

Variable

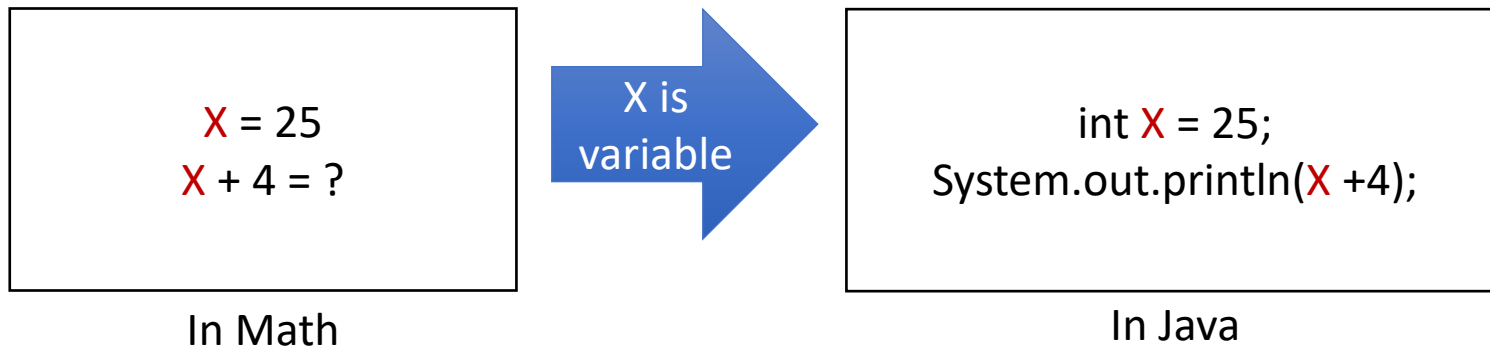
Sungchul Lee

Learning Object

- Variable
 - ❑ Name, type, memory
- Java Data Type
 - ❑ Primitive and Object
- Variable Declaration
- Initialize variable
- Re-assign

Variable

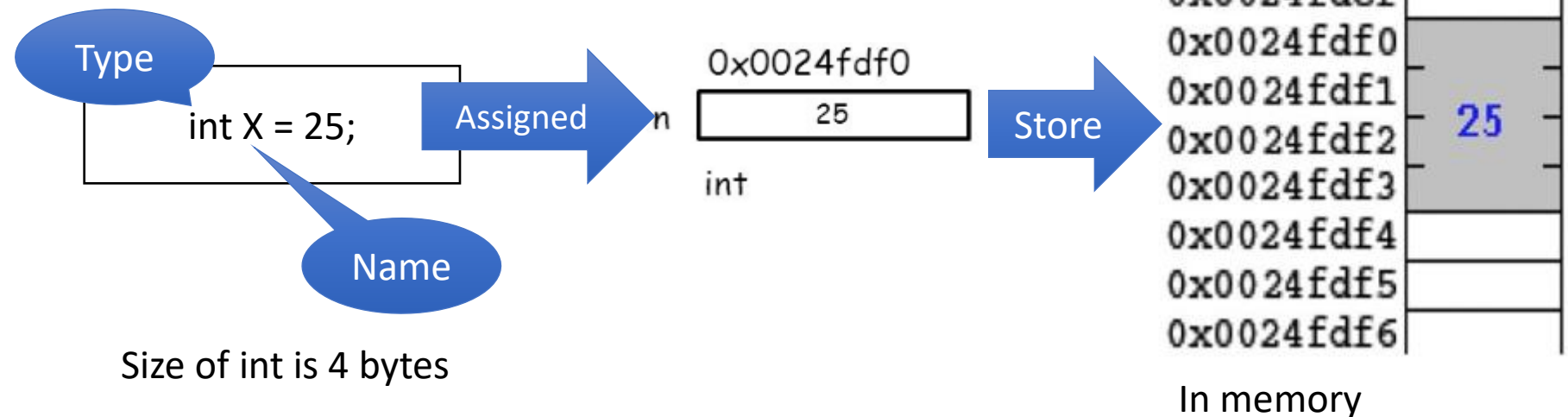
- Variable is base material for Java Program
 - ❑ Like stone and wood for building house
- Variable is a **symbolic** name associated with a **value**
 - ❑ Its associated value may be changed
 - ❑ How is it changed?
 - ❖ Value assignment



Variables in PL

➤ In a programming language (PL), a Variable has three properties

- ❑ The *name* used to refer to the memory location
- ❑ The *type* is to reserve memory size
- ❑ A *memory location* to store the data



Variable Names

- They must be **legal identifiers**
 - ❑ **Composed of letters, numbers, _ and \$**
 - ❑ No space: int ap ple
 - ❑ No start with number: int 1st
 - ❑ No special character: int a#
 - ❑ No Keyword: int class
- It must be **unique** within its scope
 - ❑ You shouldn't declare two same variables in a class

Recommended Naming convention in Java:

Variable names begin with a lowercase letter

Class names begin with an uppercase letter

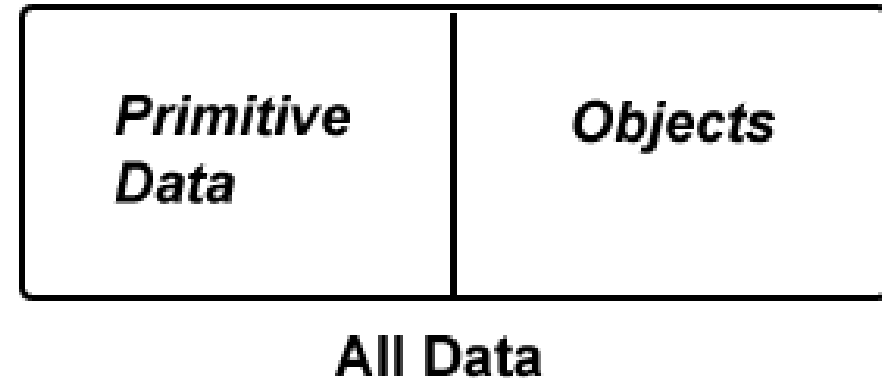
Java Keywords (Reference)

➤ Please, avoid the keywords to make variable name

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

Java Data Types

- Java is a **strongly typed language**; that is, the **compiler type-checks** all statements.
- Two categories of data types:
 - ❑ **Primitive**
 - ❑ **Object/Reference**
 - ❖ String, Class, Function



Java Primitive Types

Type	Contains	Size	Range
boolean	True or False	1 bit	NA
char	character	2 byte	Any Unicode character
byte	Integer	1 byte	-128~127
short	Integer	2 byte	-32,768~32,767
int	Integer	4 byte	$-2^{31} \sim 2^{31}-1$
long	Integer	8 byte	$-2^{63} \sim 2^{63}-1$
float	Floating point	4 byte	$\pm (1.4 \times 10^{-45} \sim \pm 3.4 \times 10^{38})$
double	Floating point	8 byte	$\pm (4.94 \times 10^{-224} \sim \pm 1.79 \times 10^{301})$

- 1 byte = 8bit
- 1 byte = 2^8 (Binary)

Variable Declaration

- Variable declaration:
 <data type> <variable name>;
 ❑ int numberOfPerson;
- If more than one variable has the same data type:
 ❑ <data type> <name1>, <name2>..;
 ❑ double x, y;

Principle: All variables in Java **MUST**
be declared before being used

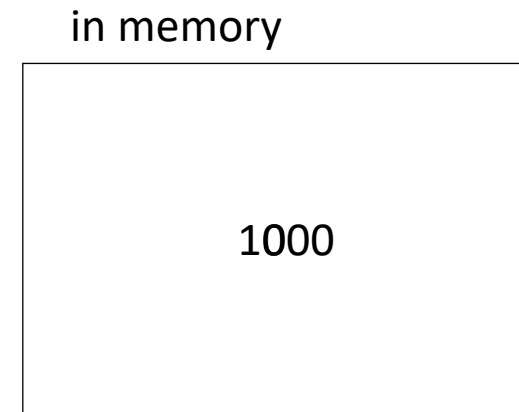
Initialize variables

- You can initialize variables when you declare them:
 - ❑ `int x = 3;`
 - ❑ `int y = 4, w, z;`
 - ❑ `w=2;`
- Must initialize the variable before use it in your code
 - ❑ `q = z+3; //error`

Re-assign variable

- Reuse the variable to contain a different values in the same memory
 - ❑ New value overwrites the previous value
 - ❑ Removed previous value

```
int loanAmount;  
❖ loanAmount = 0;  
❖ loanAmount = 1000;
```

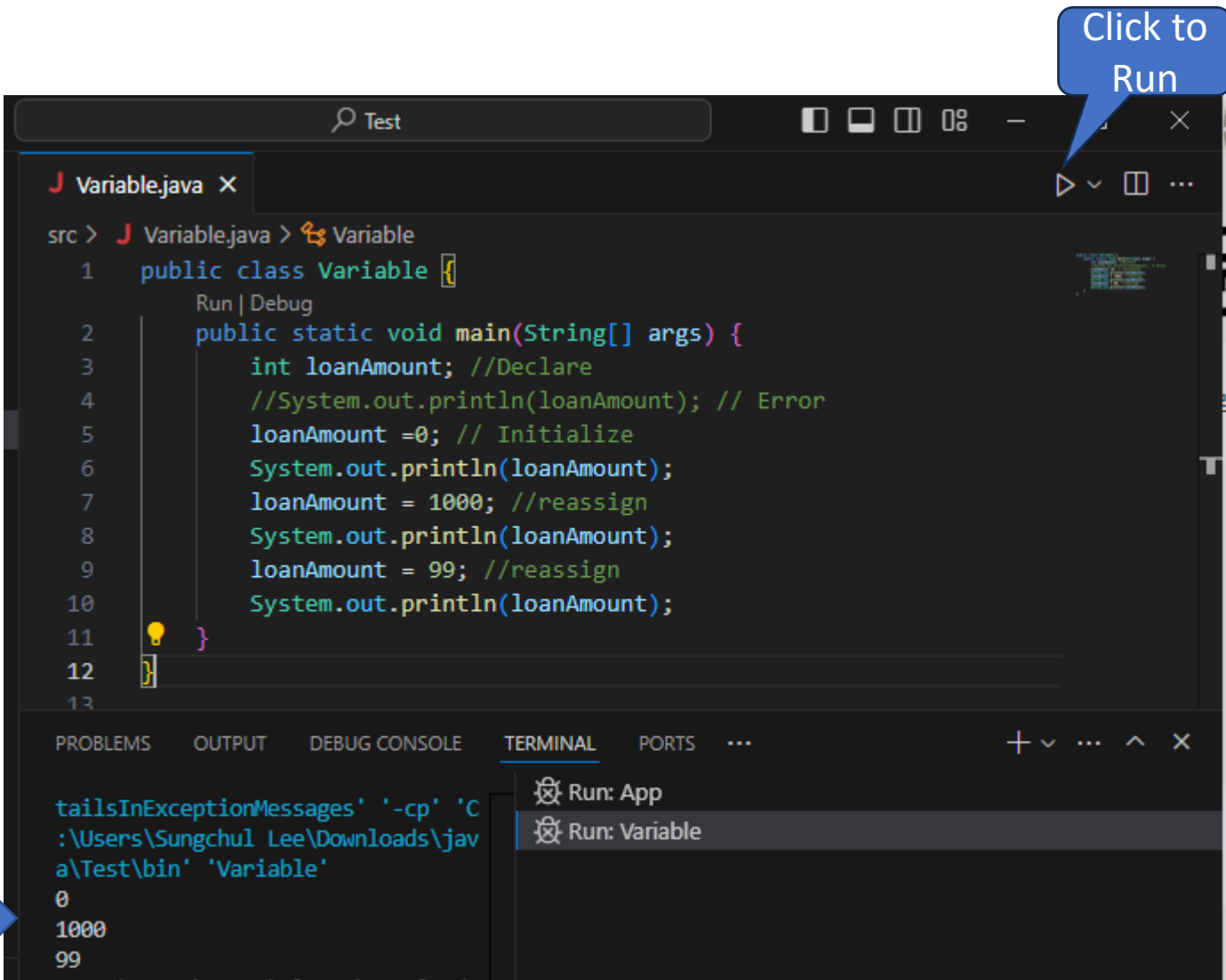


Practice

1. Make a new project (Reference: Create Project and Class File)
 - ❑ Project name: Variable
2. Create a new Class File
 - ❑ Class name: Variable
3. Coding:

```
public class Variable {  
    public static void main(String[] args) {  
        int loanAmount; //Declare  
        //System.out.println(loanAmount); // Error  
        loanAmount = 0; // Initialize  
        System.out.println(loanAmount);  
        loanAmount = 1000; //reassign  
        System.out.println(loanAmount);  
        loanAmount = 99; //reassign  
        System.out.println(loanAmount);  
    }  
}
```

Practice – Code and Result



The image shows a screenshot of an IDE with a Java file named `Variable.java`. The code defines a `public class Variable` with a `main` method. Inside `main`, an `int` variable `loanAmount` is declared, initialized to 0, and then reassigned to 1000 and then 99. The code is annotated with comments: `//Declare`, `//Error`, `// Initialize`, and `//reassign`. A blue callout bubble points to the run button in the IDE's toolbar, containing the text "Click to Run".

```
src > J Variable.java > Variable
1 public class Variable {
2     Run | Debug
3     public static void main(String[] args) {
4         int loanAmount; //Declare
5         //System.out.println(loanAmount); // Error
6         loanAmount = 0; // Initialize
7         System.out.println(loanAmount);
8         loanAmount = 1000; //reassign
9         System.out.println(loanAmount);
10        loanAmount = 99; //reassign
11        System.out.println(loanAmount);
12    }
13 }
```

The bottom panel of the IDE shows the `TERMINAL` tab. It displays the command used to run the program: `tailsInExceptionMessages' '-cp' 'C:\Users\Sungchul Lee\Downloads\java\Test\bin' 'Variable'`. The output of the program is shown below the command: `0`, `1000`, and `99`. A blue arrow points from the word "Result" to the output in the terminal.

Result

Run: App
Run: Variable

0
1000
99

Summary

➤ Variable

❑ Name, type, memory

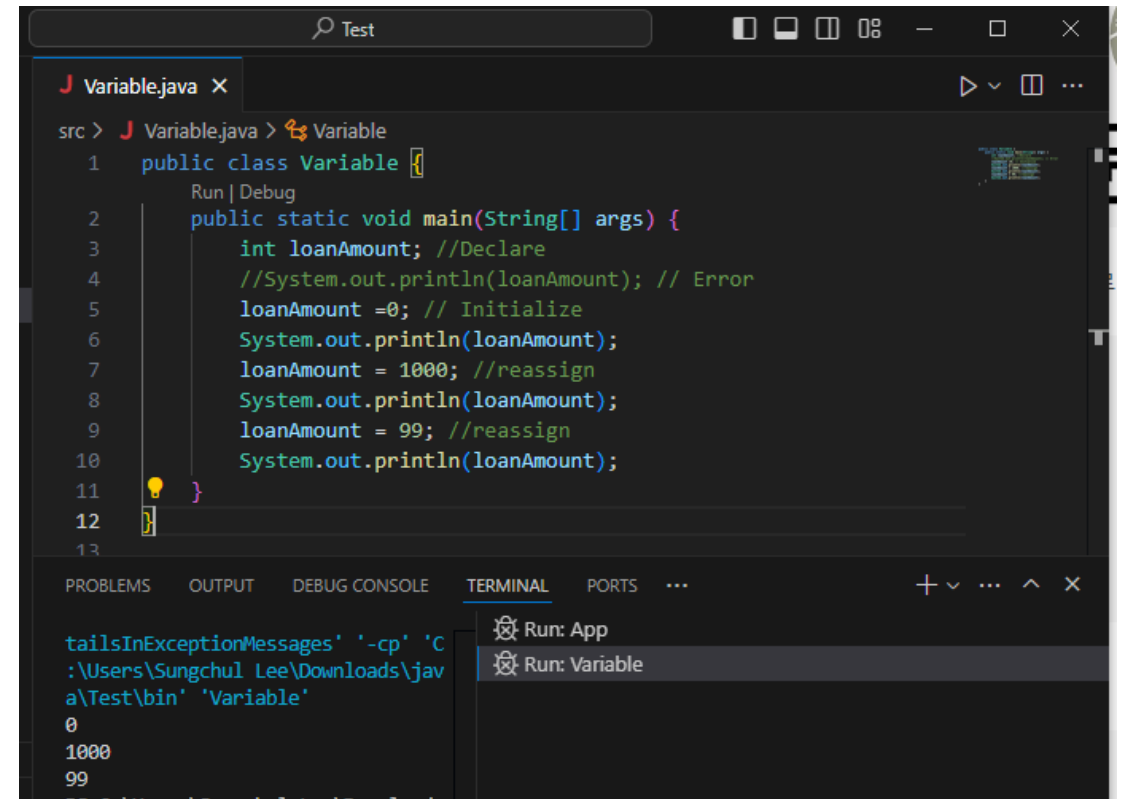
➤ Java Data Type

❑ Primitive and Object

➤ Variable Declaration

➤ Initialize variable

➤ Re-assign



```
src > J Variable.java > Variable
1 public class Variable {
2     public static void main(String[] args) {
3         int loanAmount; //Declare
4         //System.out.println(loanAmount); // Error
5         loanAmount = 0; // Initialize
6         System.out.println(loanAmount);
7         loanAmount = 1000; //reassign
8         System.out.println(loanAmount);
9         loanAmount = 99; //reassign
10        System.out.println(loanAmount);
11    }
12 }
13
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS ...

tailsInExceptionMessages' '-cp' 'C
:\Users\Sungchul Lee\Downloads\jav
a\Test\bin' 'Variable'
0
1000
99

Run: App
Run: Variable

```
int loanAmount;  
    loanAmount =0;  
    loanAmount = 1000;
```

in memory

1000