



## POLITEKNIK MASAMY INTERNASIONAL

SK Menristekdikti RI Nomor: 731/KPT/I/2018

Jalan Ikan Paus No.10-15 Kertosari Banyuwangi - 68411  
Telp (0333) 3384593 – <http://polmain.info>

Form:  
B.Ak/eva/04/20

### PROGRAM STUDI D3 TEKNIK KOMPUTER

Nama Dosen : *Arif Fahmi, S.T.,M.T.*

Mata Kuliah : Pemrograman Berorientasi Object

Semester : 4 (Empat)

Kode Mata Kuliah : TKV4044

Th. Akdm : 2019/2020.

## BAB

### DASAR PEMROGRAMAN JAVA

#### SUB BAB :

1. Elemen-elemen dasar dalam pemrograman java
2. Tipe Data dan Variabel
3. Operator

## TUJUAN MATERI

1. Mahasiswa memahami konsep elemen-elemen dasar pemrograman java
2. Mahasiswa dapat menerapkan penggunaan dasar-dasar pemrograman java
3. Mahasiswa mengetahui cara pembuatan variable dan penggunaannya.
4. Mahasiswa mengetahui konfersi tipe data
5. Mahasiswa mampu menyelesaikan permasalahan terkait penggunaan variable dan tipe data
6. Mahasiswa mengetahui berbagai macam operator java dan penggunaannya
7. Mahasiswa mampu membuat program java dengan IDE Netbeans

## REFERENSI

1. J.Eck, D. (2006). *Introduction to Programming Using Java*. Geneva.
2. Ady Wicaksono, Dasar – Dasar Pemrograman Java 2, Penerbit PT Elex Media Komputindo, Jakarta, 2002

# DASAR PEMROGRAMAN JAVA

## 1. Elemen-elemen Dasar dalam Pemograman Java

### 1.1 Komentar program

Komentar merupakan hal yang sangat penting dalam pemrograman. Komentar bisa memberikan informasi kepada programmer atau pembaca program mengenai maksud dari baris program yang diberikan komentar tersebut. Komentar tidak dieksekusi ketika program dijalankan.

Java mendefinisikan tiga jenis komentar , Diantaranya :

1. Single-Line comment,
2. Multiline comment,
3. JavaDoc comment.

#### Single-line comment

Single-line comment atau komentar satu baris merupakan komentar yang di mulai dengan tanda (“ // “) dan berakhir di akhir baris tersebut.

Contoh :

```
MyClass.java
public class MyClass {
    public static void main(String[] args) {
        // This is a comment
        System.out.println("Hello World");
    }
}
```

#### Multiline comment

Multiline comment merupakan komentar yang bisa memiliki banyak baris. komentar jenis ini dimulai dari tanda /\* sampai dengan tanda \*/

Contoh :

```
MyClass.java
public class MyClass {
    public static void main(String[] args) {
        /* The code below will print the words Hello World
        to the screen, and it is amazing */
        System.out.println("Hello World");
    }
}
```

#### JavaDoc comment

Komentar javadoc khusus digunakan untuk men-generate dokumentasi HTML untuk program Java Anda. Anda dapat menciptakan komentar javadoc dengan memulai baris dengan /\*\* dan mengakhirinya dengan \*/. Seperti Komentar C\_style, dapat juga menjangkau beberapa baris. Komentar ini juga dapat terdiri atas tag-tag untuk menambahkan lebih banyak informasi pada komentar Anda. Sebagai contoh,

```
/**
 *
 * Ini contoh penggunaan javadoc komentar \n
 * @author Florence Balagtas
 * @version 1.2
 */
```

## 1.2 Blok program

Pemrograman Java memungkinkan kita untuk mengelompokkan satu atau lebih statemen kedalam sebuah blok program. Dalam pemrograman Java, blok program diawali dengan tanda { dan diakhir oleh tanda }. Pada saat kita menuliskan kode program, kita akan banyak bekerja dengan blok program. Dalam java, blok program dapat diterapkan untuk definisi kelas, method, struktur pengulangan, struktur pemilihan, dan yang lainnya. Berikut contoh pembuatan blok program didalam java.

```
MyClass.java
public class MyClass {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

### **Petunjuk Penulisan Program:**

1. Pada saat pembuatan blok, Anda dapat meletakkan kurung kurawal buka pada baris dengan pernyataan seperti contoh sebagai berikut ,

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

atau Anda dapat meletakkan kurung kurawal pada baris selanjutnya, seperti,

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

2. Anda harus memberi jarak (indent) pernyataan selanjutnya setelah awal dari blok , seperti contoh berikut,

```
public static void main( String[] args ){
    System.out.println("Hello"); System.out.println("world");
}
```

## 1.3 Identifier

Identifier adalah suatu tanda yang mewakili nama-nama variabel, method, class, dsb. Contoh dari Identifier adalah : Hello, main, System, out.

Pendeklarasian Java adalah case-sensitive. Hal ini berarti bahwa Identifier : Hello tidak sama dengan hello. Identifier harus dimulai dengan salah satu huruf, underscore “\_”, atau tanda dollar “\$”. Hurufnya dapat berupa huruf besar maupun huruf kecil. Karakter selanjutnya dapat menggunakan nomor 0 sampai 9.

Identifier tidak dapat menggunakan kata kunci dalam Java seperti class, public, void, dsb.

### **Petunjuk Penulisan Program:**

1. Untuk pemberian nama dari class Java, diberikan huruf kapital untuk huruf pertama pada nama class. Untuk nama method dan variabel, huruf pertama dari kata harus dimulai dengan huruf kecil. Sebagai contoh:

```
ThisIsAnExampleOfClassName
```

```
me
```

```
thisIsAnExampleOfMethodName
```

```
ame
```

2. Pada kasus untuk identifier lebih dari satu kata, menggunakan huruf kapital untuk mengindikasikan awal dari kata kecuali kata pertama. Sebagai contoh, charArray, fileName, ClassName.
3. Hindari menggunakan underscores pada awal identifier seperti \_read atau \_write.

## 1.4 Literal

Literals adalah tanda bahwa tidak terjadi perubahan atau konstan. Macam-macam literals dalam Java adalah : *Integer Literals*, *Floating-Point Literals*, *Boolean Literals*, *Character Literals* dan *String Literals*.

### A. Integer Literals

*Integer literals* dibedakan dalam beberapa format yang berbeda: **desimal** (berbasis 10), **heksadesimal** (berbasis 16), and **oktal** (berbasis 8). Dalam penggunaan tipe data integer pada program, kita harus mengikuti aturan penggunaan beberapa notasi khusus.

Untuk angka desimal, kita tidak memerlukan notasi khusus. Kita hanya menulis angka desimal seperti apa adanya. untuk angka heksadesimal, hal itu harus ditandai oleh “0x” atau “0X”. untuk oktal, ditandai oleh “0”.

Sebagai contoh, mewakili angka 12. Penulisan dalam bentuk desimalnya adalah 12, Sementara dalam heksadesimal, menjadi 0xC, dan dalam oktal, nilai tersebut sama dengan 014.

Default tipe data untuk integer literals adalah **int**. Int adalah signed 32-bit value. Pada kasus-kasus tertentu Anda dapat berharap untuk memaksa integer literal untuk menjadi tipe data **long** dengan menambahkan karakter “l” or “L”. tipe data long ditandai oleh ditampilkannya data dalam 64-bit. Kita akan membahas mengenai tipe data pada kesempatan selanjutnya

### B. Floating-point Literals

*Floating point literals* mewakili bentuk desimal dengan bagian yang terpisah. Sebagai contoh adalah 3.1415. *Floating point literals* dapat dinyatakan dalam notasi standard atau scientific. Sebagai contoh, 583.45 dinyatakan dalam notasi standard, Sementara 5.8345e2 dinyatakan dalam notasi scientific. Default *Floating point literals* mempunyai tipe data **double** yang dinyatakan dalam 64-bit. Untuk menggunakan ketelitian yang lebih kecil (32-bit) **float**, hanya dengan menambahkan karakter “f” atau “F”.

### C. Boolean Literals

*Boolean literals* hanya memiliki dua nilai, true atau false

### D. Character Literals

*Character Literals* diwakili oleh karakter single Unicode. Karakter Unicode adalah 16-bit character set yang menggantikan 8-bit ASCII character set. Unicode memungkinkan penggunaan simbol dan karakter khusus dari bahasa lain. Untuk menggunakan *character literals*, karakter tersebut di dalam tanda *single quote* ( ' ') (single quote delimiters). Sebagai contoh huruf a, diwakili sebagai 'a'. Untuk menggunakan karakter khusus seperti karakter baris baru, *backslash* digunakan diikuti dengan karakter kode. Sebagai contoh, '\n' untuk karakter baris baru atau ganti baris, '\r' untuk menyatakan nilai balik (carriage return), '\b' untuk backspace.

1.5 Separator

Pada bahasa pemrograman Java, separator digunakan untuk memisahkan satu bagian program dengan program lainnya. Bahasa Java sendiri memiliki banyak sekali jenis-jenis separator. Salah satu separator yang sering kali digunakan dalam pembuatan program dalam bahasa java adalah separator berjenis semicolon/"titik koma" (;). Separator ini berfungsi untuk memisahkan satu statemen dengan statemen lainnya. Berikut adalah jenis-jenis separator yang digunakan dalam Bahasa Java :

Simbol	Nama Separator	<i>fungsi</i>
()	Parentheses (Tanda kurung)	a. Digunakan untuk mengapit daftar parameter yang digunakan dalam method <ul style="list-style-type: none"><li>Contoh</li><li>public int tambah(int a, int b)</li></ul> b. Digunakan untuk mengapit ekspresi dalam operasi tertentu <ul style="list-style-type: none"><li>Contoh</li><li>int hasil=2*(3+2)</li></ul> c. Digunakan untuk mengapit ekspresi dalam statemen kontrol <ul style="list-style-type: none"><li>Contoh</li><li>if (hasil==10)</li></ul> d. Digunakan untuk melakukan <i>typecast</i> <ul style="list-style-type: none"><li>Contoh</li><li>byte=(byte)nilai;</li></ul>
{ }	Braches (Kurung kurawal )	a. Digunakan untuk mengapit badan program (Class, method, kontrol pilihan, dll) <ul style="list-style-type: none"><li>Contoh</li><li>class Kelasku{....}</li></ul> b. Digunakan untuk mengisi nilai inisial pada Array <ul style="list-style-type: none"><li>Contoh</li><li>int nilai ={2,3,4}</li></ul>
[ ]	Bracket (Kurung siku )	a. Digunakan untuk mendeklarasikan array <ul style="list-style-type: none"><li>Contoh</li><li>int b[]= new int [5];</li></ul> b. Digunakan untuk mengakses nilai dari elemen array <ul style="list-style-type: none"><li>Contoh</li><li>b[4]= 5;</li></ul>
;	Semicolon(Titik koma )	a. Digunakan untuk memisahkan/mengakhiri statement <ul style="list-style-type: none"><li>Contoh</li><li>int a=0 ;</li></ul>
,	Comma (Koma )	a. Digunakan untuk memisahkan variable pada saat deklarasi variable <ul style="list-style-type: none"><li>Contoh</li><li>char a , b , c;</li></ul> b. Digunakan untuk memanipulasi statemen perulangan <i>for</i> <ul style="list-style-type: none"><li>Contoh</li><li>for (int i=0; i&lt; 5; i++, j++)</li></ul>
.	Period (Titik )	a. Digunakan untuk memisahkan nama paket, nama subpaket, dan nama kelas <ul style="list-style-type: none"><li>Contoh</li><li>import java.io.IOException;</li></ul> b. Digunakan untuk memisahkan data/method dari sebuah referensi objek <ul style="list-style-type: none"><li>Contoh</li><li>obj.a=10; obj.setA(4);</li></ul>



1.6 Kata kunci

*Keyword* adalah kata kunci yang digunakan dalam bahasa pemrograman. salah satunya adalah java, kata kunci ini digunakan pada sebuah class/variable/method/konstruktor, untuk menentukan sifat, jenis atau hak akses yang digunakan, didalam java terdapat banyak sekali jeni keyword yang bisa kita gunakan, terdapat 50 jenis kata kunci pada Java.

Keyword	Fungsi dan Deskripsi
Abstract	Digunakan untuk menyatakan sebuah kelas atau method menjadi abstrak.
Assert	kesalahan pada saat program dijalankan, keyword ini dapat membantu proses Debugging.
Boolean	Boolean digunakan untuk menyatakan variabel yang dapat menyimpan nilai bertipe boolean, yaitu true atau false.
Break	Break digunakan untuk mengeksekusi program ke perintah selanjutnya. Jadi, keyword break ini melewati statement yang tidak diinginkan oleh programmer, dan melanjutkan ke statement selanjutnya.
Case	Case digunakan untuk memberikan opsi atau pilihan dari pernyataan switch.
Catch	Sebuah blok satement untuk menangkap exception jika terdapat exception/error didalam blok try.
Char	Menyatakan variabel yang dapat menyimpan nilai karakter 16 bit.
Class	Digunakan untuk mendefinisikan dan mengimplementasikan objek.
Const	Const tidak digunakan dan tidak mempunyai fungsi.
Continue	Digunakan untuk melanjutkan eksekusi program perulangan.
Default	Default dapat digunakan dalam pernyataan switch untuk pernyataan yang akan dieksekusi jika case tidak sesuai dengan nilai yang ditentukan.
Do	Digunakan bersama dengan while untuk membuat do-while pengulangan, untuk mengeksekusi suatu statement sebelum kondisi pada while di evaluasi
Double	Untuk menyimpan nilai pecahan, mempunyai ukuran 8 bytes dan 64 bit serta panjang Range +/- 1.8 x 10308 dengan jumlah presisi/digit 15.
Else	Kata kunci yang digunakan bersama dengan if untuk membuat pernyataan jika kondisi pada if tidak terpenuhi
Enum	Kata kunci yang digunakan untuk mendeklarasikan tipe enumerasi. Enumerations memperpanjang kelas dasar Enum.
Extends	Digunakan untuk pewarisan sifat dari suberclass/kelas induk pada subclass/kelas anaknya.
Final	Untuk mengubah atribut menjadi konstanta, sehinggal tidak bisa diubah, diganti, atau dioverride oleh method lain.
Finally	Bagian dari blok try yang selalu dieksekusi
Float	Untuk menyatakan nilai pecahan, tipe data Float mempunyai ukuran 4 bytes dan 32 bit serta panjang Range +/- 3.4 x 1038 dengan jumlah presisi/digit 6-7.
For	Tipe pengulangan (looping)
Goto	Goto tidak digunakan dan tidak memiliki fungsi.
If	Untuk membuat statement yang bersyarat (conditional statement).
Implements	Mendefinisikan interface yang diimplementasikan oleh kelas.
Import	Meng-import paket (package).



Instanceof	Untuk menguji jika objek adalah instance dari kelas.
Int	Tipe data bilangan bulat, tipe data Int mempunyai ukuran 32 bit dan Range -2147483648 s.d. 2147483647
Interface	Tipe abstrak dengan method yang dapat diimplementasikan oleh kelas tersebut.
Long	Tipe data bilangan bulat, memiliki ukuran paling besar yaitu 64 bit dan Range -922337209 s.d. 922337209.
Native	Method yang diimplementasikan oleh host system
New	New digunakan untuk membuat sebuah instance dari sebuah kelas atau array.
Null	Null yang berarti tidak memiliki nilai
Package	Untuk mendefinisikan nama Paket kelas
Private	Hak Akses agar sifat-sifat dari suatu class tidak bisa diwariskan dan hanya bisa diakses oleh classnya sendiri.
Protected	Hak Akses yang memungkinkan terjadi pewarisan data antar class asalkan dalam satu package yang sama.
Public	Hak Akses yang memungkinkan atribut/data dapat diakses oleh class mana saja, didalam package yang sama ataupun berbeda.
Return	Untuk mengembalikan nilai dari sebuah variable
Short	Tipe data bilangan bulat, mempunyai ukuran 16 bit, tipe data ini mempunyai Range lebih tinggi dari Byte, yaitu - 32768 s.d. 32767.
Static	Salah satu sifat untuk variable dan method, agar kita dapat memanggil secara langsung tanpa harus membuat objek dari class
Super	Digunakan untuk mengakses anggota kelas yang diwariskan oleh kelas di mana ia muncul.
Switch	Statement untuk menentukan pilihan
Synchronized	Method atau blok kode yang atomic kepada thread.
This	Digunakan untuk mewakili sebuah instance dari kelas dimana ia muncul, atau digunakan jika ada nama variable yang sama dengan parameter
Throw	Keyword throw digunakan untuk melempar suatu exception dalam program
Throws	Kata kunci throws digunakan untuk mengenali daftar eksepsi yang mungkin di-throw oleh suatu method.
Transient	Merupakan suatu proses dimana state daripada objek tersebut dapat kita simpan menjadi bentuk deretan byte, dan juga sebaliknya.
Try	Untuk memantau suatu statement yang kemungkinan dapat terjadi exception.
Void	Void adalah method yang tidak memiliki nilai kembali/return, biasanya digunakan tidak untuk mencari nilai dalam suatu operasi.
Volatile	Digunakan dalam deklarasi lapangan untuk menentukan bahwa variabel yang diubahasynchronously oleh bersamaan menjalankan thread. Metode, kelas dan interface sehingga tidak dapat dideklarasikan volatile.
While	Kata kunci while digunakan untuk membuat pengulangan, selama kondisi true sampai ekspresi bernilai false.
Byte	Byte menyatakan variabel yang dapat menyimpan nilai byte (8 bit) Range -128 s.d. 127.

2. Tipe Data

Java memiliki tipe data yang dapat dikategorikan menjadi dua kelompok, yaitu tipe data primitif dan referensi.

2.1 Tipe Data Primitif

Delapan macam tipe data primitif dalam pemrograman Java, yaitu :

A. Integer (Bilangan Bulat)

Integer merupakan tipe data numerik yang digunakan apabila tidak berurusan dengan pecahan atau bilangan desimal. Tipe data numerik yang termasuk integer adalah sebagai berikut :

Type	Deskripsi
<i>Byte</i>	Memiliki nilai integer dari -128 sampai +127 dan menempati 1 byte ( 8 bits ) di memori
<i>Short</i>	Memiliki nilai integer dari -32768 sampai 32767 dan menempati 2 bytes ( 16 bits ) di memori
<i>Int</i>	Memiliki nilai integer dari -2147483648 sampai 2147483647 dan menempati 4 bytes ( 32 bits ) di memori
<i>Long</i>	Memiliki nilai dari -9223372036854775808 sampai 9223372036854775807 dan menempati 8 bytes ( 64 bits ) di memori

Bilangan integer biasanya menggunakan int, dan bukan byte, short maupun long. Bilangan integer juga mengenal nilai positif dan negatif ( *signed number* ). Tipe data byte dan short hanya digunakan pada aplikasi khusus yang memperhatikan penggunaan memori. Sedangkan long jarang digunakan karena jarang memerlukan bilangan sebesar kapasitas long.

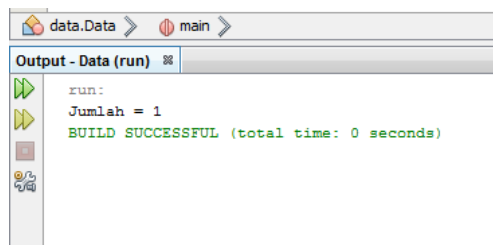
- Byte

Type byte umumnya digunakan pada saat kita bekerja dengan sebuah data stream dari suatu file maupun jaringan, yaitu untuk keperluan proses membaca/menulis. Selain itu, tipe ini juga digunakan saat bekerja dengan data biner yang tidak kompatibel dengan tipe-tipe lain yang didefinisikan di dalam Java

**Contoh ke-1 Program Byte Dengan IDE Netbeans:**

```
1
2 package data;
3
4 public class Data {
5
6     public static void main(String[] args) {
7         // deklarasi variabel
8         byte jumlah=1;
9         // output
10        System.out.println("Jumlah = "+jumlah);
11
12    }
13
14 }
15
```





```
package data;

public class Data {

    public static void main(String[] args) {

        // deklarasi variabel

        byte jumlah=1;

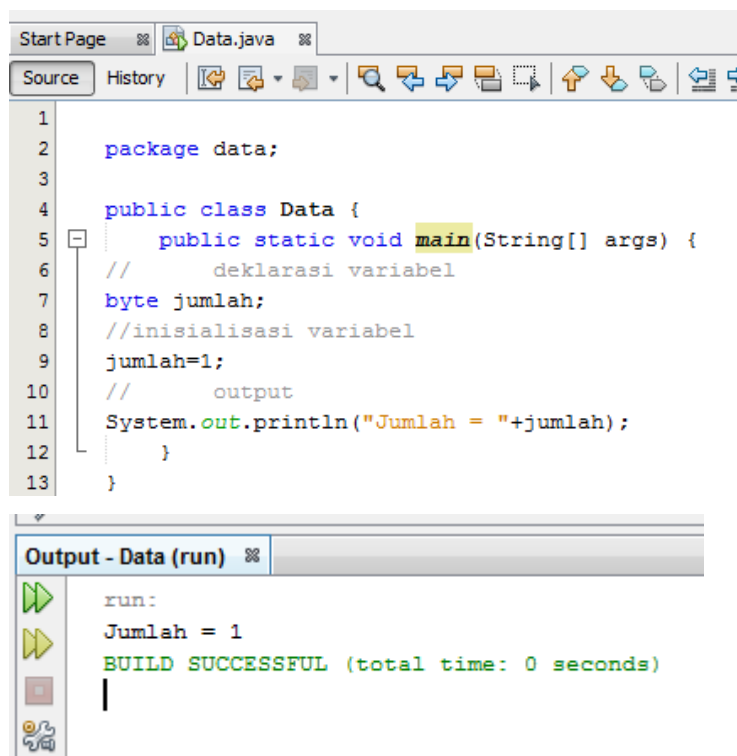
        // output

        System.out.println("Jumlah = "+jumlah);

    }

}
```

### Contoh ke-2 Program Byte Dengan IDE Netbeans:



```
package data;

public class Data {

    public static void main(String[] args) {

//    deklarasi variabel

byte jumlah;

//inisialisasi variabel

jumlah=1;

//    output

System.out.println("Jumlah = "+jumlah);

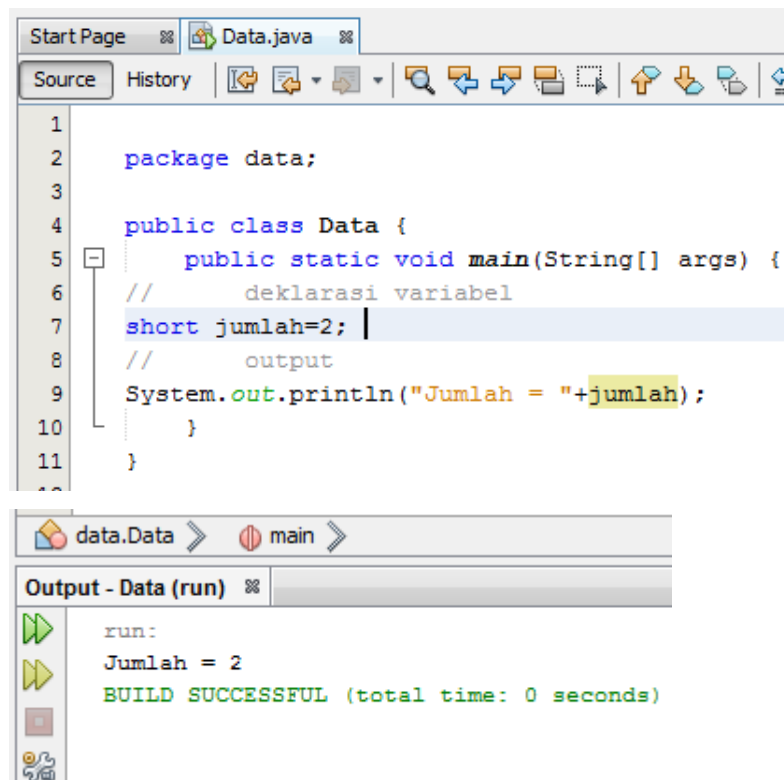
    }

}
```

- **Short**

Pada umumnya diaplikasikan pada komputer-komputer 16-bit, yang saat ini semakin jarang keberadaanya

**Contoh Program Short Dengan IDE Netbeans:**



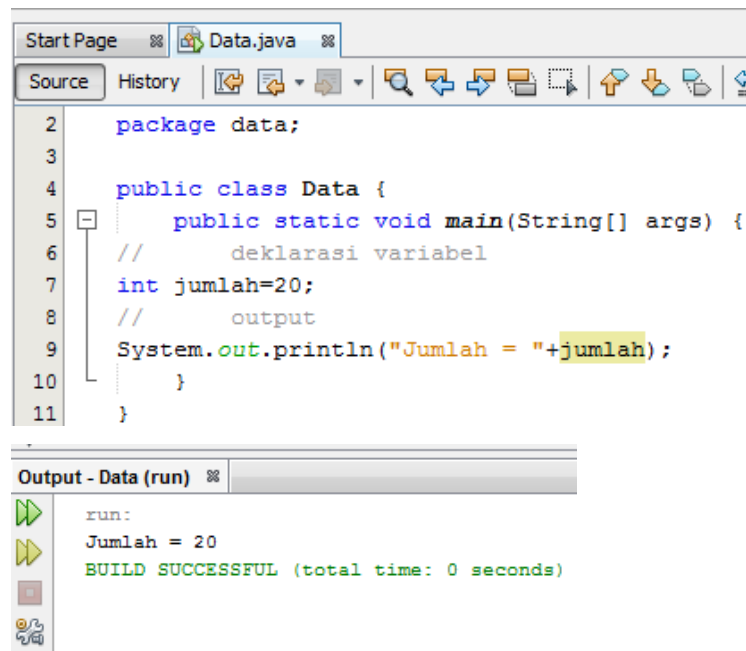
```
1 package data;
2
3
4 public class Data {
5     public static void main(String[] args) {
6         // deklarasi variabel
7         short jumlah=2;
8         // output
9         System.out.println("Jumlah = "+jumlah);
10    }
11 }
```

run:  
Jumlah = 2  
BUILD SUCCESSFUL (total time: 0 seconds)

```
package data;
public class Data {
    public static void main(String[] args) {
        // deklarasi variabel
        short jumlah=2;
        // output
        System.out.println("Jumlah = "+jumlah);
    }
}
```

- **Integer (Int)**

Tipe ini juga merupakan tipe yang paling banyak dipakai dalam merepresentasikan angka dalam Java, dikarenakan dianggap paling efisien dibandingkan dengan tipe-tipe integer lainnya. Tipe *Int* banyak digunakan untuk indeks dalam struktur pengulangan maupun dalam konstruksi sebuah *array*. Selain itu, secara teori setiap ekspresi yang melibatkan tipe integer *byte*, *short*, *int*, *long*) semuanya akan dipromosikan ke *int* terlebih dahulu sebelum dilakukan proses perhitungan.

**Contoh Program Int Dengan IDE Netbeans:**

```
2 package data;
3
4 public class Data {
5     public static void main(String[] args) {
6         // deklarasi variabel
7         int jumlah=20;
8         // output
9         System.out.println("Jumlah = "+jumlah);
10    }
11 }
```

Output - Data (run) %

```
run:
Jumlah = 20
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;

public class Data {

    public static void main(String[] args) {

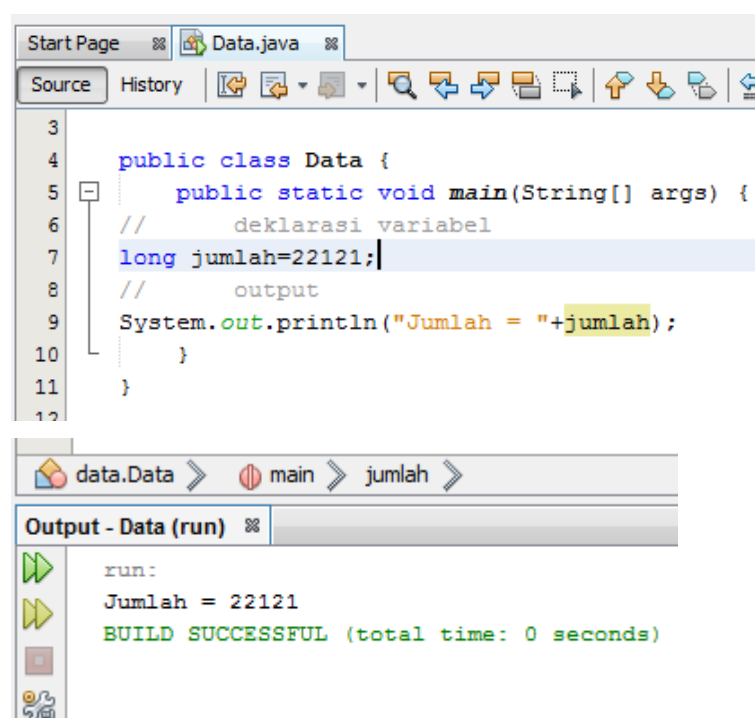
        // deklarasi variabel
        int jumlah=20;
        // output
        System.out.println("Jumlah = "+jumlah);

    }

}
```

- **Long**

Tipe ini digunakan untuk kasus-kasus tertentu yang nilainya berada di luar rentang tipe int, karna tipe ini punya range paling tinggi dibanding Integer lainnya. Dengan kata lain, tipe long terpaksa digunakan jika data memiliki range diluar range int.

**Contoh Program Long Dengan IDE Netbeans:**

```
3
4 public class Data {
5     public static void main(String[] args) {
6         // deklarasi variabel
7         long jumlah=22121;
8         // output
9         System.out.println("Jumlah = "+jumlah);
10    }
11 }
```

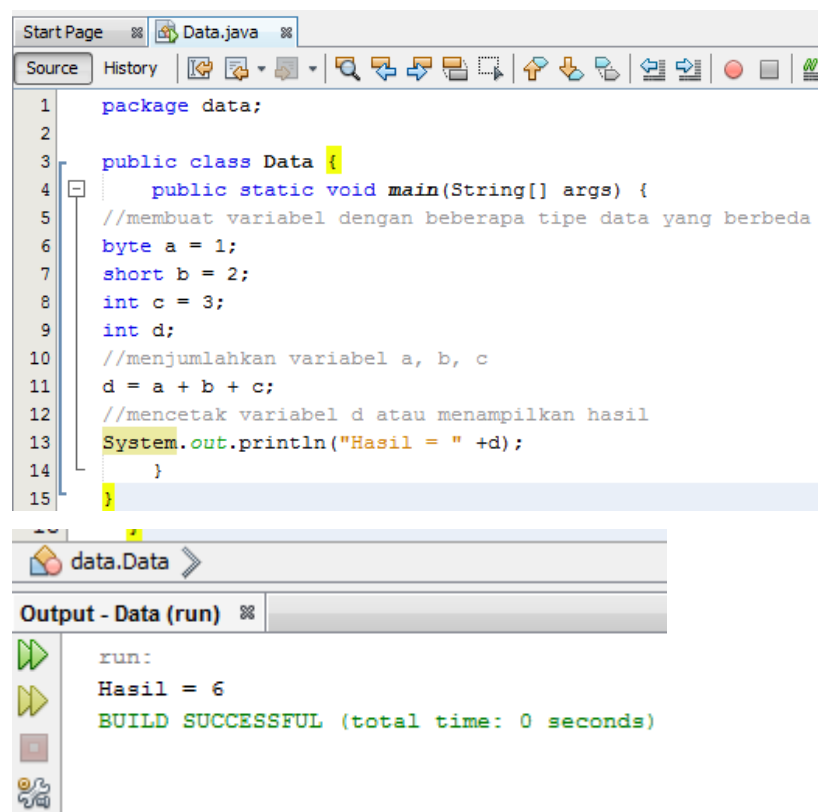
data.Data > main > jumlah >

Output - Data (run) %

```
run:
Jumlah = 22121
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Integer** adalah sebuah tipe data yang paling dasar yang berupa bilangan yang tidak mengandung pecahan desimal. Tipe data ini juga memiliki urutan tersendiri, sehingga kita dapat menggunakannya sesuai kebutuhan dalam pemrograman dan kita dapat membandingkannya satu dengan yang lainnya. Di bagian tabel diatas kita bisa tahu nama-nama tipe data integer ini dan kita bisa menggunakannya sesuai kebutuhan. Akan tetapi tipe data *int* lebih umum digunakan dalam pemrogramannya.

#### **Contoh Program Integer Dengan IDE Netbeans:**



The screenshot shows the NetBeans IDE interface. The top window displays the source code for `Data.java`. The code defines a package `data` and a public class `Data` with a `main` method. Inside the `main` method, variables `a` (byte), `b` (short), and `c` (int) are declared and assigned values 1, 2, and 3 respectively. A fourth variable `d` (int) is declared but not assigned. A comment indicates that `d` is the sum of `a`, `b`, and `c`. The code then calculates `d = a + b + c` and prints the result using `System.out.println("Hasil = " + d);`. The bottom window shows the output of the program, which is `Hasil = 6`, and a message indicating a successful build.

```
1 package data;
2
3 public class Data {
4     public static void main(String[] args) {
5         //membuat variabel dengan beberapa tipe data yang berbeda
6         byte a = 1;
7         short b = 2;
8         int c = 3;
9         int d;
10        //menjumlahkan variabel a, b, c
11        d = a + b + c;
12        //mencetak variabel d atau menampilkan hasil
13        System.out.println("Hasil = " + d);
14    }
15 }
```

Output - Data (run)

```
run:
Hasil = 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;

public class Data {

    public static void main(String[] args) {

        //membuat variabel dengan beberapa tipe data yang berbeda

        byte a = 1;
        short b = 2;
        int c = 3;
        int d;

        //menjumlahkan variabel a, b, c

        d = a + b + c;

        //mencetak variabel d atau menampilkan hasil

        System.out.println("Hasil = " + d);

    }

}
```

B. Floating Point (Bilangan Pecahan)

Floating Point digunakan untuk menangani bilangan desimal atau perhitungan yang lebih detail dibanding integer. Ada dua macam floating point, yaitu :

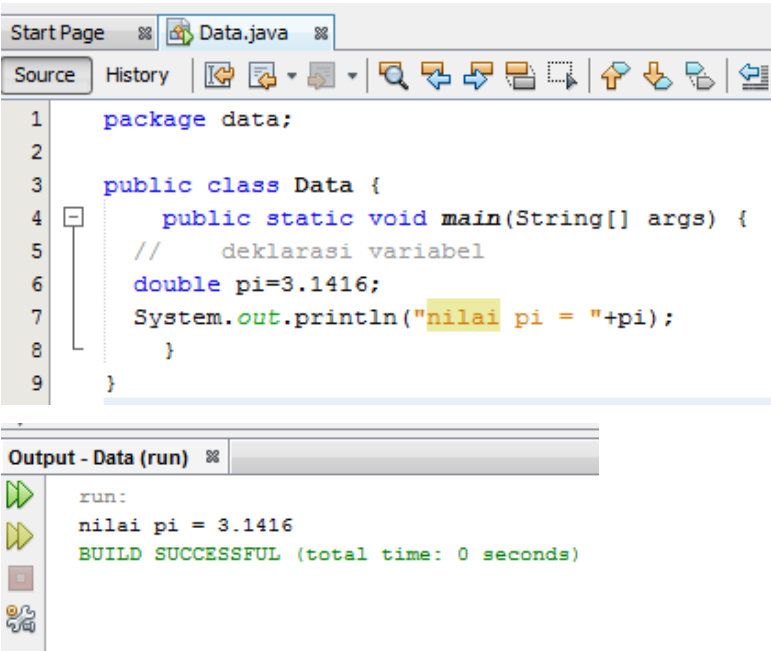
Tipe	Deskripsi
Float	memiliki nilai $-3.4 \times 10^8$ sampai $+3.4 \times 10^8$ dan menempati 4 byte di memori
Double	memiliki nilai $-1.7 \times 10^{308}$ sampai $+1.7 \times 10^{308}$

Semua bilangan pecahan atau desimal dalam Java tanpa diakhiri huruf **f** akan dianggap sebagai double. Sedangkan bilangan yang ingin dikategorikan sebagai float harus diakhiri dengan huruf **F**. Misalnya : 4.22 F atau 2.314f. Sedangkan untuk bilangan double, bisa menambah dengan huruf **D**, karena secara default bilangan dengan koma atau pecahan atau desimal akan dianggap sebagai double.

- Double

Tipe ini mengandung tingkat ketelitian ganda atau presisi ganda (double precision) dan menggunakan ruang penyimpanan 64-bit untuk menyimpan nilai. Tipe double tentu lebih cepat untuk melakukan perhitungan-perhitungan matematis daripada tipe float. Untuk perhitungan yang bersifat bilangan riil dan menghasilkan hasil yang lebih akurat, maka lebih baik menggunakan tipe double.

Contoh Program Double Dengan IDE Netbeans:



```
package data;

public class Data {
    public static void main(String[] args) {
        // deklarasi variabel
        double pi=3.1416;
        System.out.println("nilai pi = "+pi);
    }
}
```

C. Char (Karakter)

Char adalah karakter tunggal yang didefinisikan dengan diawali dan diakhiri dengan tanda ‘ ( petik tunggal ). Char berbeda dengan String, karena String bukan merupakan tipe data primitif, tetapi sudah merupakan sebuah objek. Tipe char mengikuti aturan unicode, sehingga dapat menggunakan kode /u kemudian diikuti bilangan dari 0 sampai 65535, tetapi yang biasa digunakan adalah bilangan heksadesimal dari 0000 sampai FFFF. Misalnya : ‘\u123’.

Selain karakter biasa, juga terdapat karakter khusus yang didefinisikan dengan cara mengawalinya menggunakan tanda \ seperti pada tabel berikut :

Kode	Nama	Nilai Unicode
\b	Backspace	\u0008
\t	Tab	\u0009
\n	Linefeed	\u000a
\r	Carriage return	\u000d
\*	Double quote	\u0022
\'	Single quote	\u0027
\\	Backslash	\u005c

Contoh Program Char Dengan IDE Netbeans:

Start Page

Data.java

Source

History

1

package data;

2

3

public class Data {

4

public static void main(String[] args) {

5

// deklarasi variabel

6

char HurufPertama;

7

//inisialisasi variabel

8

HurufPertama='a';

9

//output

10

System.out.println ("Huruf pertama adalah "+HurufPertama);

11

}

12

}

Output - Data (run)

run:

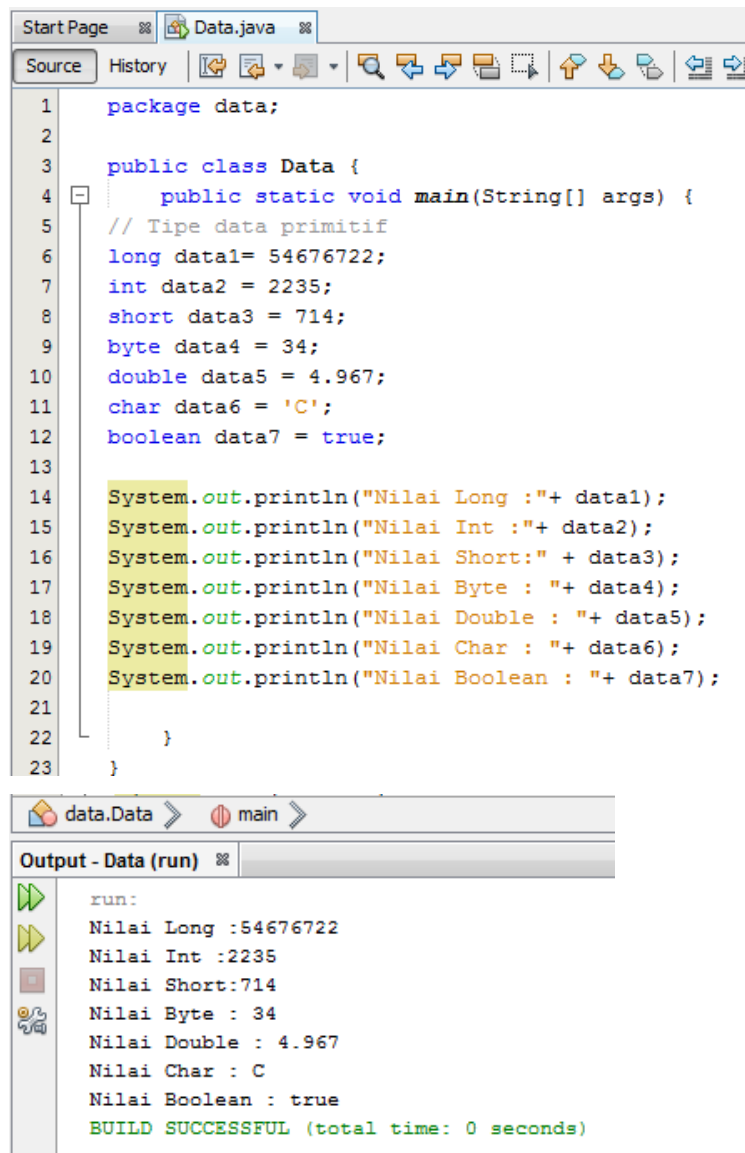
Huruf pertama adalah a

BUILD SUCCESSFUL (total time: 0 seconds)

## D. Boolean

Dalam Java dikenal tipe data boolean yang terdiri dari dua nilai saja, yaitu true dan false. Boolean sangat penting dalam mengevaluasi suatu kondisi, dan sering digunakan untuk menentukan alur program.

**Contoh Program Boolean Dengan IDE Netbeans:**



```
1 package data;
2
3 public class Data {
4     public static void main(String[] args) {
5         // Tipe data primitif
6         long data1= 54676722;
7         int data2 = 2235;
8         short data3 = 714;
9         byte data4 = 34;
10        double data5 = 4.967;
11        char data6 = 'C';
12        boolean data7 = true;
13
14        System.out.println("Nilai Long :"+ data1);
15        System.out.println("Nilai Int :"+ data2);
16        System.out.println("Nilai Short:" + data3);
17        System.out.println("Nilai Byte : "+ data4);
18        System.out.println("Nilai Double : "+ data5);
19        System.out.println("Nilai Char : "+ data6);
20        System.out.println("Nilai Boolean : "+ data7);
21
22    }
23 }
```

data.Data > main >

Output - Data (run) >

```
run:
Nilai Long :54676722
Nilai Int :2235
Nilai Short:714
Nilai Byte : 34
Nilai Double : 4.967
Nilai Char : C
Nilai Boolean : true
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 2.2 Tipe Data Referensi

Kelebihan pemrograman berorientasi objek adalah dapat mendefinisikan tipe data baru yang merupakan objek dari class tertentu. Tipe data ini digunakan untuk mereferensikan objek atau class tertentu, seperti **String**.

### 3. Variabel

Variabel merupakan *container* yang digunakan untuk menyimpan suatu nilai pada sebuah program dengan tipe tertentu. Untuk mendefinisikan variabel, kita dapat menggunakan identifier untuk menamai variabel tersebut.

#### 3.1 Identifier

Identifier adalah kumpulan karakter yang dapat digunakan untuk menamai variabel, method, class, interface, dan package. Sebagaimana bahasa pemrograman pada umumnya, Java memiliki peraturan untuk identifier yang valid atau sah. Identifier bisa disebut valid atau sah apabila diawali dengan :

- Huruf / abjad
- Karakter mata uang
- Underscore ( \_ )

Identifier dapat terdiri dari :

- Huruf / abjad
- Angka
- Underscore ( \_ )

Identifier tidak boleh mengandung @, spasi atau diawali dengan angka. Selain itu, identifier tidak boleh menggunakan keyword atau kata-kata yang memiliki arti atau digunakan dalam pemrograman Java. Daftar Keyword Java :

abstract	double	int	strictfp
boolean	flse	static	super
break	fxtdends	long	switch
byte	final	native	synchronized
case	finally	new	this
catch	float	package	throw
char	for	private	throws
class	goto	protected	transient
const	if	public	try
continue	implements	return	void
default	import	short	volatile
do	instanceof	interface	while

Selain menggunakan karakter biasa, kita juga dapat menggunakan unicode sebagai identifier.



### 3.2 Mendeklarasikan Variabel

Sintaks dasar :

***[tipe data] [nama variabel]***

Menuliskan tipe data dari variabel, contoh :

***int bilangan;***

***char karakter;***

***float bildesimal;***

***boolean status;***

Setelah mendeklarasikan variabel dengan tipe data, selanjutnya memberikan nilai variabel tersebut dengan tanda =

***bilangan = 20;***

***karakter = 'k';***

***bildesimal = 22.2f;***

***status = true;***

Dapat juga mendeklarasikan dan memberikan nilai dalam satu baris.

***int bilangan = 20;***

***char karakter = 'k';***

***float bildesimal = 22.2f;***

***boolean status = true;***

Kita dapat membuat variabel menjadi konstanta yang tidak dapat diubah nilainya dengan menambahkan keyword sebelum tipe data dari variabel. Contoh:

***final int konstantainteger = 10;***

***final float pajak = 15.5;***

Agar konstanta ini dapat diakses oleh class lain tanpa harus membuat objek terlebih dahulu, maka kita dapat menambahkan modifier public dan keyword static seperti berikut :

***public static final konstantainteger = 10;***

4. Operator

Dalam Java, ada beberapa tipe operator. Ada operator aritmatika, operator relasi, operator logika, dan operator kondisi. Operator ini mengikuti bermacam-macam prioritas yang pasti sehingga compilernya akan tahu yang mana operator untuk dijalankan lebih dulu dalam kasus beberapa operator yang dipakai bersama-sama dalam satu pernyataan.

4.1 Operator Aritmatika

Operator aritmetik adalah operator-operator yang digunakan untuk melakukan perhitungan-perhitungan matematis. Adapapun yang termasuk operator aritmatika sebagaimana tabel berikut

Oprator	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus (Sisa hasil bagi)
++	Increment (menaikkan nilai dengan 1)
--	Decrement (menurunkan nilai dengan 1)

Contoh Program Operator Aritmatika Dasar Dengan IDE Netbeans:

```
Start Page  Data.java
Source History
1 package data;
2 public class Data {
3     public static void main(String[] args) {
4         System.out.println("Operasi aritmetika " + "pada tipe integer");
5         int a = 7 + 4;
6         int b = a - 4;
7         int c = a * b;
8         int d = c / 3;
9         int e = -a;
10        System.out.println("Nilai a: " + a);
11        System.out.println("Nilai b: " + b);
12        System.out.println("Nilai c: " + c);
13        System.out.println("Nilai d: " + d);
14        System.out.println("Nilai e: " + e);
15        System.out.println();
16        System.out.println("Operasi aritmetika " + "pada tipe floating-point");
17        double fa = 7 + 4;
18        double fb = fa - 4;
19        double fc = fa * fb;
20        double fd = fc / 3;
21        double fe = -a;
22        System.out.println("Nilai fa: " + fa);
23        System.out.println("Nilai fb: " + fb);
24        System.out.println("Nilai fc: " + fc);
25        System.out.println("Nilai fd: " + fd);
26        System.out.println("Nilai fe: " + fe);
27    }
28 }
```

```
Output - Data (run) %
run:
Operasi aritmetika pada tipe integer
Nilai a: 11
Nilai b: 7
Nilai c: 77
Nilai d: 25
Nilai e: -11

Operasi aritmetika pada tipe floating-point
Nilai fa: 11.0
Nilai fb: 7.0
Nilai fc: 77.0
Nilai fd: 25.666666666666668
Nilai fe: -11.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;

public class Data {

    public static void main(String[] args) {

        System.out.println("Operasi aritmetika " + "pada tipe integer");
        int a = 7 + 4;
        int b = a - 4;
        int c = a * b;
        int d = c / 3;
        int e = -a;

        System.out.println("Nilai a: " + a);
        System.out.println("Nilai b: " + b);
        System.out.println("Nilai c: " + c);
        System.out.println("Nilai d: " + d);
        System.out.println("Nilai e: " + e);
        System.out.println();

        System.out.println("Operasi aritmetika " + "pada tipe floating-point");
        double fa = 7 + 4;
        double fb = fa - 4;
        double fc = fa * fb;
        double fd = fc / 3;
        double fe = -a;

        System.out.println("Nilai fa: " + fa);
        System.out.println("Nilai fb: " + fb);
        System.out.println("Nilai fc: " + fc);
        System.out.println("Nilai fd: " + fd);
        System.out.println("Nilai fe: " + fe);

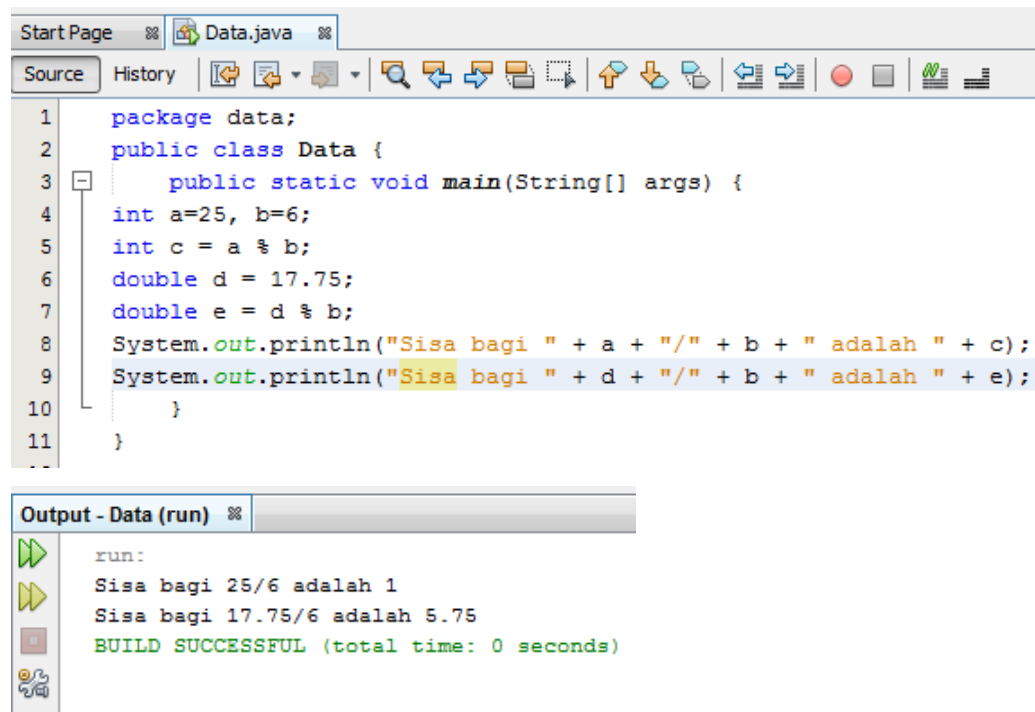
    }

}
```

- Operator Modulus

Operator modulus (%) digunakan untuk menentukan sisa hasil bagi dari sebuah operasi pembagian bilangan bulat maupun bilangan riil.

**Contoh Program Operator Modulus Dengan IDE Netbeans:**



The screenshot shows the NetBeans IDE interface. The top toolbar includes icons for Start Page, History, and various editing actions. The main editor window displays a Java file named 'Data.java' with the following code:

```
1 package data;
2 public class Data {
3     public static void main(String[] args) {
4         int a=25, b=6;
5         int c = a % b;
6         double d = 17.75;
7         double e = d % b;
8         System.out.println("Sisa bagi " + a + "/" + b + " adalah " + c);
9         System.out.println("Sisa bagi " + d + "/" + b + " adalah " + e);
10    }
11 }
```

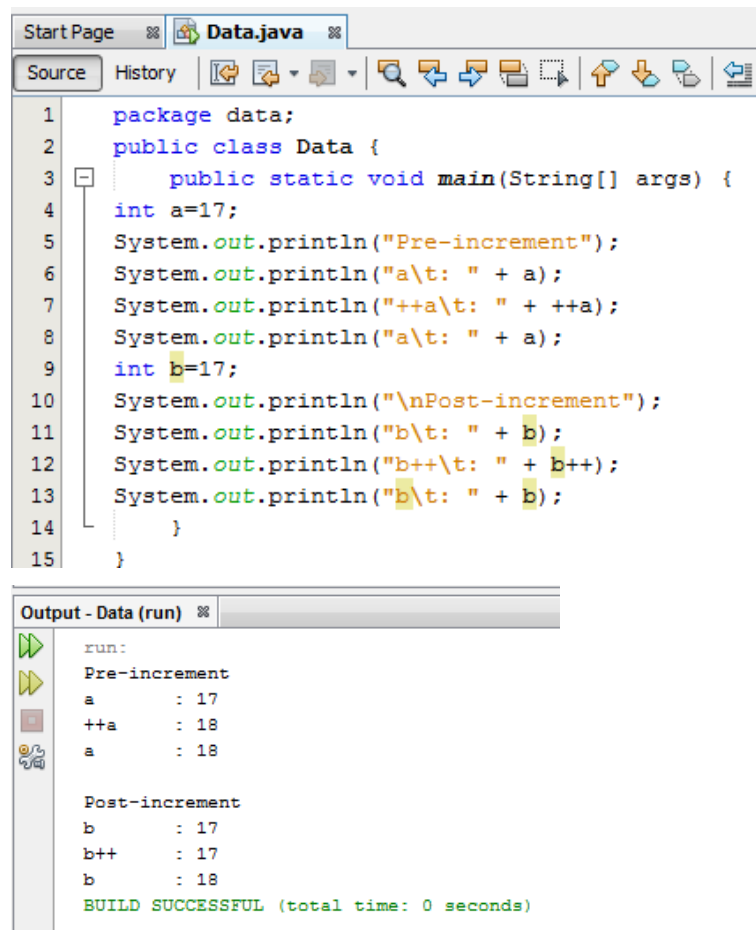
Below the editor, the 'Output - Data (run)' window shows the execution results:

```
run:
Sisa bagi 25/6 adalah 1
Sisa bagi 17.75/6 adalah 5.75
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;
public class Data {
    public static void main(String[] args) {
int a=25, b=6;
int c = a % b;
double d = 17.75;
double e = d % b;
System.out.println("Sisa bagi " + a + "/" + b + " adalah " + c);
System.out.println("Sisa bagi " + d + "/" + b + " adalah " + e);
    }
}
```

- Operator Increment dan Decrement

Increment decrement operator adalah operator yang berguna untuk menaikkan 1 nilai (increment) dan menurunkan 1 nilai (decrement). Berdasarkan urutan eksekusi penaikann dan penurunan nilainya, increment-decrement operator ini dapat diklasifikasikan menjadi 2 macam, yaitu pre-increment/decrement dan post-increment/decrement.

**Contoh Program Operator Increment Dengan IDE Netbeans:**

The screenshot shows the NetBeans IDE with a Java file named `Data.java`. The code defines a package `data` and a class `Data` with a `main` method. It demonstrates pre-increment and post-increment operators on variables `a` and `b`. The output window shows the results of the program execution.

```
1 package data;
2 public class Data {
3     public static void main(String[] args) {
4         int a=17;
5         System.out.println("Pre-increment");
6         System.out.println("a\t: " + a);
7         System.out.println("++a\t: " + ++a);
8         System.out.println("a\t: " + a);
9         int b=17;
10        System.out.println("\nPost-increment");
11        System.out.println("b\t: " + b);
12        System.out.println("b++\t: " + b++);
13        System.out.println("b\t: " + b);
14    }
15 }
```

Output - Data (run)

```
run:
Pre-increment
a      : 17
++a    : 18
a      : 18

Post-increment
b      : 17
b++    : 17
b      : 18
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;

public class Data {

    public static void main(String[] args) {

int a=17;

System.out.println("Pre-increment");

System.out.println("a\t: " + a);

System.out.println("++a\t: " + ++a);

System.out.println("a\t: " + a);

int b=17;

System.out.println("\nPost-increment");

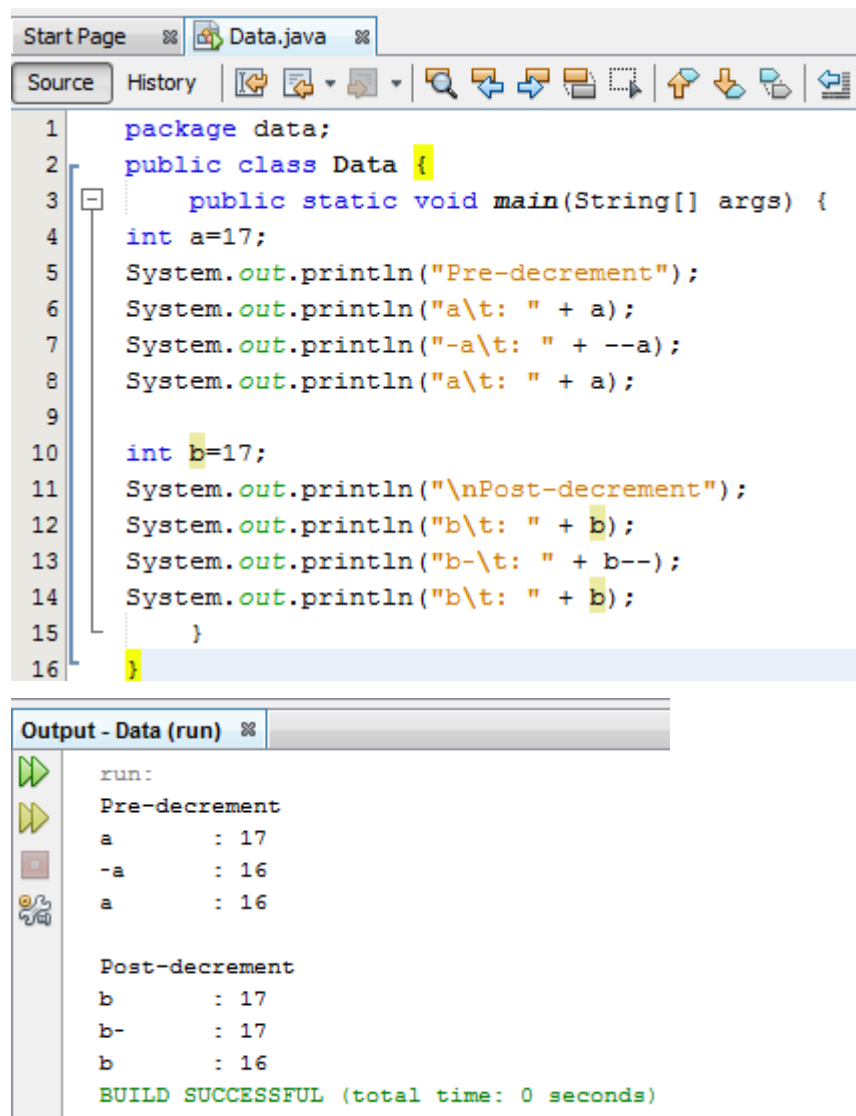
System.out.println("b\t: " + b);

System.out.println("b++\t: " + b++);

System.out.println("b\t: " + b);

    }

}
```

**Contoh Program Operator Decrement Dengan IDE Netbeans:**

```
1 package data;
2 public class Data {
3     public static void main(String[] args) {
4         int a=17;
5         System.out.println("Pre-decrement");
6         System.out.println("a\t: " + a);
7         System.out.println("-a\t: " + --a);
8         System.out.println("a\t: " + a);
9
10        int b=17;
11        System.out.println("\nPost-decrement");
12        System.out.println("b\t: " + b);
13        System.out.println("b-\t: " + b--);
14        System.out.println("b\t: " + b);
15    }
16 }
```

Output - Data (run)

```
run:
Pre-decrement
a      : 17
-a     : 16
a      : 16

Post-decrement
b      : 17
b-     : 17
b      : 16
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;

public class Data {

    public static void main(String[] args) {

int a=17;

System.out.println("Pre-decrement");

System.out.println("a\t: " + a);

System.out.println("-a\t: " + --a);

System.out.println("a\t: " + a);


int b=17;

System.out.println("\nPost-decrement");

System.out.println("b\t: " + b);

System.out.println("b-\t: " + b--);

System.out.println("b\t: " + b);

    }

}
```

### 4.2 Operator rasional

Operator relasional adalah operator yang menyatakan hubungan antara satu operand dengan operand lainnya. Hasil yang diberikan dari operasi ini akan bernilai boolean (true/false).

Oprator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan

Contoh Program Operator Relasional Dengan IDE Netbeans:

StartPage % Data.java %

SourceHistory

1package data;

2

3public class Data {

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

5        int a=7, b=17;

6        System.out.println("a == b bernilai " + (a == b));

7        System.out.println("a != b bernilai " + (a != b));

8        System.out.println("a > b bernilai " + (a > b));

9        System.out.println("a < b bernilai " + (a < b));

10        System.out.println("a >= b bernilai " + (a >= b));

11        System.out.println("a <= b bernilai " + (a <= b));

12    }

13}

14

Output - Data (run) %

run:

a == b bernilai false

a != b bernilai true

a > b bernilai false

a < b bernilai true

a >= b bernilai false

a <= b bernilai true

BUILD SUCCESSFUL (total time: 0 seconds)

```
package data;

public class Data {
    public static void main(String[] args) {
        int a=7, b=17;
        System.out.println("a == b bernilai " + (a == b));
        System.out.println("a != b bernilai " + (a != b));
        System.out.println("a > b bernilai " + (a > b));
        System.out.println("a < b bernilai " + (a < b));
        System.out.println("a >= b bernilai " + (a >= b));
        System.out.println("a <= b bernilai " + (a <= b));
    }
}
```

### 4.3 Operator Logika

Operator logika berguna ketika kita ingin menguji dua kondisi atau lebih secara bersamaan apakah syarat itu bernilai benar atau tidak. Operator logika digunakan untuk melakukan operasi terhadap dua operand yang bertipe Boolean. Hasil yang diberikan dari operasi ini juga akan bertipe Boolean.

Oprator	Keterangan
&&	AND
	OR
^	XOR
!	NOT

Contoh Program Operator Logika Dengan IDE Netbeans:

Start PageData.java

SourceHistory

1package data;

2

3public class Data {

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

5        System.out.println("Operasi AND");

6        System.out.println("true && true = " + (true && true));

7        System.out.println("true && false = " + (true && false));

8        System.out.println("true || true = " + (true || true));

9        System.out.println("true || false = " + (true || false));

10        System.out.println("false || true = " + (false || true));

11        System.out.println("true ^ true = " + (true ^ true));

12        System.out.println("true ^ false = " + (true ^ false));

13        System.out.println("false ^ true = " + (false ^ true));

14        System.out.println("!true = " + (!true));

15        System.out.println("!false = " + (!false));

16    }

17}

Output - Data (run)

run:  
Operasi AND  
true && true = true  
true && false = false  
true || true = true  
true || false = true  
false || true = true  
true ^ true = false  
true ^ false = true  
false ^ true = true  
!true = false  
!false = true  
BUILD SUCCESSFUL (total time: 0 seconds)

```
package data;

public class Data {

    public static void main(String[] args) {

        System.out.println("Operasi AND");
        System.out.println("true && true = " + (true && true));
        System.out.println("true && false = " + (true && false));
        System.out.println("true || true = " + (true || true));
        System.out.println("true || false = " + (true || false));
        System.out.println("false || true = " + (false || true));
        System.out.println("true ^ true = " + (true ^ true));
        System.out.println("true ^ false = " + (true ^ false));
        System.out.println("false ^ true = " + (false ^ true));
        System.out.println("!true = " + (!true));
        System.out.println("!false = " + (!false));

    }

}
```



#### 4.4 Operator Assignment (Penugasan)

Operator penugasan berguna untuk memberikan nilai kesuatu variabel.Operator penugasan menggunakan tanda sama dengan ( = ). Dibawah ini beberapa operator penugasan.

Oprator	Keterangan
=	Pemberian Nilai
+=	Penambahan bilangan
-=	Pengurangan bilangan
*=	Pengalian bilangan
/=	Pembagian bilangan
%=	Pemrolehan sisa bagi

**Operator +=** digunakan untuk menaikkan nilai terhadap suatu variabel contohnya a +=2, jika semula variabel a berisi 5 maka a saat ini akan bernilai 7.

**Operator -=** digunakan untuk menurunkan nilai terhadap suatu variabel contohnya a -=2, jika semula variabel a berisi 5 maka a saat ini akan bernilai 3.

**Operator /=** digunakan untuk membagi nilai terhadap suatu variabel contohnya  $a /= 2$ , jika semula variabel  $a$  berisi 5 maka  $a$  saat ini akan bernilai 2.5

**Operator %**= digunakan untuk memperoleh sisa pembagian nilai terhadap suatu isi variabel contohnya `x%=2`, berarti nilai variabel `x` (bertipe `int`) akan diisi dengan sisa pembagian `x` dengan `2`. Kalau misal `x` berisi `5` maka `x` saat ini akan bernilai `1`.

**Contoh Program Operator Assignment Dengan IDE Netbeans:**

The screenshot displays an IDE with a Java file named `Data.java`. The code defines a `public class Data` with a `main` method. Inside `main`, variables `a`, `b`, `c`, `d`, and `e` are declared and assigned values. `a` is 1, `b` is 2, `c` is 3, `d` is 4, and `e` is 5. Then, `a` is incremented by 5, `b` is multiplied by 4, `c` is decremented by 1, and `d` is divided by 2. Finally, the values of `a`, `b`, `c`, `d`, and `e` are printed. The output window shows the results: `a = 6`, `b = 8`, `c = 2`, `d = 2`, and `e = 1`. The build is successful, taking 0 seconds.

```
package data;

public class Data {
    public static void main(String[] args) {
        int a = 1; int b = 2; int c = 3;
        int d = 4; int e = 5;
        a += 5; b *= 4;
        c -= 1; d /= 2; e %= 2;
        System.out.println("a = " + a);
        System.out.println("b = " + b);
        System.out.println("c = " + c);
        System.out.println("d = " + d);
        System.out.println("e = " + e);
    }
}
```

Output - Data (run)

```
run:
a = 6
b = 8
c = 2
d = 2
e = 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

### 4.5 Operator Bitwise

Operator bitwise digunakan untuk memanipulasi bit-bit dari nilai data yang ada di memori. Operator bitwise dalam bahasa Java :

Oprator	Keterangan	Contoh	Hasil
>>	Pergeseran bit ke kanan	12 >> 2	3
<<	Pergeseran bit ke kiri	7 << 2	28
&	Manipulasi bit dengan logika AND	11 & 7	8
	Manipulasi bit dengan logika OR	9   3	11
^	Manipulasi bit dengan logika XOR	8 ^ 6	14
~	Manipulasi bit dengan logika NOT	~ 13	- 14

Setiap numerik yang dimanipulasi harus dirubah dulu kedalam bentu biner, pada contoh di atas maka binernya ada sebagai berikut.

Desimal	Biner
3	11
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100

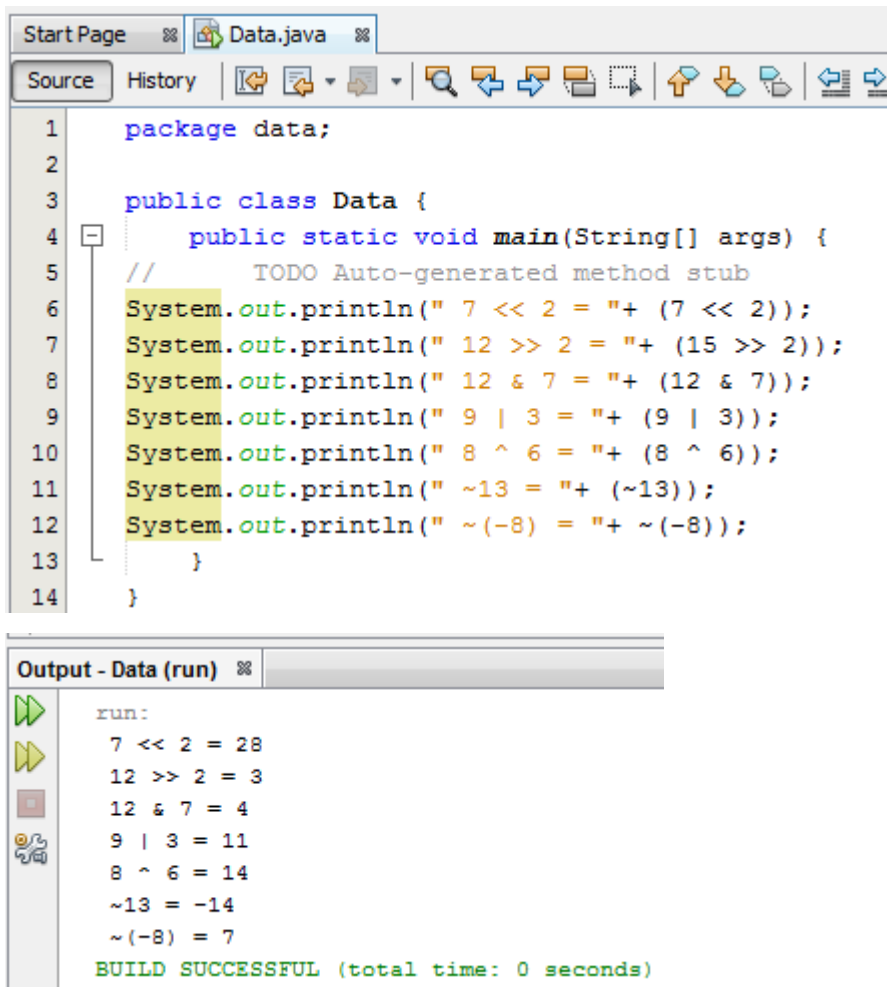
Setelah didapatkan biner dari sertiap numerik yang akan dimanipulasi maka barulah dilakukan operasi bitwise sesuai dengan operator yang di gunakan. Penjelasan dapat dilihat pada tabel berikut ini :

Penggunan	Penjelasan	Hasil Biner	Hasil Desimal
12 >> 2	1100 digeser bit kekanan	11	3
7 << 2	111 digeser 2 bit ke kiri	11100	28
12 &7	1100 & 0111 maka setiap bit dibandingkan dengan logika AND	11 & 7	8
9   3	0101   0011 maka setiap bit dibandingkan dengan logika OR	9   3	11
8 ^ 6	0100 ^ 0110 maka setiap bit dibandingkan dengan logika XOR	8 ^ 6	14
~13	~ (1101)	~ 13	- 14

Pada operator ~ (Not) ada perhitungan tersendiri, secara sederhana rumusnya sebagai berikut :

~(n) = - (n+1), contoh: ~(82) = -83

~(-n) = n-1, contoh: ~(-43) = 42

**Contoh Program Operator Bitwise Dengan IDE Netbeans:**

The screenshot shows the NetBeans IDE interface. The top toolbar includes icons for Start Page, History, and various editing actions. The main editor window displays the source code for a Java class named `Data` in the `data` package. The code defines a `main` method that prints the results of several bitwise operations. Below the editor, the 'Output - Data (run)' window shows the execution results, confirming that the program ran successfully in 0 seconds.

```
1 package data;
2
3 public class Data {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         System.out.println(" 7 << 2 = "+ (7 << 2));
7         System.out.println(" 12 >> 2 = "+ (15 >> 2));
8         System.out.println(" 12 & 7 = "+ (12 & 7));
9         System.out.println(" 9 | 3 = "+ (9 | 3));
10        System.out.println(" 8 ^ 6 = "+ (8 ^ 6));
11        System.out.println(" ~13 = "+ (~13));
12        System.out.println(" ~(-8) = "+ ~(-8));
13    }
14 }
```

run:

```
7 << 2 = 28
12 >> 2 = 3
12 & 7 = 4
9 | 3 = 11
8 ^ 6 = 14
~13 = -14
~(-8) = 7
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
package data;
```

```
public class Data {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println(" 7 << 2 = "+ (7 << 2));
        System.out.println(" 12 >> 2 = "+ (15 >> 2));
        System.out.println(" 12 & 7 = "+ (12 & 7));
        System.out.println(" 9 | 3 = "+ (9 | 3));
        System.out.println(" 8 ^ 6 = "+ (8 ^ 6));
        System.out.println(" ~13 = "+ (~13));
        System.out.println(" ~(-8) = "+ ~(-8));
    }
}
```



# Tugas

## Catatan.

- a. Tugas dikumpulkan maks pada hari Jum'at tgl 08-04-2020 pukul 16.00
- b. Tugas boleh dikumpulkan dalam bentuk (PDF,PPT,WORD)
- c. File dikirim ke email : [fahmi03031995@gmail.com](mailto:fahmi03031995@gmail.com)
- d. Format file dengan tata cara: NamaMhs\_NIM\_PBO

## Soal

1. Buatlah resume Tipe data, Variabel,dan Operator JAVA
2. Buatlah program tentang penggunaan Tipe data, variabel dan Operator

Catatan ; boleh menggunakan Notepad, Decoder,maupun IDE Java (Netbeans, Jcreator)