JA111

Day-2

Comments in java

Single line comment

Comment have length limited to just one line.
To insert a single line comment, simply start comment with two consecutive slashes (/).
Example: //This is a single line comment.

Multi line comment

Comment have length more than one line.
To insert a multi-line comment, simply start comment with a slash (/) and asterisk sign (*), and after completing comment just place an asterisk (*) sign and slash (/)
Example:

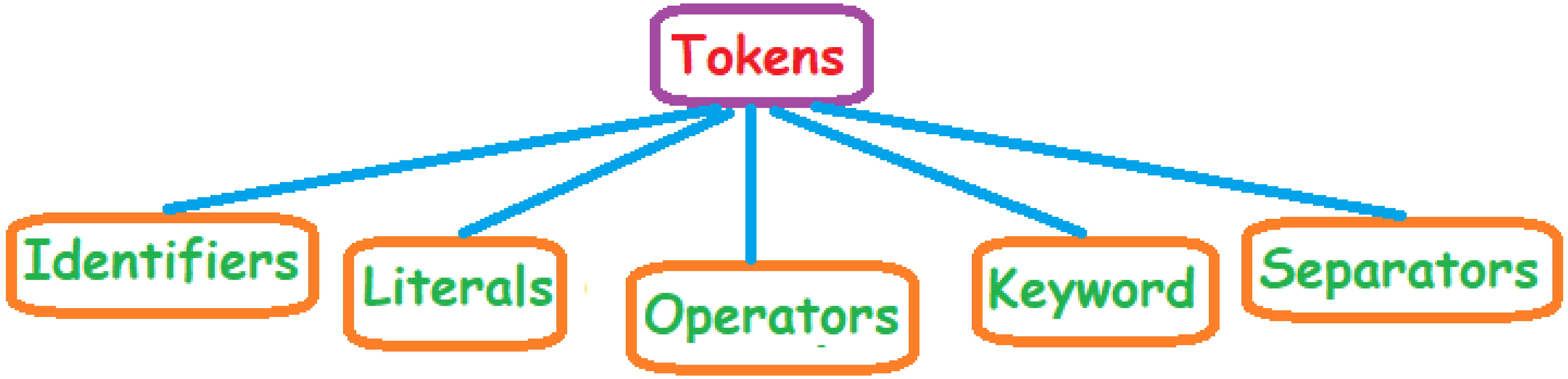
```
/* I am multi line comment.
This is another line of comment */
```

javadocs

They are used to generate the HTML pages of API documentation from Java source files.
Separate syntaxes are used to write javadoc for classes, methods, packages etc. We will example of such comments after making a significant progress in the java.

Tokens

- It refer to smallest lexical unit of a languages
- java tokens are classified into 5 categories as follow



Identifier

- It refers to sequence of one or more character that is used to identify an entity like a memory location (i.e. variable) or to a block of code (i.e. method, class etc.)

data-type identifier;
or
data-type identifier = value;

<u>Variable Naming Rules</u>

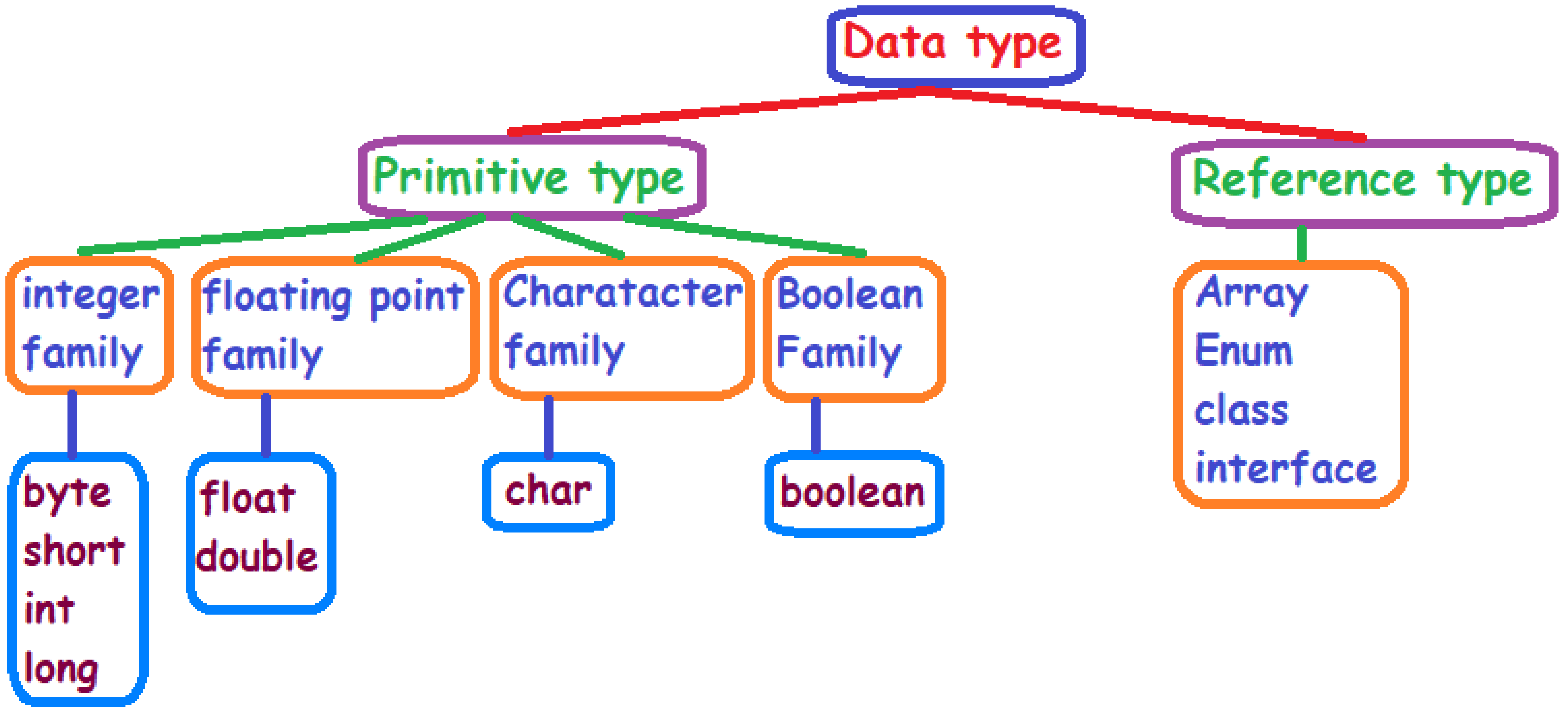
- ❑ Can be made from [A-Z], [a-z], [0-9], _ , $
- ❑ Can't be started with [0-9]
- ❑ Can't be a reserved keyword
- ❑ *Variable name should not be started with _ or $ yet they are allowed. It is good to start them with a letter.*

Why Data Types

A variable's data type defines following properties about a variable-

❑ Values it may contain

❑ Operations that may be performed on it.



Data Types

Family	Data-type	Size	Range	Example
Integer family	byte	1 byte/8 bits	-128 to 127	byte c = 1;
	short	2 bytes/16 bits	-32768 to 32767	short t = 150;
	int	4 bytes/32 bits	-2147483648 to 2147483647	int j = 150000;
	long	8 bytes/64 bits	-9223372036854775808 to 9223372036854775807	long l = 2147483647L;
Floating point family	float	4 bytes/32 bits	1.4e−045f to 3.4e+038f	float f = 1.5f;
	double	8 bytes/64 bits	4.9e–324 to 1.8e+308	double d = 1.5;
Character family	char	2 bytes/16 bits	0 to 65535	char ch = 'A';
Boolean family	boolean	-	{true, false }	boolean b = false;

- ✓ All the non-fractional numerical values belong to integer type
- ✓ Long values must value must be appended with L (Capital Lion) or l (Small lion)
- ✓ All the fractional numerical values belong to double type.
- ✓ To make the value belong to the float type the value must be appended with F or f.

- ✓ You may append double value with d or D
- ✓ float data type : single-precision
- ✓ double data type : double-precision
- ✓ character data type uses Unicode encoding

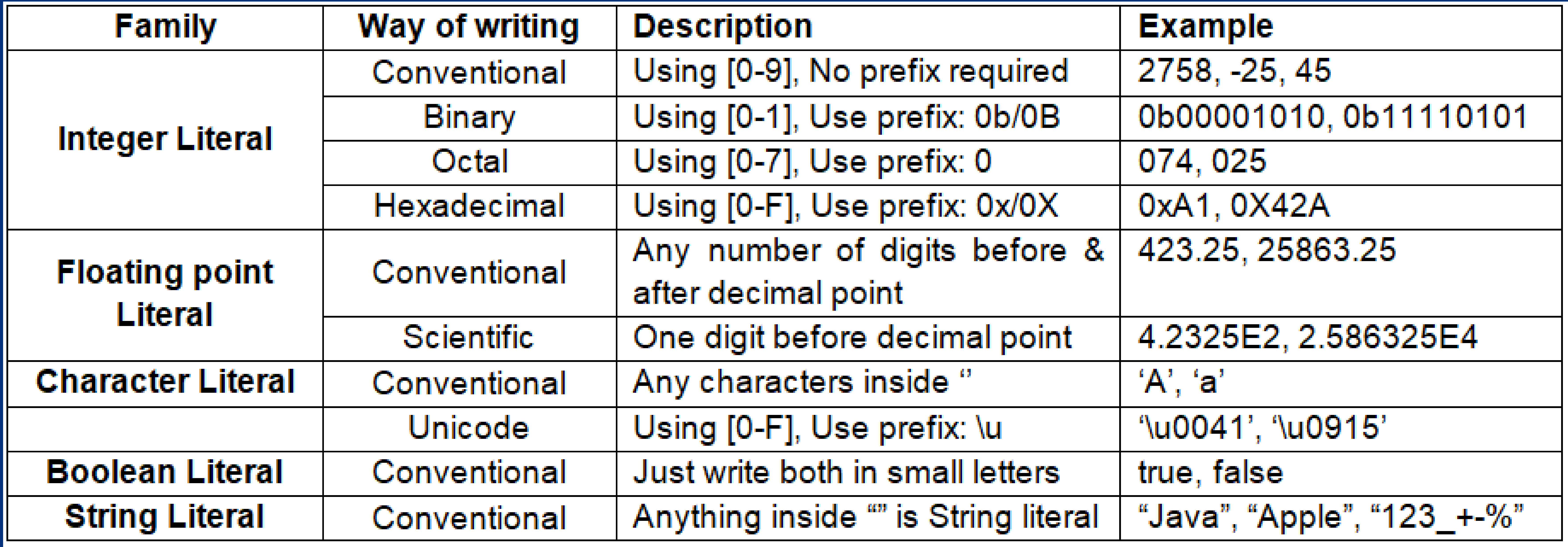
Wrapper classes

✓ java is an object oriented language. To support the same theme java provides object implementation of the primitive data type using Wrapper classes.

Primitive type	**Wrapper class**
byte	Byte
short	Short
int	Integer
long	Long
char	Character
float	Float
double	Double
boolean	Boolean

Literals

- ✓A literal is a source code representation of a fixed value. They are represented directly in the code without any computation.
- ✓You can place underscore character between number for the sake of improving readability.



Family	Way of writing	Description	Example
Integer Literal	Conventional	Using [0-9], No prefix required	2758, -25, 45
	Binary	Using [0-1], Use prefix: 0b/0B	0b00001010, 0b11110101
	Octal	Using [0-7], Use prefix: 0	074, 025
	Hexadecimal	Using [0-F], Use prefix: 0x/0X	0xA1, 0X42A
Floating point Literal	Conventional	Any number of digits before & after decimal point	423.25, 25863.25
	Scientific	One digit before decimal point	4.2325E2, 2.586325E4
Character Literal	Conventional	Any characters inside ‘’	‘A’, ‘a’
	Unicode	Using [0-F], Use prefix: \u	‘\u0041’, ‘\u0915’
Boolean Literal	Conventional	Just write both in small letters	true, false
String Literal	Conventional	Anything inside “” is String literal	“Java”, “Apple”, “123_+-%”

Type conversion and casting

Automatic type conversions/widening conversion

When we assign one type of data to another type of variable then automatic type conversion take place in the following conditions met

- ❑ The two types are compatible.
- ❑ The destination type is larger than the source type.

Narrowing conversion & Truncation

When source type is larger than the destination type or when variable is incompatible with the data to be assigned then you have to explicitly make the value narrower so that it can fit into the target type.

To perform conversion between two incompatible types, you must use a cast. A cast is simply an explicit type conversion. It has this general form:

```
(target-type) value
```

Java is known as *strongly typed language.*

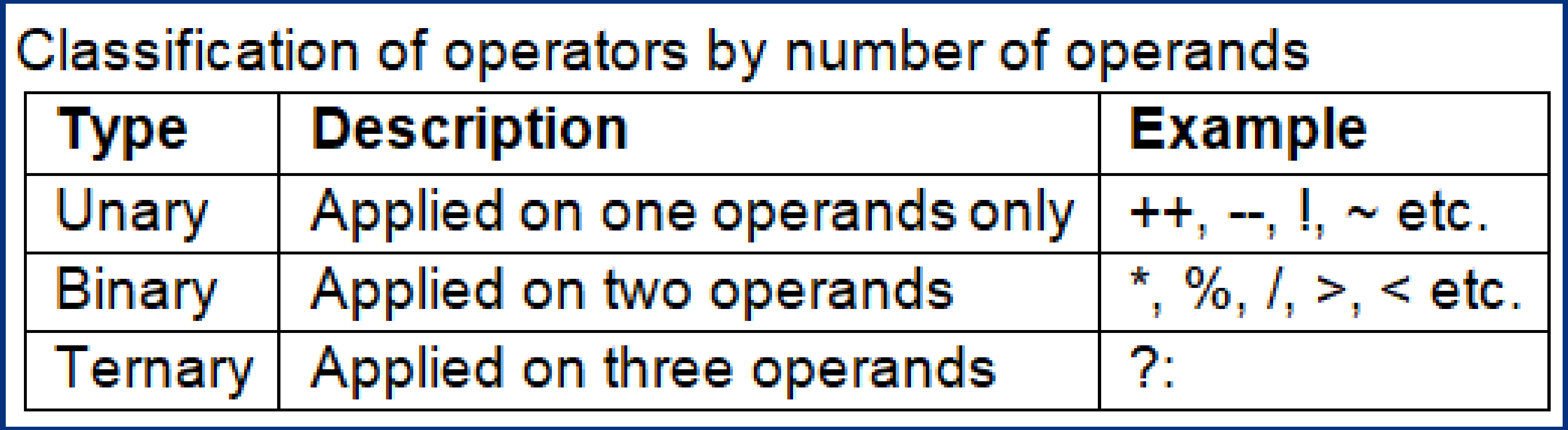
Keyword

- ✓ It refer to words whose meaning is already defined to the compiler
- ✓ Keyword cannot be used an identifier
- ✓ Java has total 50 keywords

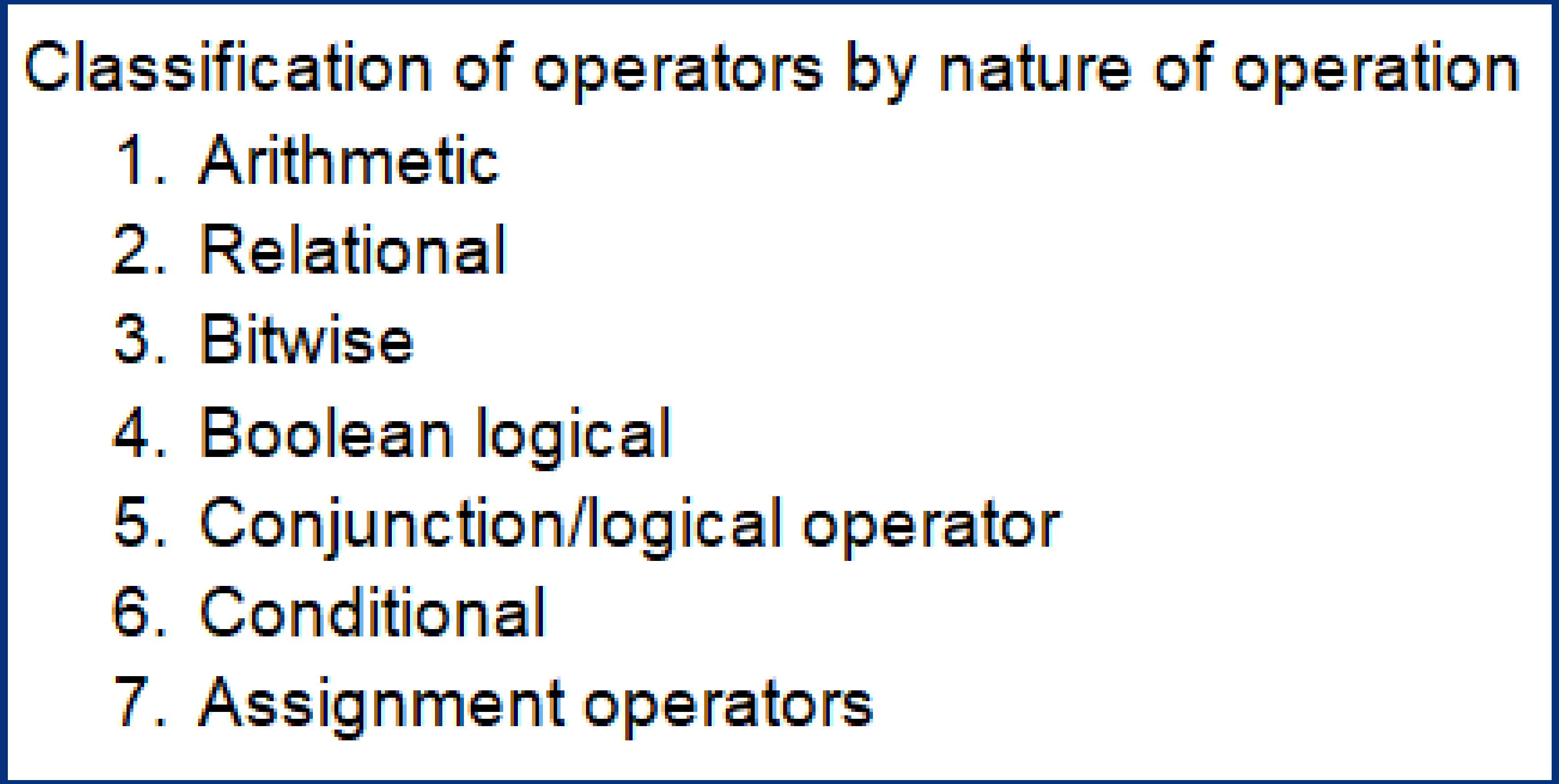
switch	new	for	continue	abstract
synchronized	package	goto (not used)	default	assert
this	private	if	do	boolean
throw	protected	implements	double	break
throws	public	import	else	byte
transient	return	instanceof	enum	case
try	short	int	extends	catch
void	static	interface	final	char
volatile	strictfp	long	finally	class
while	super	native	float	const (not used)

Operators

- ✓ They are the symbols that are used to specify an operation
- ✓ The values/variables on which operations applied are called operands
- ✓ A valid combination of operators and operands is called expression


Classification of operators by number of operands

Type	Description	Example
Unary	Applied on one operands only	++, --, !, ~ etc.
Binary	Applied on two operands	*, %, /, >, < etc.
Ternary	Applied on three operands	?:


Classification of operators by nature of operation

1. Arithmetic
2. Relational
3. Bitwise
4. Boolean logical
5. Conjunction/logical operator
6. Conditional
7. Assignment operators

Operator Evaluation in Java

No BODMAS

Operators are evaluated according to priority, if priority cannot decide then associativity becomes the decider

Operators		Priority
Brackets	() []	Highest
Postfix	expr++ expr--	
Unary$^{@}$	++expr –expr ~ ! –expr +expr	
Multiplicative	/ % *	
Additive	+ -	
Shift	>> << >>>	
Relational	< > <= >= instanceof	
Equality	== !=	
Bitwise AND	&	
Bitwise XOR	^	
Bitwise OR	|	
Logical AND	&&	
Logical OR	||	
Ternary	?:	
Assignment$^{@}$	= += -= /= *= >>= <<= >>>=	

The operators superscripted with @ have associativity from Right to left, rest all operators have associativity from left to right.

Operator Evaluation in Java

- ❑ Java automatically converts short and byte operand to int whenever evaluating an expression.
- ❑ If an expression involves several data type then final result of expression will belong to largest data type.

```
int a = 5;
int b = (--a + --a);      //7
int c = (++a – a--);      //0
int d = (--a + a--);      //4
Int e = (++a + a++);      //4
Int f = b + c + d + e
System.out.println(f);
```

```
byte a = 20;
byte b = 2;
//byte c = a/b; Compile Time Error
byte c = (byte)(a/b);
System.out.println("c is " + c);
```

```
byte b = 2;
short s = 5;
int i = 10;
float f = 11.2f;
double d = 111.23;
double result = ((b*f)/i) + f*s + d;
System.out.println(result); //169.47000167846682
```

Operators

Arithmetic Operators

+, -, *, /
%, ++, --

Relational Operators

>, <, >=
<=, !=, ==

Bitwise Operators

~, &, |, <<,
>>, >>>

Boolean Logical Operators

!, &, |, ^

Assignment Operators

=, +=, -=,
&=, |= etc.

Conjunction Operators

&&, ||

Conditional Operators ?:

Control Statements

- If statement
- Nested if statement
- If-else-if statement
- If-else-if ladder
- switch-case statement

Control Statements (contd.)

while loop

for loop

do-while loop

break

continue