



**MODUL PEMROGRAMAN BERORIENTASI OBJEK**  
**CTI 211**

**MODUL PERTEMUAN 2**  
**PENGENALAN VARIABLE, TIPE DATA, METODE, DAN OPERATOR**

**DISUSUN OLEH**  
**7174 - SAWALI WAHYU, S.KOM, M.KOM**

Universitas  
**Esa Unggul**

**UNIVERSITAS ESA UNGGUL**  
**FAKULTAS ILMU KOMPUTER**  
**TAHUN 2020**

## PENGENALAN VARIABLE, TIPE DATA, METODE, DAN OPERATOR

### A. Kemampuan Akhir Yang Diharapkan

Setelah mempelajari modul ini, diharapkan mahasiswa mampu :

1. Mahasiswa mampu memahami inisialisasi *tipe data*, *method*, *expression*, *variable* dan *operation* dalam *java programming*
2. Mahasiswa mampu membuat dan mengimplementasikan fungsi dasar tipe data, *method*, *expression*, *variable* dan *operation* ke dalam sebuah aplikasi
3. Mahasiswa mampu membuat fungsi dari inisialisasi *tipe data*, *method*, *expression*, *variable* dan *operation* dalam *java programming*

### B. Uraian dan Contoh

- 1) Pengenalan Konsep Dasar Data Type, Operation, Dan Variable
- 2) Penggunaan Data Type dan Latihan
- 3) Penggunaan Operation dan Latihan
- 4) Penggunaan Variable dan Latihan



Universitas  
**Esa Unggul**

## 1. Pengenalan Konsep Dasar Data Type, Operation, Dan Variable

Tipe Data Adalah Sekumpulan Data Dengan Nilai-Nilai Yang Memiliki Karakteristik Yang Telah Ditetapkan.

Method Adalah Sebuah Blok Program Terpisah (Diluar Program Utama) Yang Kita Gunakan Untuk Menyelesaikan Masalah Khusus.

Variabel Adalah Lokasi Penyimpanan Dan Terkait Nama Simbolis Yang Berisi Beberapa Kuantitas Yang Diketahui Atau Tidak Diketahui Atau Informasi, Nilai.

Ekspresi Dalam Bahasa Pemrograman Adalah Kombinasi Dari Nilai-Nilai Eksplisit, Konstanta, Variabel, Operator, Dan Fungsi Yang Ditafsirkan Menurut Aturan Prioritas Tertentu Dan Asosiasi Untuk Sebuah Bahasa Pemrograman

## 2. Penggunaan Data Type dan Latihan

Tipe data merupakan karakteristik sebuah variabel. Karakteristik ini akan menentukan fungsi variabel itu nantinya. Ada tipe data yang menyimpan data angka, ada juga tipe data yang menyimpan data teks. Perbedaan yang jelas antara tipe data angka dan teks adalah, tipe data angka dapat dihitung menggunakan operasi matematika, sedangkan tipe data teks tidak bisa dihitung.

Pada Skrip 2-1 di atas, String adalah tipe data teks, sehingga nilai yang disimpan di dalam variabel tersebut adalah teks, meskipun diisi dengan nilai "1000" atau "seribu" tetap dianggap sebagai teks yang tidak bisa dihitung.

Tipe data bukan hanya tentang menyimpan jenis data angka atau teks, namun juga tentang kapasitas penyimpanannya yang berbeda-beda. Sebagai contoh, tipe data angka Byte hanya bisa menyimpan angka dari -128 sampai 127, sedangkan tipe data Short dapat menyimpan angka -32768 sampai 32767 begitu seterusnya. Untuk tipe-tipe data lainnya dijelaskan pada sub bab berikutnya.

Ada 2 jenis tipe data pada pemrograman Java, yaitu:

- 1) Tipe data primitif, yaitu tipe data yang paling dasar pada Java. Tipe data ini terdiri dari 3, yaitu :
  - a) Tipe data angka, tipe data yang menyimpan data angka.
  - b) Tipe data karakter, tipe data yang menyimpan 1 karakter teks.

- c) Tipe data boolean, tipe data yang hanya bernilai True atau False.
- 2) Tipe data turunan, yaitu tipe data yang dibuat dari satu atau lebih tipe data primitif. Pada umumnya tipe data ini berbentuk class.

Pada modul ini tidak membahas tentang tipe data turunan, kecuali tipe data String. Pembahasan tipe data String digabungkan dengan tipe data karakter.

## TIPE DATA ANGKA

Tipe data ini dapat menyimpan nilai angka dengan kapasitas tertentu. Sehingga variabel yang menggunakan tipe data ini tidak bisa menyimpan teks ataupun karakter di dalamnya. Salah satu fungsi tipe data angka adalah untuk menyimpan data angka yang dapat dihitung dengan operasi matematika seperti penjumlahan, pengurangan, perkalian dan sebagainya.

Ada beberapa jenis tipe data angka, antara lain :

- a) Bilangan Bulat (Integer), merupakan tipe data untuk menyimpan bilangan bulat. Tipe data ini terdiri dari beberapa jenis sesuai dengan kapasitas penyimpanannya.

Tipe	Keyword	Kapasitas	Rentang Nilai
Byte	byte	1 byte	-128 sampai 127
Short Integer	short	2 byte	-32768 s/d 32767
Integer	int	4 byte	-2147483648 s/d 2147483647
Long Integer	long	8 byte	-9223372036854775808 s/d 9223372036854775807

- b) Bilangan Berkoma (Decimal), merupakan tipe data untuk menyimpan bilangan berkoma. Tingkat presisi dan kapasitas nilai yang dapat disimpan terdiri dari 2 tipe.

Tipe	Keyword	Kapasitas	Rentang Nilai
Float	float	4 byte	$-3.4 \times 10^{38}$ to $3.4 \times 10^{38}$
Double	double	8 byte	$-1.8 \times 10^{308}$ to $1.8 \times 10^{308}$

## LATIHAN :

Perhatikan skrip berikut ini. Lengkapi skrip berikut ini sehingga program berjalan dengan benar.

### LATIHAN 1. PENJUMLAHAN :

```
public class Penjumlahan {  
    public static void main(String[] args) {  
        int nilaiA = 1500;  
        int nilaiB = 75;  
        // Lengkapi kode pada baris ini  
        System.out.println("Nilai A = " + nilaiA);  
        System.out.println("Nilai B = " + nilaiB);  
        System.out.println("Jumlah = " + jumlah);  
    }  
}
```

### LATIHAN 2. PEMBAGIAN

Perhatikan skrip berikut ini. Lengkapi skrip berikut ini sehingga program berjalan dengan benar.

```
public class Pembagian {  
    public static void main(String[] args) {  
        int a = 20;  
        int b = 8;  
        // Lengkapi kode pada baris ini  
        System.out.println("A = " + a);  
        System.out.println("B = " + b);  
        System.out.println("A dibagi B = " + hasilBagi);  
    }  
}
```

## TIPE DATA TEKS

Tipe data teks adalah tipe data yang menyimpan data dalam bentuk teks. Data teks tidak hanya terdiri dari huruf saja, angka yang disimpan ke variabel bertipe teks akan dianggap sebagai teks sehingga tidak bisa digunakan untuk menghitung.

Ada beberapa jenis tipe data teks, antara lain:

- 1) Char, tipe data yang hanya menyimpan 1 karakter teks saja.
- 2) String, tipe data yang menyimpan rangkaian teks paling banyak 2<sup>31</sup>-1 karakter atau sekitar 2GB teks.

## LATIHAN 1. SALAM 1

Perhatikan skrip berikut. Perbaikilah skrip berikut ini sehingga menghasilkan output sesuai dengan output yang diinginkan.

```
public class Salam2 {  
    public static void main(String[] args) {  
        // Buat karakter char c1 = 'S';  
        char c2 = 'A'; char c3 = 'L'; char c4 = 'A'; char c5  
        = 'M';  
        // Print    System.out.println(c1+c2+c3+c4+c5);  
        System.out.println(c5+c4+c3+c2+c1);  
    }  
}
```

Output :

```
SALAM  
MALAS
```

## LATIHAN 2. SALAM 2

Perhatikan skrip berikut. Perbaikilah skrip berikut ini sehingga menghasilkan output sesuai dengan output yang diinginkan.

```
public class Kalimat {  
    public static void main(String[] args) {  
        // Buat variabel String subjek = "Saya";  
        String prediket = "menulis"; String objek = "program Java";  
        String keterangan = "hari ini";  
        // Print kalimat S+P+O+K  
        System.out.println(subjek+prediket+objek+keterangan);  
    }  
}
```

Output :

```
Saya menulis program Java hari ini
```

## LATIHAN 3. KALIMAT

Perhatikan skrip berikut. Perbaikilah skrip berikut ini sehingga menghasilkan output sesuai dengan output yang diinginkan.

```

public class Kalimat {
    public static void main(String[] args) {
        // Buat variabel String subjek = "Saya";
        String prediket = "menulis"; String objek = "program Java";
        String keterangan = "hari ini";
        // Print kalimat S+P+O+K
        System.out.println(subjek+prediket+objek+keterangan);
    }
}

```

## TIPE DATA BOOLEAN

Tipe data boolean merupakan tipe data yang hanya menyimpan nilai True atau False. Tipe data ini digunakan untuk keperluan percabangan. Percabangan lebih lanjut akan dibahas pada bab 5.

## LATIHAN 1. GABUNGAN SEMUA TIPE DATA

Tuliskan program berikut ini. Kemudian, berikan komentar untuk setiap baris kode program yang ada. Komentar yang dibuat berisi penjelasan singkat masing-masing baris kode.

```

public class TrueFalse {
    public static void main(String[] args) {
        String nama = "Budi";
        int nilai = 54;
        boolean lulus = false; System.out.println("Nama
        : "+nama); System.out.println("Nilai : "+nilai);
        System.out.println("Status : "+lulus);
    }
}

```

### 3. Penggunaan Operation dan Latihan

Java menyediakan banyak set operator untuk memanipulasi variabel. Kita dapat membagi semua operator Java ke dalam kelompok berikut :

- **Arithmetic Operators**

Operator	Hasil
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus
++	Increment
--	Decrement
+=	Persamaan penjumlahan
-=	Persamaan pengurangan

- **Relational Operators**

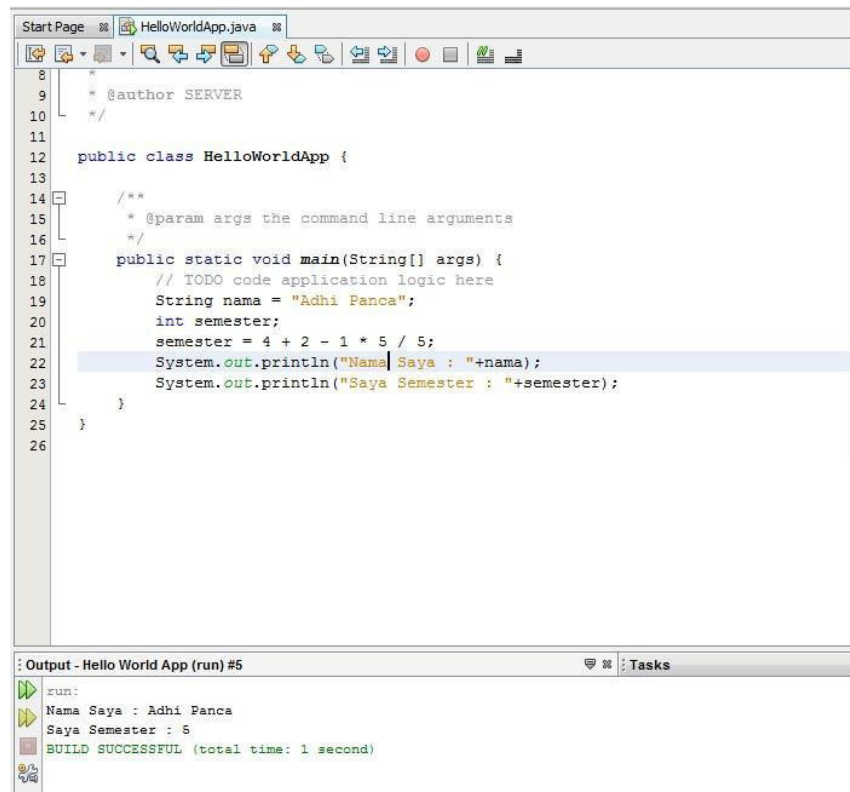
Operator	Penggunaan	Keterangan
>	op1 > op2	op1 lebih besar dari op2
>=	op1 >= op2	op1 lebih besar dari atau sama dengan op2
<	op1 < op2	op1 kurang dari op2
<=	op1 <= op2	op1 kurang dari atau sama dengan op2
==	op1 == op2	op1 sama dengan op2
!=	op1 != op2	op1 tidak sama dengan op2

- **Logical Operators**

Operator	Keterangan
&&	Operasi AND
	Operasi OR
^	Operasi XOR (exclusive OR)
!	Operasi NOT (negasi)

Pada contoh Operator Java kita gunakan operator aritmatika, karena operator relational dan operator logika biasa digunakan pada sebuah kondisi atau perulangan, maka untuk contoh operator tersebut akan di cantumkan pada penjelasan percabangan dan perulangan.



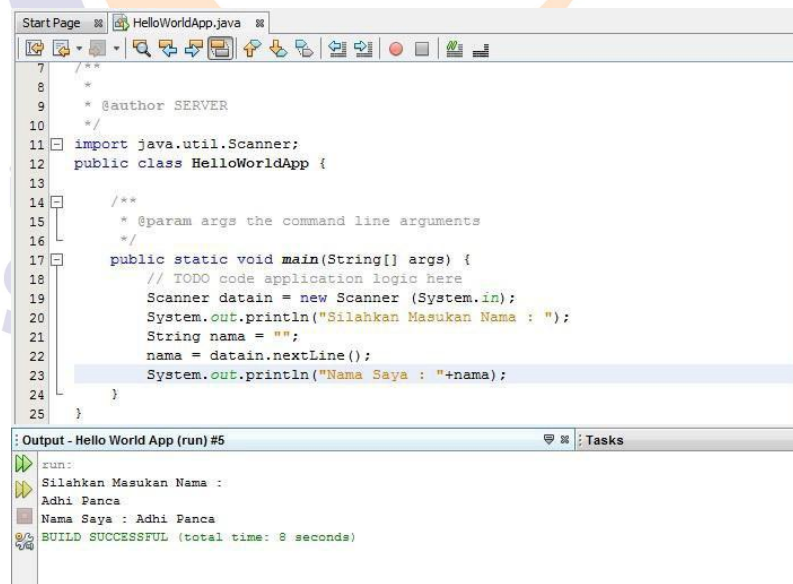


```
8
9  * @author SERVER
10 */
11
12 public class HelloWorldApp {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         String nama = "Adhi Panca";
20         int semester;
21         semester = 4 + 2 - 1 * 5 / 5;
22         System.out.println("Nama Saya : "+nama);
23         System.out.println("Saya Semester : "+semester);
24     }
25 }
26
```

Output - Hello World App (run) #5

```
run:
Nama Saya : Adhi Panca
Saya Semester : 5
BUILD SUCCESSFUL (total time: 1 second)
```

Pada Scanner memerlukan file bawaan java dimana kita harus mengimport file `java.util.Scanner`; tersebut diluar main class atau pada baris pertama. Seperti pada contoh berikut :



```
7
8
9  * @author SERVER
10 */
11 import java.util.Scanner;
12 public class HelloWorldApp {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         Scanner datain = new Scanner (System.in);
20         System.out.println("Silahkan Masukkan Nama : ");
21         String nama = "";
22         nama = datain.nextLine();
23         System.out.println("Nama Saya : "+nama);
24     }
25 }
```

Output - Hello World App (run) #5

```
run:
Silahkan Masukkan Nama :
Adhi Panca
Nama Saya : Adhi Panca
BUILD SUCCESSFUL (total time: 8 seconds)
```

#### 4. Penggunaan Variable dan Latihan

Variabel merupakan tempat atau wadah untuk menyimpan nilai / value pada bahasa pemrograman. Pada Pemrograman java, semua variabel harus

dideklarasikan sebelum mereka dapat digunakan. Bentuk dasar dari sebuah deklarasi variabel yang ditampilkan di sini:

type identifier [= value]

int a, b, c; // deklarasi 3 variabel a,b,c bertipe integer.

long d = 3, e, f = 5; // deklarasi 3 variabel d,e,f bertipe long

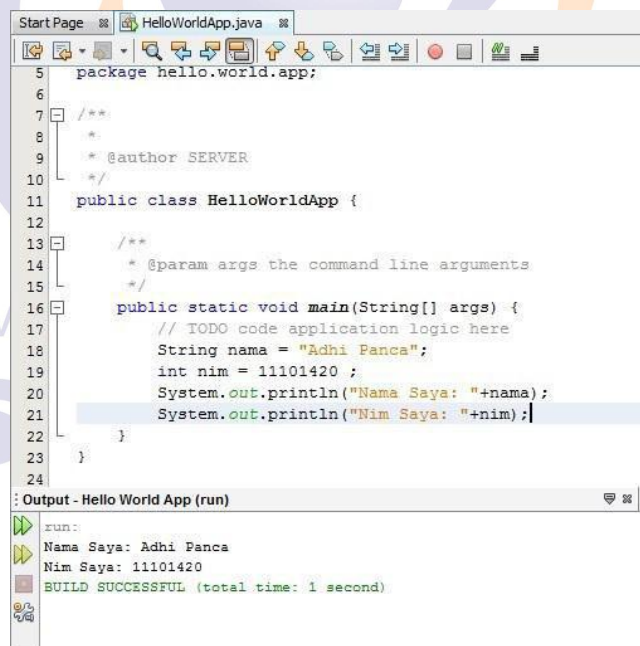
byte z = 22; // deklarasi dan inisialisasi variabel z. char

x = 'x'; // variabel x yang diberikan value / nilai x.

String nama = "adi"; // variabel nama yang diberikan value / nilai adi.

Jenis ini merupakan salah satu tipe data Java. Identifier adalah nama variabel. Menyatakan lebih dari satu variabel dari jenis tertentu, menggunakan daftar dipisahkan koma. Berikut adalah beberapa contoh deklarasi variabel dari berbagai jenis. Perhatikan bahwa beberapa mencakup inisialisasi.

Berikut merupakan contoh project dari variabel yang sudah di jelaskan.



```
5 package hello.world.app;
6
7 /**
8  *
9  * @author SERVER
10 */
11 public class HelloWorldApp {
12
13     /**
14      * @param args the command line arguments
15      */
16     public static void main(String[] args) {
17         // TODO code application logic here
18         String nama = "Adhi Panca";
19         int nim = 11101420;
20         System.out.println("Nama Saya: "+nama);
21         System.out.println("Nim Saya: "+nim);
22     }
23 }
24
```

Output - Hello World App (run)

```
run:
Nama Saya: Adhi Panca
Nim Saya: 11101420
BUILD SUCCESSFUL (total time: 1 second)
```

## 5. Penggunaan Method Atau Fungsi Dalam Java

Method adalah fungsi yang berada di dalam Class. Sebutan ini, biasanya digunakan pada OOP. Dengan menggunakan fungsi atau method, kode program yang ditulis akan lebih sederhana dan terstruktur rapi, sehingga saat adanya perubahan akan lebih mudah ditemukan dan diperbaiki. Selain itu,

menggunakan fungsi atau method dalam Java juga sangat bermanfaat untuk menulis kode program yang memproses hal yang sama. Proses tersebut tidak perlu ditulis berulang-ulang, cukup ditulis sekali di dalam sebuah fungsi atau method, selanjutnya hanya memanggil fungsi atau method tersebut di banyak proses. Hal tersebut dapat dilakukan karena Fungsi atau Method dalam Java dapat digunakan atau dipanggil lebih dari satu kali.

Setidaknya terdapat dua jenis Fungsi atau Method dalam Java, yaitu Method yang mengembalikan nilai menggunakan return type dan method yang tidak mengembalikan nilai menggunakan keyword **void**.

Format kode program untuk Fungsi atau Method return type dalam Java adalah sebagai berikut.

```
public static int namaMethod(int angka1, int angka2) {  
    // kode proses  
}
```

Contoh Program Mengembalikan Nilai:

```
public static int penjumlahanDuaAngka(int angka1, int angka2) {  
    int hasil = angka1 + angka2;  
    return hasil;  
}
```

#### **Keterangan :**

public static	: Modifier
int	: Return type untuk mengembalikan nilai
namaMethod	: Nama fungsi atau method yang diinginkan
angka1 dan angka2	: Nama variabel parameter
int angka1 dan int angka2	: List parameter (boleh satu aja atau lebih)

#### **Fungsi Static dan Non-Static**

Kata kunci static akan membuat fungsi dapat dieksekusi langsung, tanpa harus membuat instansiasi objek dari class.

```
public class FungsiStatic {  
    // Fungsi non-static
```

```

void makan(String makanan){
    System.out.println("Hi!");
    System.out.println("Saya sedang makan " + makanan);
}
// fungsi static
static void minum(String minuman){
    System.out.println("Saya sedang minum " + minuman);
}
// fungsi main
public static void main(String[] args) {
    // pemanggilan fungsi static
    minum("Kopi");
    // membuat instansiasi objek saya dari class FungsiStatic
    FungsiStatic saya = new FungsiStatic();
    // pemanggilan fungsi non-static
    saya.makan("Nasi Goreng");
}
}

```

Hasil output dari program di atas:

```

Saya sedang minum Kopi
Hi!
Saya sedang makan Nasi Goreng

```

### Pemanggilan Fungsi di Fungsi Lain

Fungsi-fungsi dapat saling memanggil untuk memproses data.

Contoh : sebuah program Kalkulator Bangun Ruang memiliki fungsi.

Fungsi : luasPersegi(), luasPersegiPanjang(), luasSegitiga(), luasBalok(), luasKubus()dsb. Fungsi-fungsi tersebut dapat saling membantu, contoh fungsi luasKubus() membutuhkan fungsi luasPersegi().

```

public class BangunRuang {

    public static void main(String[] args) {

```

```

        int s = 12;
        int luas = luasKubus(s);

        System.out.println(luas);
    }

    // membuat fungsi luasPersegi()
    static int luasPersegi(int sisi){
        return sisi * sisi;
    }

    // membuat fungsi luasKubus()
    static int luasKubus(int sisi){

        // memanggil fungsi luasPersegi
        return 6 * luasPersegi(sisi);
    }
}

```

**Hasil Output :**

864

### Overload Method

Ketika sebuah kelas memiliki dua method atau lebih dengan nama yang sama dan jumlah parameter yang berbeda, hal ini disebut overload method.

Contoh Program :

```

public class ContohOverload
{
    public static void main (String[] args)
    {
        int a = 11;
        int b = 6;
    }
}

```

```
double c = 7.3;
double d = 9.4;

int hasil fungsiMin (a, b);
//nama fungsi sama dengan parameter yang berbeda
double hasil2 = fungsiMin (c, d);

System.out.println ("Nilai minimum = " + hasil1);
System.out.println ("Nilai minimum = " + hasil2);
}
//untuk integer
public static int fungsiMin(int n1, int n2)
{
    int min;
    if (n1 > n2)
        min = n2;
    else
        min = n1;

    return min;
}
// untuk double
public static double fungsiMin (double n1, double n2)
{
    double min;
    if (n1 > n2)
        min = n2;
    else
        min = n1;
    return min;
}
}
```

### Hasil Output :

Nilai minimum = 6

Nilai minimum = 7.3

### C. Latihan

#### Percobaan I :

```
class IncDec {  
    public static void main (String args[]) {  
        int x = 8, y = 13;  
        System.out.println("x = " + x);  
        System.out.println("y = " + y);  
        System.out.println("x = " + ++x);  
        System.out.println("y = " + y++);  
        System.out.println("x = " + x--);  
        System.out.println("y = " + --y);  
    }  
}
```

#### Percobaan II :

```
class Bitwise {  
    public static void main (String args[]) {  
        int x = 5, y = 6;  
        System.out.println("x = " + x);  
        System.out.println("y = " + y);  
  
        System.out.println("x & y = " + (x & y));  
        System.out.println("x | y = " + (x | y));  
        System.out.println("x ^ y = " + (x ^ y));  
    }  
}
```

#### Percobaan III :

```
class BitwiseComplement {  
    public static void main (String args[]) {  
        int x = 8;  
  
        System.out.println("x = " + x);  
        int y = ~x;  
        System.out.println("y = " + y);  
    }  
}
```

#### Percobaan IV :

```
class LogicalOperator {  
    public static void main (String args[]) {  
        int x = 7, y = 11, z = 11;  
        System.out.println("x = " + x);  
        System.out.println("y = " + y);  
        System.out.println("z = " + z);  
  
        System.out.println("x < y = " + (x < y));  
        System.out.println("x > z = " + (x > z));  
        System.out.println("y <= z = " + (y <= z));  
        System.out.println("x >= y = " + (x >= y));  
    }  
}
```

#### D. Daftar Pustaka

1. Herbert Schildt, 2014, E-Book Java A Beginner Guide Sixth Edition : Create, Compile and Run Programs Today, by McGraw-Hill Education (Publisher).
2. Reges, Stuart; Stepp, Marty. 2016. Building Java Programs: A Back to Basic Approach 4th Edition. Pearson. <http://www.buildingjavaprograms.com/>
3. Java Platform Standard Edition Documentation.  
<http://docs.oracle.com/javase/8/docs/>

Universitas  
**Esa Unggul**