Java I

Chapter 2

Class Declaring

Class Names and Identifiers:

- By convention, begin with a capital letter and capitalize the first letter of each word they include (e.g., SampleClassName).
- A class name can be a series of characters consisting of:
    - Letters
    - Digits
    - Underscores (_)
    - Dollar signs ($)
- **But** the class name does not begin with a digit and does not contain spaces.

Note: Uppercase and lowercase letters are distinct (Java is case sensitive), so a1 and A1 are different (but both valid) identifiers.

Comments

The Java compiler *ignores* comments, so they do *not* cause the computer to perform any action when the program is run.

Single Line Comment	// Text	Indicates that the line is a comment.	
Traditional comment (Multi Line Comment)	/* Text*/	All text between the delimiters is ignored by the compiler	
Documentation Comment	/**Text*/	All text between the Javadoc comment delimiters is ignored by the compiler.	*(If specific tool is used, it reads Javadoc comments and uses them to prepare program documentation in HTML format. "Not covered in this course")*

Printing in Java

(**System.out**. …) is an object which allows a Java application to display information in the command window from which it executes.

1-System.out.print method

- Displays (or prints) a line of text in the command window.

Shortcut: 1. Type "sysout"
2. click SPACE+CTRL

- In case of using **System.out.**println method. It will place the output cursor at the beginning of the next line in the command window.
- Do not forget use quotation marks in case of typing text (or String)
  e.g. System.out.println("It always seems impossible until it's done");

Most statements end with a **semicolon** in Java.

2-System.out.printf method

- f means "formatted" displays *formatted* data.
- Method printf's first argument is a format string (" ") that may consist of:
  - Fixed text (output as it would be by print or println)
  - Format specifiers (placeholder for a value and specifies the type of data to output.)
- Format specifiers begin with a percent sign (%

Letter that represents the data type

%s	String
%n	Inserts a newline character
%d	Decimal integer
%f	Decimal floating-point
%c	Character

For example:

Format specifier

System.out.printf("Eng%d are the cutest",19);

Fixed text

Escape Sequence

A letter after a backslash (\) is an escape sequence and has special meaning to the java compiler. When an escape sequence is encountered in a print statement, the compiler interprets it accordingly.

\n (New line)	We use it to shift the cursor control to the new line
\t (Horizontal tab)	We use it to shift the cursor to a couple of spaces to the right in the same line.
\r (Carriage Return)	We use it to position the cursor to the beginning of the current line.
\\ (Backslash)	We use it to display the backslash character
\" (Double quote)	Used to print a double-quote character. For example, System.out.println("\"in quotes\""); displays "in quotes".

Variables in Java:

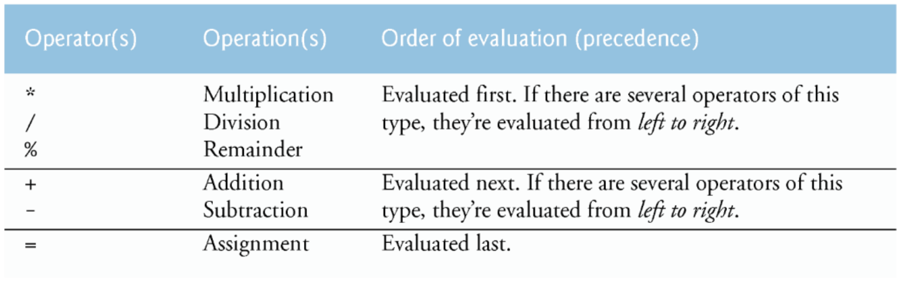
1-Variables types:

Type	Description	Default
String	Stores text, such as "Hello". String values are surrounded by double quotes	null
int	Stores integers (whole numbers), without decimals, such as 123 or -123	0
float	Stores floating point numbers, with decimals, such as 19.99 or -19.99 (32 bits)	0.0
double	Stores floating point numbers, with decimals Use a double (instead of float) if you need to save large data). (64 bits)	0.0
Boolean	Stores values with two states: true or false	false

2-Declarion of variables

	type	variable	=	value;
Examples	int	x	=	10;
	String	name	=	"Layan";
	float	high	=	100.09;

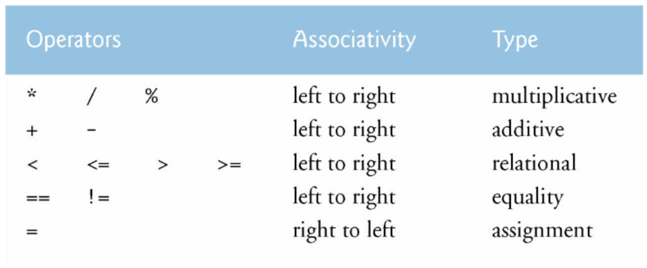
3-Arithmetic ()



Operator(s)	Operation(s)	Order of evaluation (precedence)
* / %	Multiplication Division Remainder	Evaluated first. If there are several operators of this type, they're evaluated from *left to right*.
+ -	Addition Subtraction	Evaluated next. If there are several operators of this type, they're evaluated from *left to right*.
=	Assignment	Evaluated last.

Decision making

- Condition: an expression that can be true or false.
- if selection statement
  - Allows a program to make a decision based on a condition's value.
- Equality operators (== and !=)
- Relational operators (>, <, >= and <=)
- Both equality operators have the same level of precedence, which is *lower* than that of the relational operators.
- The relational operators all have the same level of precedence



Operators	Associativity	Type
`*` `/` `%`	left to right	multiplicative
`+` `-`	left to right	additive
`<` `<=` `>` `>=`	left to right	relational
`==` `!=`	left to right	equality
`=`	right to left	assignment

Fig. 2.16 | Precedence and associativity of operators discussed.

Scanner

- Enables a program to read data for use in a program.
- Data can come from many sources, such as the user at the keyboard or a file on disk.

Here are 3 steps to enable user to enter data that will be used in the program:

1. Importing Scanner class

Before using a Scanner, you must import its class by typing `import java.util.Scanner;` at the top of the current class.

2. Scanner declaration statement

Scanner input = new Scanner(System.in);

Specifies the name (input) and type (Scanner) of a variable that is used in this program.

3. Scanner method nextInt

number1 = input.nextInt()

it will read first number from user

- Obtains an integer from the user at the keyboard.
- Program waits for the user to type the number and press the Enter key to submit the number to the program.
- The result of the call to method nextInt is placed in variable number1 by using the assignment operator, =.
- “number1 gets the value of input.nextInt().”
- Operator = is called a binary operator—it has two operands.
- Everything to the right of the assignment operator, =, is always evaluated before the assignment is performed.