0111
//_____
//0010 -> 2
```

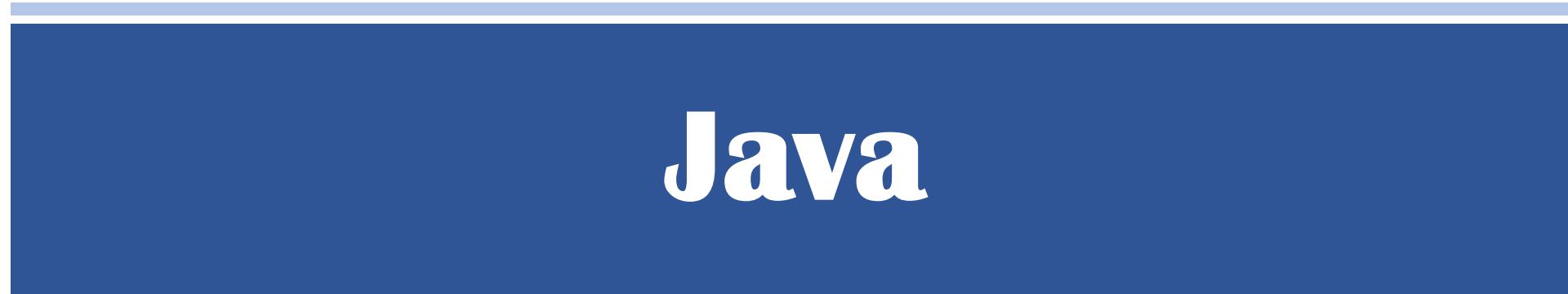
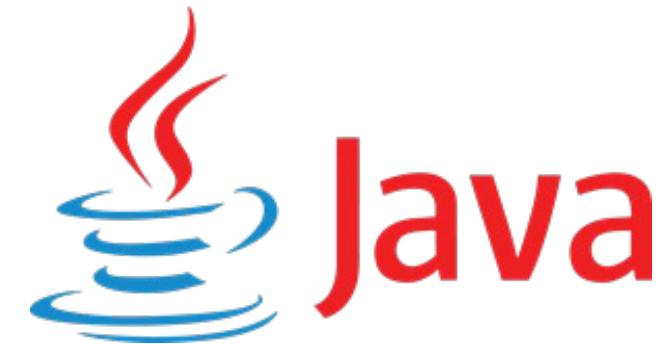
6. Operator Bitwise

- Bitwise Complement(\sim)
- Nilai bit yang berkebalikan, ketika nilai bit bernilai 1 maka menghasilkan 0 sedangkan bernilai 0 menghasilkan 1.

- Contoh:

```
int a = 5; //0101
System.out.println(~a); //output -10
//0101
//_____
//1010 -> 10
```

$$\begin{aligned}\sim n &= -(n+1) \\ \sim(-n) &= (n-1)\end{aligned}$$



Java

Penggunaan Inputan Pada Java

- Untuk membaca input dari keyboard menggunakan library **Scanner** yang di import kedalam program java.
- Caranya : menuliskan perintah **import java.util.Scanner** di baris paling atas dari kode program yang akan dibuat.
- Selanjutnya tuliskan perintah deklarasi scanner berikut ini didalam fungsi main() :

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

Penggunaan Inputan Pada Java

Selanjutnya, tergantung dari jenis input yang akan dimasukkan, berupa bilangan bulat (int), bilangan koma (float/double), atau karakter (String).

1. Jika input berupa bilangan bulat, maka perintahnya adalah: **nextInt();**
2. Jika input berupa bilangan koma, maka perintahnya adalah: **nextFloat();**
3. Jika input berupa teks, maka perintahnya adalah: **nextLine();**

Menampilkan Output Pada Java

1. **System.out.print("Hello world");** Perintah menampilkan kata Hello world ke layar, atau apapun yang dituliskan didalam tanda petik.
2. **System.out.println("Hello world");** Perintah menampilkan kata Hello world ke layar, atau apapun yang kita tuliskan didalam tanda petik, sekaligus memberi perintah ganti baris di akhir kata/kalimat.

Menampilkan Output Pada Java

3. **System.out.println(panjang);** → Perintah menampilkan isi variabel panjang ke layar.
4. **System.out.println("Panjang segi empat: " + panjang);** → Perintah menampilkan kalimat “Panjang segi empat: “ kemudian disambung dengan isi variabel panjang ke layar.

Perhatikan untuk menyambung kalimat dengan isi variabel, digunakan tanda plus (+).

Contoh Kasus 1

- Pak Adi mempunyai sebuah kebun berbentuk persegi Panjang.
- Pak Adi ingin membuatkan pagar kayu untuk mengelilingi kebun tersebut.
- Sebelum membuat program untuk membantu pak adi menghitung keliling kebunnya, maka bantulah pak adi untuk mengidentifikasi **variable** dan **tipe data** beserta **algoritmanya!**

Contoh Kasus 1

1. Menentukan Algoritma

Input: panjang, lebar

Output: keliling

Proses:

1. input panjang, lebar
2. keliling = $2 \times (\text{panjang} + \text{lebar})$
3. Output keliling

2. Mengidentifikasi variable dan jenis tipe data berdasarkan algoritma

Variabel	Tipe data
panjang	int
lebar	int
keliling	int

Contoh Kasus 1

```
1 import java.util.Scanner;
2
3 public class Coba {
4     Run | Debug
5     public static void main (String [] args){
6         Scanner input =new Scanner(System.in);
7
8         int panjang;
9         int lebar;
10        int keliling;
11
12        panjang = input.nextInt();
13        lebar = input.nextInt();
14
15        keliling = 2 * (panjang + lebar);
16
17        System.out.println(keliling);
18    }
19
20 }
```

Contoh Studi Kasus 2

- Bu Dina adalah salah satu nasabah bank ABC yang menabung sebesar Rp. 5 juta rupiah.
- Bank tersebut memberikan bunga sebesar 2% setiap tahun.
- Bu Dina menabung selama 5 tahun.
- Berapakah bunga dan jumlah tabungan yang dapat diambil sekarang!

Contoh Studi Kasus 2

1. Menentukan Algoritma

- Input: jumlah tabungan awal, lama menabung
- Output: bunga, jumlah tabungan akhir
- Data lain = prosentase bunga = 0,02
- Proses:
 1. Input jumlah tabungan awal, lama menabung
 2. Hitung bunga = lama menabung x prosentase bunga x jumlah tabungan awal
 3. Hitung jumlah tabungan akhir = bunga + jumlah tabungan awal
 4. Output bunga dan jumlah tabungan akhir

2. Mengidentifikasi variable dan jenis tipe data berdasarkan algoritma

Variabel	Tipe data
jml_tabungan_awal	int
lama_menabung	int
jml_tabungan_akhir	double
bunga	double
prosentase_bunga = 0.02	double

Contoh Studi Kasus 2

```
1 import java.util.Scanner;
2
3 public class bank {
4     Run | Debug
5     public static void main (String [] args){
6         Scanner input =new Scanner(System.in);
7
8         int jml_tabungan_awal, lama_menabung;
9         double prosentase_bunga =0.02, bunga, jml_tabungan_akhir;
10
11        System.out.println ("masukkan jumlah tabungan awal anda");
12        jml_tabungan_awal = input.nextInt();
13        System.out.println ("masukkan lama menabung anda");
14        lama_menabung= input.nextInt();
15
16        bunga= lama_menabung*prosentase_bunga*jml_tabungan_awal;
17        jml_tabungan_akhir=bunga+jml_tabungan_awal;
18
19        System.out.println ("Bunga adalah " +bunga);
20        System.out.println ("Jumlah tabungan akhir anda adalah " +jml_tabungan_akhir);
21
22    }
23 }
```

Baris 8, 9 → deklarasi variable dan tipe data

Baris 12, 14 → inputan

Baris 16, 17 → proses

Baris 19, 20 → output