MATA KULIAH : STRUKTUR DATA
SESI PERTEMUAN : II (DUA)
MATERI : VARIABEL & TIPE DATA
DOSEN : ALUN SUJJADA, S.KOM., M.T

A. VARIABEL DAN TIPE DATA

Variabel merupakan sebuah *container* yang digunakan untuk menyimpan suatu nilai secara sementara pada sebuah program dengan tipe data tertentu. Sementara disini berarti nilai dalam variabel dapat diubah sesuai keperluan dan ketika program dihentikan, maka nilai dari variabel akan hilang. Variabel dapat dianalogikan sebagai box (kotak) untuk menyimpan barang. Setiap box(kotak) mempunyai jenis masing-masing sesuai dengan jenis barang yang dapat disimpan. Misal kotak amal digunakan untuk menyimpan uang, kotak lemari digunakan untuk menyimpan pakaian. Contoh hal-hal tersebut itulah yang dapat mengilustrasikan tentang konsep tipe data. Tipe data adalah suatu kelompok yang mempunyai jenis-jenis tertentu. Dengan kata lain, tipe data adalah sebuah cara yang digunakan untuk menentukan jenis suatu data tersebut, kata lain dari hal ini ialah "deklarasi variabel".

Pada dasarnya ada dua macam tipe data pada sebuah variabel yaitu tipe primitive dan tipe reference.

1. Tipe primitive meliputi:
    a. Tipe Boolean adalah tipe data logika yang hanya mempunyai 2 nilai yaitu *true* dan *false*.
    b. Tipe Numerik yang meliputi:
        - byte, adalah tipe data Integral 8-bit. Memiliki rentang nilai antara -27 sampai 27–1 atau dari -128 sampai 127
        - short, adalah tipe data Integral 16-bit. Memiliki rentang nilai antara -215 sampai 215–1 atau dari -32768 sampai 32768.
        - int, adalah tipe data Integral 32-bit. Memiliki rentang nilai antara -231 sampai 231–1 atau dari -2,147,483,648 sampai 2,147,483,647.
        - long, adalah tipe data Integral 64-bit. Memiliki rentang nilai antara -263 sampai 263 –1.
        - char, adalah tipe data Textual, yang merepresentasikan karakter unicode 16-bit. Nilai literalnya harus diapit dengan tanda petik tunggal (‘).
        - float, adalah tipe data Floating Point 32-bit. Nilai literalnya mengandung pecahan (dipisahkan dengan tanda titik ‘.’)
        - double, adalah tipe data Floating Point 64-bit. Nilai literal default untuk float dan double adalah double, kecuali diberi akhiran F

2. Tipe data *reference* meliputi
   a. Tipe class
   b. Tipe array
   c. Tipe interface

Untuk mendeklarasikan variable dan tipe data menggunakan sintaks **<tipe-data> <nama variable>**. Perhatikan contoh source code 2.1 berikut ini untuk memahami penggunaan tipe data dan variabel.

```
public class Variabel {

    public static void main(String[]args){
        byte   nilai_byte   = 70;
        short  nilai_short  = 500;
        int    nilai_int    = 263456;
        long   nilai_long   = 1200000;
        char   nilai_char   = 'A';
        float  nilai_float  = 190;
        double nilai_double = 387;
        boolean nilai_bool  = true;

        System.out.println(nilai_byte);
        System.out.println(nilai_short);
        System.out.println(nilai_int);
        System.out.println(nilai_long);
        System.out.println(nilai_char);
        System.out.println(nilai_float);
        System.out.println(nilai_double);
        System.out.println(nilai_bool);
    }
}
```

Source code 2.1 Penggunaan variabel dan tipe data

Berdasarkan *scope* atau ruang lingkupnya, maka variabel dibedakan menjadi 2 jenis yaitu:

1. *Member Variable / Class Variable*

   Yaitu variabel yang dimiliki oleh *class*, dimana deklarasinya diletakkan didalam *class*, sehingga variabelnya akan dapat diakses oleh semua *method* dan minimal didalam class yang mendefinisikannya.

2. *Local Variable*

   Yaitu variabel yang dimiliki oleh *method*, dimana deklarasinya diletakkan didalam *method*, sehingga variabelnya hanya dapat diakses didalam method yang mendeklarasikannya sendiri.

   Untuk lebih memperjelas pengetahuan tentang *member variable* dan *local variable*, perhatikanlah contoh gambar 2.1.

Gambar 2.1 *Member Variable dan Local Variable*

Pada *class* **Guru** terdapat *member variable* **nip**, **nama**, **gender** dan **alamat** dimana ruang lingkupnya dapat diakses disemua *method*, sehingga ketika variabel nama diakses oleh *method* **setTitelNamaGuruRPL**, maka kode program akan berjalan dengan benar. Sedangkan pada *local variable* **gaji3B** yang didefinisikan didalam method **setGajiGuru**, maka variabel tersebut tidak dapat diakses didalam *method* lain, terbukti ketika diakses didalam *method* **main** muncul garis merah yang menandakan kode program dalam keadaan error.

B. OPERATOR

Operator dalam bahasa pemrograman merupakan simbol yang dapat kita gunakan untuk melakukan suatu operasi. Contohnya, operasi matematika seperti perkalian, pembagian, pengurangan, dan penjumlahan. Selain itu, operator juga digunakan untuk memberikan nilai ke suatu variabel atau membandingkan dua buah nilai atau lebih. Operator adalah sebuah simbol yang digunakan untuk melakukan operasi tertentu. Misalnya penjumlahan nilai dari variabel x dan y, maka dapat menggunakan operator penjumlahan (+). Terdapat 6 (enam) jenis kelompok operator dalam pemrograman Java yaitu:

1. Operator Aritmatika

Operator yang digunakan untuk melakukan operasi perhitungan matematika seperti pada tabel 2.1

NO	Simbol	Fungsi
1.	+	Penjumlahan
2.	-	Pengurangan
3.	*	Perkalian
4.	/	Pembagian
5.	%	Sisa bagi

Tabel 2.1 Operator aritmatika

Contoh cara penggunaannya dapat dilihat pada source code 2.2 dan 2.3 berikut ini.

```
1  import java.util.Scanner;
2  public class OperatorAritmatika {
3      public static void main(String[] args) {
4          int bil_satu;
5          int bil_dua;
6          int hasilJumlah, hasilKurang, hasilKali, hasilBagi, hasilSisaBagi;
7
8          Scanner input = new Scanner(System.in);
9          System.out.print("Isikan Bilangan 1 : ");
10         bil_satu = input.nextInt();
11         System.out.print("Isikan Bilangan 2 : ");
12         bil_dua = input.nextInt();
13
14         hasilJumlah = bil_satu + bil_dua;
15         hasilKurang = bil_satu - bil_dua;
16         hasilKali = bil_satu * bil_dua;
17         hasilBagi = bil_satu / bil_dua;
18         hasilSisaBagi = bil_satu % bil_dua;
19
20         System.out.println("Hasil Penjumlahan : " + hasilJumlah);
21         System.out.println("Hasil Kurang : " + hasilKurang);
22         System.out.println("Hasil Kali : " + hasilKali);
23         System.out.println("Hasil Bagi : " + hasilBagi);
24         System.out.println("Hasil Sisa Bagi : " + hasilSisaBagi);
25     }
26 }
```

Source code 2.2 Penggunaan operator aritmatika

```java
import java.util.Scanner;
public class KelilingPersegi {
    public static void main(String[] args) {
        double keliling, panjang, lebar;

        Scanner input = new Scanner(System.in);
        System.out.print("Isikan panjang : ");
        panjang = input.nextInt();
        System.out.print("Isikan lebar : ");
        lebar = input.nextInt();

        keliling = (2 * panjang) + (2 * lebar);
        System.out.println("Keliling persegi panjang adalah " + keliling);
    }
}
```

Source code 2.3 Menghitung keliling persegi

2. Operator Penugasan

Operator yang digunakan untuk memberikan sebuah nilai kedalam variabel *(assignment operator)*. Seperti pada tabel 2.2

NO	Simbol	Fungsi
1.	=	Pengisian nilai
2.	-=	Pengisian dan pengurangan
3.	*=	Pengisian dan perkalian
4.	/=	Pengisian dan pembagian
5.	%=	Pengisian dan sisa bagi

Tabel 2.2 Operator Penugasan

Contoh cara penggunaannya dapat dilihat pada source code 2.4 berikut ini.

```java
public class OperatorPenugasan {
    public static void main(String[] args) {
        // memberikan nilai 50 ke variabel mynumber
        double mynumber = 50;
        System.out.println("Awalnya variabel mynumber bernilai " + mynumber);

        // menambahkan nilai yang ada pada variabel mynumber dengan 50
        mynumber +=50;
        System.out.println("Setelah ditambahkan dengan 50, maka variabel mynumber bernilai " + mynumber);

        mynumber -=22;
        System.out.println("Tetapi setelah dikurangi dengan 22, maka variabel mynumber bernilai " + mynumber);

        mynumber *=2;
        System.out.println("Kemudian dikalikan dengan 2, maka variabel mynumber bernilai " + mynumber);

        mynumber /=4;
        System.out.println("Berikutnya dibagi dengan 4, maka variabel mynumber bernilai " + mynumber);

        mynumber %=2;
        System.out.println("Setelah di modulus dengan 2, maka variabel mynumber bernilai " + mynumber);
    }
}
```

Source code 2.4 Penggunaan operator penugasan

3. Operator Perbandingan

Sesuai dengan namanya, operator tersebut digunakan untuk membandingkan antara beberapa kondisi. Nilai yang dihasilkan adalah boolean yaitu berupa **true** atau **false** seperti pada tabel 2.3 berikut ini.

NO	Simbol	Fungsi
1.	>	Lebih dari
2.	<	Kurang dari
3.	>=	Lebih dari atau sama dengan
4.	<=	Kurang dari atau sama dengan
5.	==	Sama dengan
6.	!=	Tidak sama dengan

Tabel 2.3 Operator Perbandingan

Untuk lebih jelasnya, dapat dilihat implementasi pada source code 2.5 berikut ini.

```
public class OperatorPerbandingan {

    public static void main(String[] args) {
        int nilai_awal = 80;
        int nilai_akhir = 70;
        boolean isTerbesar;

        isTerbesar = nilai_awal > nilai_akhir;
        System.out.println("Apakah " + nilai_awal + " lebih besar dari " + nilai_akhir + "? " + isTerbesar);

        isTerbesar = nilai_awal < nilai_akhir;
        System.out.println("Apakah " + nilai_awal + " lebih kecil dari " + nilai_akhir + "? " + isTerbesar);

    }
}
```

Source code 2.5 Penggunaan operator perbandingan

4. Operator Logika

Operator logika adalah operator yang mempunyai nilai kebenaran sesuai dengan kaidah matematis. Beberapa operator yang dapat digunakan dalam pemrograman java adalah:

NO	Simbol	Fungsi
1.	&&	Logika And (dan)
2.	||	Logika Or (atau)
3.	!	Logika Not / Negasi (kebalikan)

Tabel 2.4 Operator Logika

Sebagai contoh permasalahan yang diselesaikan menggunakan operator logika adalah:

Contoh 1 :

Kondisi 1 : Cuaca cerah

Kondisi 2 : Kondisi badan sedang sehat

Pernyataan : Jika cuaca cerah **dan** kondisi badan sedang sehat, maka saya akan pergi berbelanja ke pasar".

Hasil : Pergi berbelanja ke pasar

Contoh 2 :

Kondisi 1 : Cuaca Mendung

Kondisi 2 : Kondisi badan sedang sehat

Pernyataan : Jika cuaca cerah dan kondisi badan sedang sehat, maka saya akan pergi berbelanja ke pasar".

Hasil : Tidak Pergi berbelanja ke pasar.

Contoh 3 :

Kondisi 1 : Cuaca Mendung

Kondisi 2 : Kondisi badan sedang sehat

Pernyataan : Jika cuaca cerah atau kondisi badan sedang sehat, maka saya akan pergi berbelanja ke pasar".

Hasil : Pergi berbelanja ke pasar

Jadi operator **and** akan bernilai **true**, jika semua kondisi bernilai **true.** Sedangkan operator **or** akan bernilai **true** jika salah satu kondisinya bernilai **true.** Berikut ini adalah tabel kebenaran dari masing-masing operator logika.

No	Kondisi 1	Kondisi 2	Hasil
1.	True	True	True
2.	True	False	False
3.	False	True	False
4.	False	False	False

Tabel 2.5 Tabel kebenaran operator AND

No	Kondisi 1	Kondisi 2	Hasil
1.	True	True	True
2.	True	False	True
3.	False	True	True
4.	False	False	False

Tabel 2.6 Tabel kebenaran operator OR

No	Kondisi	Hasil
1.	True	False
2.	False	True

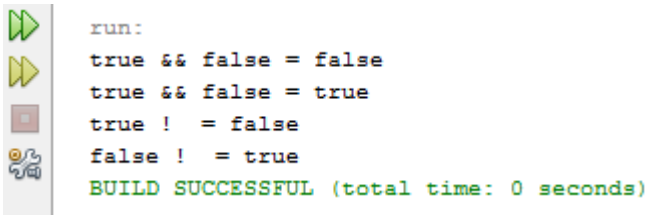
Tabel 2.7 Tabel kebenaran operator Not

Contoh penggunaan pada program dapat dilihat pada source code 2.6 berikut ini.

```
public class OperatorLogika {

    public static void main(String[] args) {
        boolean data_1 = true;
        boolean data_2 = false;
        boolean result;
        result = data_1  && data_2;
        System.out.println(data_1 + " && " + data_2 + " = " + result);

         result = data_1  || data_2;
        System.out.println(data_1 + " && " + data_2 + " = " + result);

         result = !data_1 ;
        System.out.println(data_1 + " ! " +  " = " + result);

        result = !data_2 ;
        System.out.println(data_2 + " ! " +  " = " + result);
    }
}
```

Source code 2.6 Penggunaan operator perbandingan

Jika program tersebut dijalankan, maka hasil *output* seperti pada gambar 2.2



Gambar 2.2 Hasil *output* program logika

5. Operator Bitwise

Operator bitwise merupkan operator yang digunakan untuk operasi bit (biner). Operator bitwise terdiri dari:

NO	Simbol	Fungsi
1.	&	AND
2.	|	OR
3.	^	XOR
4.	~	Negasi
5.	<<	Left Shift
6.	>>	Right Shift
7.	<<<	Left shift unsigned
8.	>>>	Right shift unsigned

Tabel 2.8 Operator Bitwise

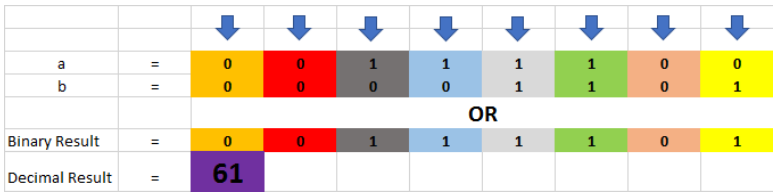
Operator ini berlaku untuk tipe data **int**, **long**, **short**, **char**, dan **byte**. Operator ini akan menghitung dari bit-ke-bit. Misalnya, kita punya variabel **a = 60** dan **b = 13**. Bila dibuat dalam bentuk biner, akan menjadi seperti ini:

a = 00111100

b = 00001101

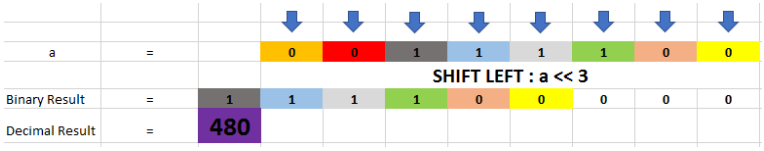
Kemudian dilakukan operator bitwise, hasilnya adalah 12 seperti pada gambar berikut ini.

Gambar 2.3 Ilustrasi perhitungan operator bitwise AND



Gambar 2.4 Ilustrasi perhitungan operator bitwise OR

Gambar 2.5 Ilustrasi perhitungan operator bitwise XOR



Gambar 2.6 Ilustrasi perhitungan operator bitwise SHIFT LEFT

6. Operator Ternary

   Opertor ini unik, seperti membuat pertanyaan. Simbolnya menggunakan tanda tanya (?) dan titik-dua (:) untuk memisah jawabannya. Gambar 2.7 berikut ini adalah ilustrasi tentang pemahaman operator ternary

Gambar 2.7 Ilustrasi operator ternary

Pada contoh ilustrasi pada gambar 2.7, “Kamu suka aku” adalah pertanyaan atau kondisi yang akan diperiksa. Jika jawabannya benar, maka hasilnya adalah **iya**. Sebaliknya hasilnya adalah **tidak**. Untuk lebih jelasnya, lihatlah contoh implementasi pada source code 2.7 berikut ini.

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

        int number = 10;

        String jawab = number > 20 ? number + " Lebih besar dari 20" : number + " lebih kecil dari 20";

        System.out.println(jawab);
    }
}
```

Source code 2.7 Penggunaan operator ternary

Hasil *output* ketika program tersebut dijalankan adalah **"10 lebih kecil dari 20"**