

INPUT & OUTPUT

Fundamen Pengembangan Aplikasi



VARIABEL

- **Variabel**
 - Lokasi pada memori untuk menyimpan data sementara (wadah data)
- **Kriteria Variabel:**
 - Harus memiliki nama yang unik
 - Harus memiliki tipe data
 - Bisa menyimpan nilai tertentu
 - Dapat melakukan operasi

VARIABEL

Gelas



Kosong

Gelas1



Susu

Gelas2



Kopi

Gelas3



Sirup

Gelas1, Gelas2, Gelas3 → **Nama**
Susu, Kopi, dan Sirup → **Nilai**

TIPE DATA

- **Type Data**
 - Klasifikasi jenis dari data yang kita ingin simpan dalam sebuah variabel
- Setiap tipe data memiliki **jenis data** yang dapat disimpan serta **operasi apa saja** yang bisa dilakukan terhadap tipe data tersebut

TIPE DATA

Gelas



Minuman

Mangkuk



**Makanan
Berkuah**

Piring



**Makanan
Tidak Berkuah**

Lepek



**Makanan
Ringan**

Makanan berkuah tidak bisa diletakkan di lepek \Rightarrow **Jenis Data**
Makanan tidak berkuah tidak bisa disedot (sedotan) \Rightarrow **Operasi**

TIPE DATA di JAVA

Tipe Data Primitive

- **boolean** (true/false)
- **byte** (-128 ... 127)
- **short** (-32768 ... 32767)
- **int** ($-2^{31} \dots 2^{31} - 1$)
- **long** ($-2^{63} \dots 2^{63} - 1$)
- **float** (32 bit floating point)
- **double** (64 bit floating point)
- **char** (16 bit Unicode)

Tipe Data Reference

- **String**
- **ArrayList**

Ciri-Ciri Tipe Data Reference

- Huruf awal ditulis besar
- Gabungan beberapa tipe data primitive
- Tipe data bentukan

DEKLARASI VARIABEL

(penentuan tipe data suatu variabel)

<tipe data> <nama>;

boolean jalan1;

byte umur;

short tahun_lahir;

int \$harga;

long _total;

double hasilAkhir;

char kelas;

String nama_asli;

Aturan Penamaan Variabel

- Bisa terdiri dari huruf, angka, dan garis bawah (_)
- Karakter pertama **tidak boleh** angka
- **Boleh** diawali huruf abjad, underscore (_), atau karakter \$
- Tidak boleh mengandung spasi
- Tidak boleh menggunakan *reserved word* di Java
- Tidak diperbolehkan untuk menggunakan simbol-simbol operator
- Bisa menggunakan teknik **CamelCase** untuk menggantikan spasi (jeda pembacaan)

INISIALISASI VARIABEL

(pengisian nilai awal suatu variabel)

```
<type data> <nama> = <nilai>;
```

//Alternatif Lain

```
boolean jalan1 = true;
```

```
byte umur = 22;
```

```
short tahun_lahir = 2001;
```

```
int harga = 125000;
```

```
long total = 30000000000L;
```

```
double hasilAkhir = 98.715;
```

```
char kelas = 'a';
```

```
String namaAsli = "Agus";
```

```
byte umur;
```

```
umur = 24;
```

```
String nama;
```

```
nama = "Saya";
```

```
int a, b;
```

```
a = 20;
```

```
b = 21;
```


KONSTANTA

- Variabel yang nilainya bersifat **konstan** (tidak dapat berubah)
- Secara kaidah penamaan, nama dari suatu konstanta *idealnya* ditulis dalam **huruf besar semua**
- Untuk membuat konstanta kita cukup menambahkan kata kunci *(keyword)* **final** di bagian awal

```
final double PI = 3.14159;
```

```
final int PEMBAGI = 7;
```

KOMENTAR

- Komentar adalah baris perintah yang tidak dieksekusi oleh program
- Komentar dapat dimanfaatkan untuk memberikan informasi tambahan pada baris program yang kita buat
- Komentar satu baris cukup diawali dengan `//`
`// Ini tidak akan dieksekusi`
- Komentar lebih dari satu baris diawali dengan `/*` dan diakhiri dengan `*/`
`/* Komentar baris pertama`
`Ini komentar juga, tapi baris kedua */`

DEBUGGING



DEBUGGING

- Debugging adalah metode untuk mencari dan mengurangi **bug** (kesalahan) pada suatu program
- Nilai variabel ketika suatu baris program dieksekusi **dapat dilihat**, sehingga letak kesalahan program dapat ditemukan
- Pada VS Code, proses debugging dapat dilakukan dengan menentukan *breakpoint* sesuai dengan kebutuhan

File Edit Selection View Go Run Terminal ... Hello.java - myHello - Visual Studio Code

RUN AND D... No Corv

VARIABLES

Local

args: String[0]@8
a: 2
b: 3

WATCH

CALL STACK

Thread [Common-Clean... RUNNING
Thread [Attach Listener] RUNNING
Thread [Signal Dispatch... RUNNING
Thread [Finalizer] RUNNING
Thread [Reference Hand... RUNNING
Thread [mai... PAUSED ON BREAKPOINT
Hello.main(String[]) Hello...

BREAKPOINTS

☐ Uncaught Exceptions
☐ Caught Exceptions
☒ Hello.java myHello\src

myHello > src > Hello.java > Hello > main(String[])

```
1 public class Hello {  
2     Run | Debug  
3     public static void main(String[] args) throws Exception { args = String  
4         int a, b, c; a = 2, b = 3  
5         a = 2; a = 2  
6         b = 3; b = 3  
7         c = a + b; a = 2, b = 3  
8         System.out.println(c);  
9     }  
10 }
```

breakpoint

tombol debugging

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Debug: Hello

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS D:\CODING\myHello> & 'C:\Program Files\Java\jdk-9.0.4\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:50855' '-cp' 'C:\Users\Acer\AppData\Roaming\Code\User\workspaceStorage\84a59ee8458a36684f299b28d359cff0\redhat.java\jdt_ws\myHello_37afaf9a\bin' 'Hello'

Ln 6, Col 1 Spaces: 4 UTF-8 CRLF {} Java

DEBUGGING

File Edit Selection View Go Run Terminal ... Hello.java - myHello - Visual Studio Code

RUN AND D... No Cor ▾ ⚙ ...

VARIABLES

Local

args: String[0]@8

a: 2

b: 3

c: 5

WATCH

CALL STACK

Thread [Attach Listener] RUNNING

Thread [Signal Dispatch...] RUNNING

Thread [Finalizer] RUNNING

Thread [Reference Hand...] RUNNING

Thread [mai... PAUSED ON BREAKPOINT

Hello.main(String[]) Hello...

BREAKPOINTS

☐ Uncaught Exceptions

☐ Caught Exceptions

☒ Hello.java myHello\src

myHello > src > Hello.java > Hello >

```
1 public class Hello {
2     Run | Debug
3     public static void main(String[] args) throws Exception { args = String
4         int a, b, c; a = 2, b = 3, c = 5
5         a = 2; a = 2
6         b = 3; b = 3
7         c = a + b; c = 5, a = 2, b = 3
8         System.out.println(c); c = 5
9     }
10 }
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** Debug: Hello + ▾

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS D:\CODING\myHello> & 'C:\Program Files\Java\jdk-9.0.4\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:50921' '-cp' 'C:\Users\Acer\AppData\Roaming\Code\User\workspaceStorage\84a59ee8458a36684f299b28d359cff0\redhat.java\jdt_ws\myHello_37afaf9a\bin' 'Hello'

Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Java

DEBUGGING

- Hasil ketika dijalankan (break point berada di line 7)
- Dapat dilihat masing-masing nilai variabelnya

LATIHAN

J App.java X

myHello > src > J App.java > ...

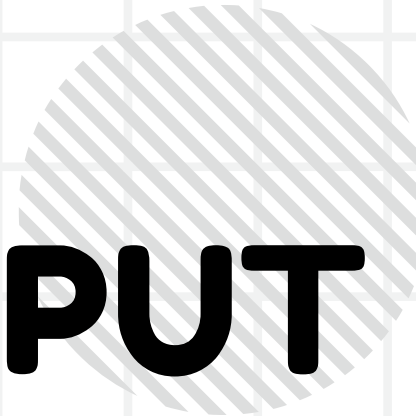
1 public class App {
2 Run | Debug
3 public static void main(String[] args) throws Exception {
4 int a, b, c;
5 a = 2;
6 b = 3;
7 c = a + b;
8 a = 5;
9 b = 7;
10 System.out.println(c);
11 }
12 }

⋮ ▶ ↺ ⬇ ⬆ ↻ □ ⚡

Gunakan tombol "Step Into" untuk berpindah ke breakpoint selanjutnya

Berapa nilai variabel **a**, **b**, dan **c** di masing-masing breakpoint?

INPUT & OUTPUT



Menampilkan Output

- Java Basic Output

System.out.print(<nilai>);

Menampilkan tanpa enter (tidak pindah baris)

System.out.println(<nilai>);

Menampilkan dengan enter (pindah baris)

System.out.printf(<nilai>);

Menampilkan dengan format khusus

Menampilkan Output

```
public class Output {  
    public static void main(String[] args) {  
  
        System.out.print("Tanpa pindah baris");  
        System.out.println("Pindah baris");  
        System.out.printf("|%-10s|    %S    |%n", "PATTERN", "result");  
    }  
}
```

```
Tanpa pindah barisPindah baris  
|PATTERN |    RESULT |  
PS D:\CODING JAVA\HelloWorld\HelloWorld>
```

Membaca Input

- Menggunakan library

```
import java.util.Scanner;
```

- Mendefinisikan objek kelas Scanner

```
Scanner <nama_objek> = new Scanner(System.in);
```

- Mendapatkan masukan dan menyimpan masukan

```
String nama = <nama_objek>.nextLine();  
int umur = <nama_objek>.nextInt();  
double ipk = <nama_objek>.nextDouble();
```

Membaca Input

```
import java.util.Scanner;

public class Input {
    public static void main(String[] args) {
        Scanner pemindai = new Scanner(System.in);

        System.out.print("Masukkan nama Anda: ");
        String nama = pemindai.nextLine();

        System.out.print("Masukkan umur Anda: ");
        int umur = pemindai.nextInt();

        System.out.println("Nama Anda "+nama+" dan umur Anda "+umur);
    }
}
```

Dimasukkan
oleh pengguna

Masukkan nama Anda: Agus
Masukkan umur Anda: 23
Nama Anda Agus dan umur Anda 23
PS D:\CODING JAVA\HelloWorld\HelloWorld>

pemanggilan
variabel

Assignment Operator

=

+=

-=

*=

/=

%=

- Digunakan untuk *mengisikan nilai* ke dalam variabel.

- Contoh:

```
int nilai;
```

```
nilai = 10;    //Saat ini nilainya 10
```

```
nilai += 20;   //Saat ini nilainya 30 (10 + 20)
```

```
nilai *= 10;   //Saat ini nilainya 300 (30 * 10)
```

Arithmetics Operator

+

-

*

/

%

- Digunakan untuk melakukan operasi aritmatika.

- Contoh:

```
int nilai;
```

```
nilai = 2 + 5; //Hasilnya 7
```

```
nilai = 2 * 7; //Hasilnya 14
```

```
nilai = 7 / 2; //Hasilnya 3 (karena bilangan bulat)
```

```
nilai = 7 % 2; //Hasilnya 1 (siswa hasil bagi)
```

Unary Operator

+

-

++

--

!

- Digunakan untuk melakukan operasi pada **satu operand** saja.
- Contoh:

```
int nilai = -5;           //Nilainya minus
boolean benar = !false;   //Nilainya true (not false)

int i = 3;
int a = i++;               // a = 3, i = 4 (post-increment)
int b = ++a;               // b = 4, a = 4 (pre-increment)
```

Equality and Relational Operator

`==`

`!=`

`>`

`<`

`>=`

`<=`

- Digunakan untuk membandingkan nilai (hasil akhir akan bernilai true/false).

- Contoh:

<code>(4 == 5)</code>	<code>//Hasilnya false</code>
<code>('g' != 'p')</code>	<code>//Hasilnya true</code>
<code>(4 > 4)</code>	<code>//Hasilnya false</code>
<code>(4 >= 4)</code>	<code>//Hasilnya true</code>

Logical Operator

|| &&

- Digunakan untuk membandingkan kondisi (hasil akhir akan bernilai true/false).

- Contoh:

<code>(true false)</code>	<code>//Hasilnya true</code>
<code>(true && false)</code>	<code>//Hasilnya false</code>
<code>(4 > 4 5 > 4)</code>	<code>//Hasilnya true</code>
<code>(4 >= 4 && 5 >= 4)</code>	<code>//Hasilnya true</code>

Perintah Pindah Baris

- Untuk berpindah baris, kita dapat menuliskan nilai `\n` (perlu diperhatikan karakter tersebut adalah *backslash* buka slash dan hanya menggunakan huruf `n` kecil)
- Tuliskan perintah berikut:

```
System.out.println("Jurusan\nInformatika");
```

- Apa yang muncul pada Panel Output?

TERIMA KASIH

