Algoritma dan Pemrograman

Leon Andretti Abdillah

03

Variables and Data Types

Identifier 1/2

- Identifier merupakan suatu nama variable sederhana yang didefinisikan sebagai kontainer nilai. Jenis nilai yang disimpan oleh identifier didefinisikan oleh special java keyword dikenal sebagai tipe data sederhana (primitive data type).
- Dalam pemrograman Java identifier dapat digunakan untuk menyatakan **variabel, konstanta, class, method, parameter**.
- *Identifier* dapat berupa sembarang symbolic name yang merujuk ke sesuatu pada suatu Java program.
- Identifier dapat diawali dengan letter, an underscore (_), or a Unicode currency symbol (e.g., $, £, ¥). Inisial letter ini dapat diikuti oleh sejumlah letters, digits, underscores, atau currency symbols.

Identifier 2/2

- Identifiers dapat berisi numbers, tapi tidak dapat dimulai dengan suatu number. Tambahan, identifiers tidak dapat berisi punctuation characters apa saja selain underscores dan currency characters.
- Secara convention, dollar signs dan currency characters lain disiapkan (are reserved) untuk identifiers secara otomatis dihasilkan oleh compiler atau semacam code preprocessor. Sebaiknya hindari penggunaanya dalam identifiers anda.

Java Language Keywords or Reserved Words

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

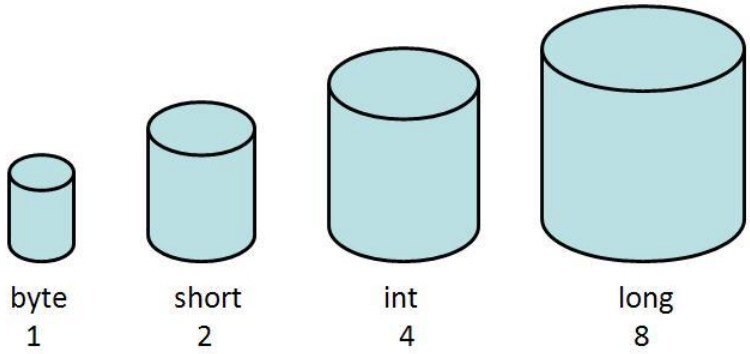
*	not used
**	added in 1.2
***	added in 1.4
****	added in 5.0

Primitive Data Types

- Tipe data primitif merupakan tipe data dasar yang dikenal oleh Java.
- Bahasa pemrograman Java merupakan statically-typed, yang berarti bahwa semua variables harus dideklarasikan terlebih dahulu sebelum mereka bisa digunakan. Ini melibatkan pernyataan jenis variabel dan namanya, contoh:
  - `int gear;`
  - `int gear = 1;`

Data types 1/2

- **Integer data types**
  1. **`byte`** (1 byte or 8 bits)
  2. **`short`** (2 bytes or 16 its)
  3. **`int`** (4 bytes or 32 bits)
  4. **`long`** (8 bytes or 64 bits)



- **Floating data types**: Real numbers in Java are represented with the float and double data types
  5. **`float`** (4 bytes)
  6. **`double`** (8 bytes)

Data types 2/2

Textual data types

7. **`Char`**. Tipe data char digunakan untuk menangani data berupa karakter-karakter ASCII. Tipe data char ditandai dengan penggunaan tanda kutip tunggal. Contoh tipe data char adalah: 'a', 'B', '4', dan lain sebagainya.

Logical data types

8. **`Boolean`**. Tipe data boolean digunakan untuk menentukan nilai benar atau salah. Oleh karena itu boolean hanya terdiri atas dua nilai, yaitu **`True`** dan **`False`**. Tipe data ini biasanya digunakan pada operasi logika.

String Tipe data string digunakan untuk menangani data berupa untaian beberapa karakter yang diistilahkan dengan string. Tipe data string ditandai dengan penggunaan tanda kutip ganda yang melingkupi data string. Contoh data tipe string adalah "Hello World".

String

- Sebagai tambahan dari delapan daftar primitive data types di atas, Java programming language juga menyediakan special support untuk character strings
  melalui java.lang.String class.
- Gunakan double quotes (“) untuk string anda, akan otomtis menciptakan String object baru; contoh, String s = "this is a string";. String objects bersifat *immutable*, yang berarti bahwa seklai diciptakan, nilainya tidak bisa diubah.
- Class String secara teknis bukanlah primitive data type, tetapi banyak beranggapan iya.

Data types summary 1/2

No	Data Types	Description	Size/formats	Contains
1	byte	Byte-length integer	8-bit two's complement	Signed integer
2	short	Short integer	16-bit two's complement	Signed integer
3	int	Integer	32-bit two's complement	Signed integer
4	long	Long integer	64-bit two's complement	Signed integer
5	float	Single-precision floating point	32-bit IEEE 754	IEEE 754 floating point
6	double	Double-precision floating point	64-bit IEEE 754	IEEE 754 floating point
7	char	One character	16 bits	Unicode character
8	boolean	Logical	1 bit	true or false

Data types summary 2/2

No	Data Types	Default value	Min value	Max value
1	byte	0	-128	127 (inclusive)
2	short	0	-32,768	32,767 (inclusive)
3	int	0	-2,147,483,648	2,147,483,647 (inclusive)
4	long	0L	-9,223,372,036,854,775,808	9,223,372,036,854,775,807 (inclusive)
5	float	0.0f	±1.4E-45	±3.4028235E+38
6	double	0.0d	±4.9E-324	±1.7976931348623157E+308
7	char	'\u0000'	'\u0000' (or 0)	'\uffff' (or 65,535 inclusive)
8	boolean	false	-	-

Variable (variabel) 1/2

- Variabel merupakan lokasi penyimpanan yang ada di memori. Setiap variabel memiliki kemampuan menyimpan suatu informasi sesuai dengan tipe data yang dideklarasikan untuk variabel tersebut saja.
- Suatu variable dapat dianggap sebagai suatu wadah/kontainer yang menampung nilai untuk anda selama program anda aktif. Setiap variable
- Setiap variable diberi **data type** tertentu yang menunjuk jenis dan kuantiti atas nilai yang ditampungnya.

Variable (variabel) 2/2

- Sintaks pendeklarasian variabel baru secara umum adalah sebagai berikut:
  - **`Data-type variable-name;`**
- Tipe-data meliputi semua tipe data yang dikenal oleh Java, sedangkan nama-variabel adalah identifier yang akan digunakan untuk merujuk ke variabel tersebut di dalam program.
- Contoh code:
  - **`int counter;`**
- Code di atas mendeklarasikan suatu variabel yang bernama **`counter`** dengan tipe data **`int`**.

Naming 1/3

Aturan2 dan konvensi2 penamaan variables:

1. Variable names are case-sensitive.
   a) A variable's name can be any legal identifier — an unlimited-length sequence of Unicode letters and digits, beginning with a letter, the dollar sign "$", or the underscore character "_".
   b) The convention, however, is to always begin your variable names with a letter, not "$" or "_".
   c) Additionally, the dollar sign character, by convention, is never used at all. You may find some situations where auto-generated names will contain the dollar sign, but your variable names should always avoid using it.
   d) A similar convention exists for the underscore character; while it's technically legal to begin your variable's name with "_", this practice is discouraged. White space is not permitted.

Naming 2/3

2. Subsequent characters may be letters, digits, dollar signs, or underscore characters.
   a) Conventions (and common sense) apply to this rule as well.
   b) When choosing a name for your variables, use full words instead of cryptic abbreviations (singkatan2samar).
   c) Doing so will make your code easier to read and understand.
   d) In many cases it will also make your code self-documenting; fields named cadence, speed, and gear, for example, are much more intuitive than abbreviated versions, such as s, c, and g.
   e) Also keep in mind that the name you choose must not be a keyword or reserved word.

Naming 3/3

3. If the name you choose consists of only one word, spell that word in all lowercase letters.
    a) If it consists of more than one word, capitalize the first letter of each subsequent word. The names gearRatio and currentGear are prime examples of this convention.
    b) If your variable stores a constant value, such as static final int NUM_GEARS = 6, the convention changes slightly, capitalizing every letter and separating subsequent words with the underscore character.
    c) By convention, the underscore character is never used elsewhere.

Java variable names conventions

- In general:
  1. always **start** the name with a **lower case** letter (the first character *has* to be a letter);
  2. **omit *the* and *a*** (and usually *of* if the meaning is clear);
  3. **capitalize the first letter of every word** that your variable name is made up of (as in our example)— except that you should **NEVER capitalize the first letter of the variable name in Java**;
  4. **don't use accented characters** or other "silly" characters such as the yen symbol in variable names, even though strictly speaking you may be allowed to (this can just cause problems with character encoding and/or when your colleague isn't used to typing accents);
  5. use a couple of common abbreviations:
     1. **no** = *number*
     2. **ix** = *index*
  6. **don't put underscores** between the words making up your variable name— there are certain *types* of variable where this is the convention, but for ***general* variables, it's not the convention**.

Java variable names conventions

- A name will start with a letter in lowercase. Examples are age, f4, name, g_14, country
- When a name is a combination of words, only the first name will start in lowercase. Examples are firstName, dateOfBirth, pi_314159
- When the name is an abbreviation, we will use uppercase on all characters. Examples are EAU, UN, CIA, NSA

Java Language Keywords

- Here is a list of keywords in the Java programming language. You cannot use any of the following as identifiers in your programs. The keywords `const` and `goto` are reserved, even though they are not currently used. `true`, `false`, and `null` might seem like keywords, but they are actually literals; you cannot use them as identifiers in your programs.

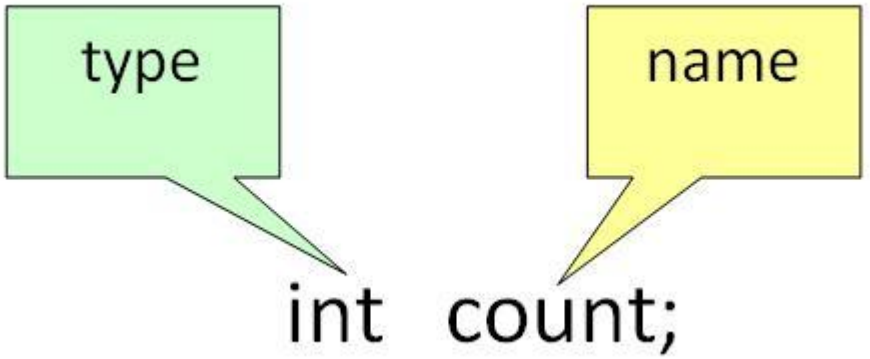
Two Steps to Making A Variable 1/3

- There are two steps to creating a variable;
  - Declaration, and
  - Initialization.
- Declaration is creating a name and saying what type of variable it names:

Two Steps to Making A Variable 2/3

- Declaration is creating a name and saying what type of variable it names:

- ```
  int count;
  String name;
  ```


  ```

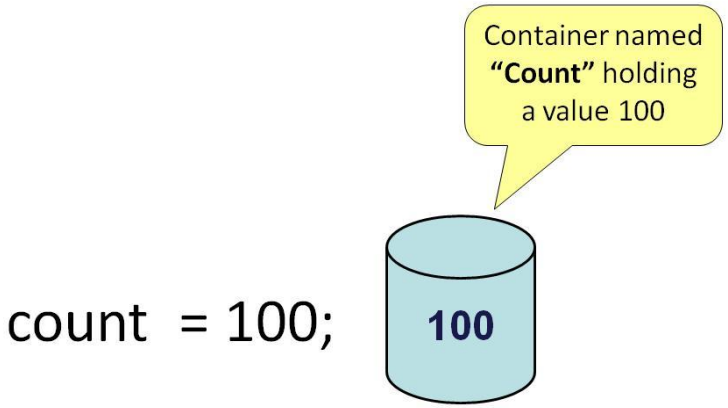
Two Steps to Making A Variable 3/3

- Initialization. Kita beri nilai awal untuk tiapvariable yang dideklarasikan

Contoh:

```
count=100;
name="";
```

We initialized count to 100, name to an empty string.

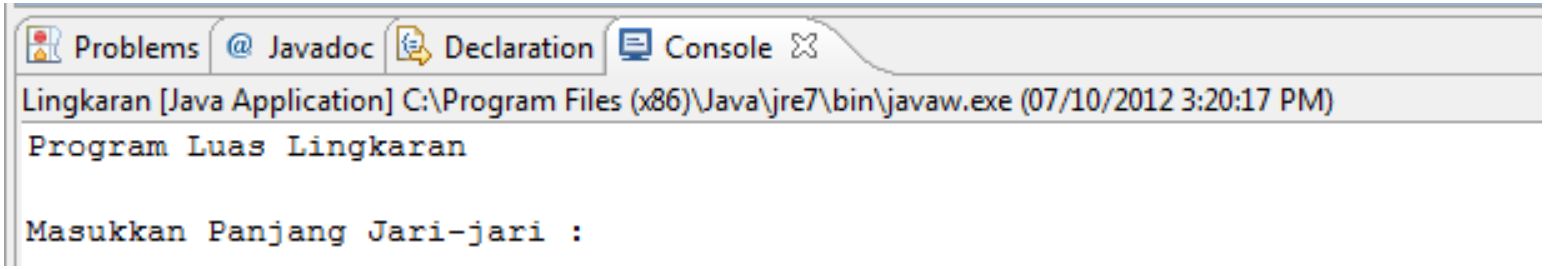


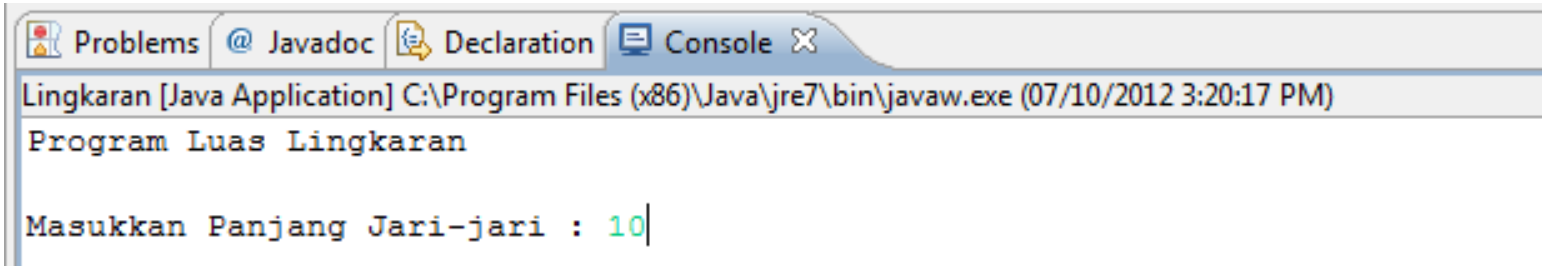
Two Steps in One

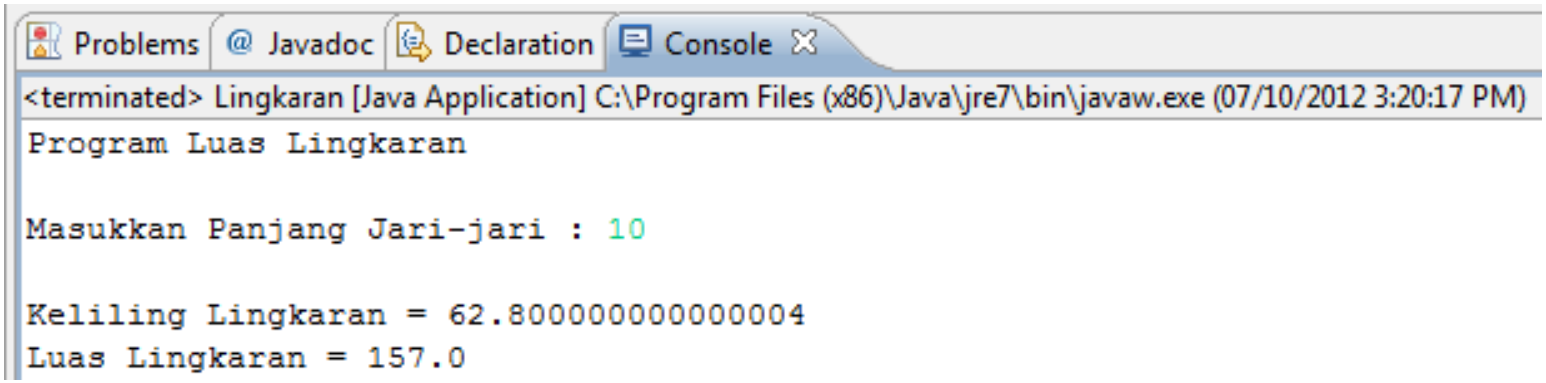
- You can declare and initialize a variable in one statement. But you still have to do both.*
-

```
int count=0;
String name="";
Scanner input=new  Scanner(System.in);
```

```java
package Package01;

import java.util.Scanner;

public class Lingkaran {

    /**
     * @param args
     */
    public static void main(String[] args) {

        double phi = 3.14;
        double r, luas, keliling;
        Scanner input = new Scanner(System.in);

        System.out.println("Program Luas Lingkaran\n");
        System.out.print("Masukkan Panjang Jari-jari : ");
        // input
        r = input.nextDouble();

        // process
        keliling = 2 * phi * r;
        luas = 0.5 * phi * r * r;

        // ouput
        System.out.print("\nKeliling Lingkaran = " + keliling + "\n");
        System.out.print("Luas Lingkaran = " + luas);
    }
```


Problems
Javadoc
Declaration
Console
Lingkaran [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (07/10/2012 3:20:17 PM)
Program Luas Lingkaran
Masukkan Panjang Jari-jari :


Problems
Javadoc
Declaration
Console
Lingkaran [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (07/10/2012 3:20:17 PM)
Program Luas Lingkaran
Masukkan Panjang Jari-jari : 10


Problems
Javadoc
Declaration
Console
<terminated> Lingkaran [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (07/10/2012 3:20:17 PM)
Program Luas Lingkaran
Masukkan Panjang Jari-jari : 10
Keliling Lingkaran = 62.800000000000004
Luas Lingkaran = 157.0

Data Input 1/3

- When you type a value in a program, to retrieve it, you can the **in** object of the **System** package:
  - **`System.in`**
- After getting that value, you must first store it somewhere. One of the classes you can use is called **Scanner**. Before using the Scanner class, you must import the **java.util.Scanner** package into your program. This would be done by writing the following in the top section of the file:
  - **`import java.util.Scanner;`**

Data Input 2/3

- To use the **Scanner** class to retrieve a value, use the following formula:
  - `Scanner VariableName = new Scanner(System.in);`
- The only think we need to mention at this time is that, after the **Scanner** class, you must give a variable name. An example would be:
  - `Scanner scnr = new Scanner(System.in);`

Data Input 3/3

- After declaring a **Scanner** class, its variable is ready to receive the value. The value depends on a type. When getting a value, the Scanner class must be able to convert it to its appropriate type. To support this, the **Scanner** class is equipped with a mechanism (actually called a method) for each type of value. To retrieve a value, you will write the name of the Scanner variable, followed by a period, followed by the mechanism as we will indicate, then assign it to the variable whose value you want to retrieve. The formula will be:
  - ***VariableName = ScannerVariable.Mechanism();***
- Notice the parentheses and the semi-colon.

Exercise

- Diketahui suatu tabung memiliki **`jarijariTabung`**, **`tinggiTabung`**
- Hitunglah;
  - **`luasAlasTabung`**
  - **`kelilingDindingTabung`**
  - **`luasDindingTabung`**
  - **`volumeTabung`**