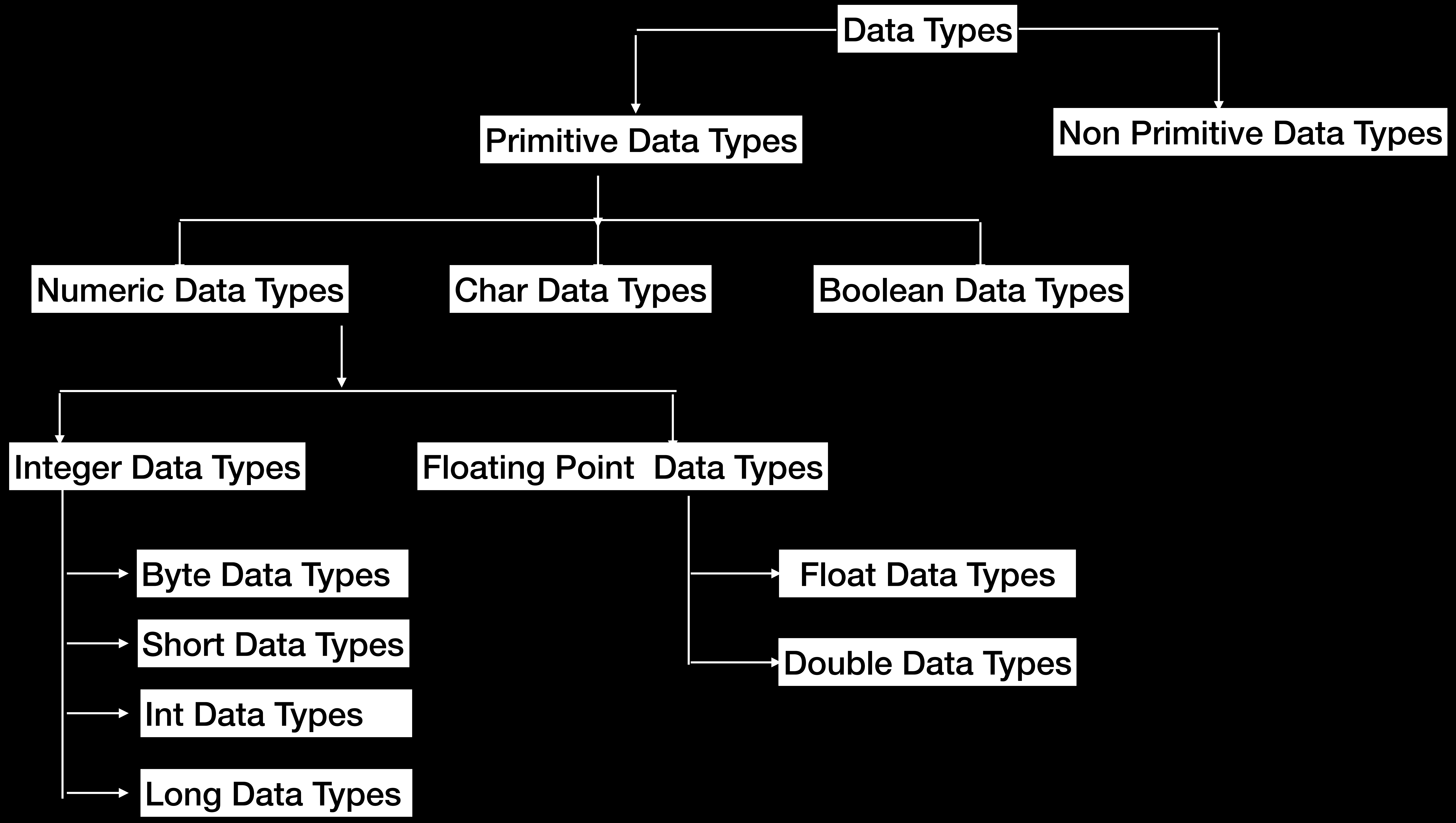
Day-2

Agenda

. Data Types.
. Literals
. Operators

Data Types



Data Types

Primitive Data Types

Non Primitive Data Types

Numeric Data Types

Char Data Types

Boolean Data Types

Integer Data Types

Floating Point Data Types

Byte Data Types

Short Data Types

Int Data Types

Long Data Types

Float Data Types

Double Data Types

byte Data Types

Size	8 bits or 1 byte
Max Value	127
Min Value	-128
Range	-128 to +127
Syntax	**byte** variable_Name = value
Example	byte b = 110

short Data Types

Size	2 bytes or 16 bits
Max Value	32767
Min Value	-32768
Range	-32768 to +32767
Syntax	**short** variable_Name = value
Example	short b = 32767

int Data Types

Size	4 bytes
Max Value	2,147,483,647 or $2^{31}-1$
Min Value	-2,147,483,648 or -2^{31}
Range	-2,147,483,648 to +2,147,483,647
Syntax	int variable_Name = value
Example	int b = 5

long Data Types

Size	8 bytes
Max Value	9,223,372,036,854,775,807 or $2^{63} - 1$
Min Value	-9,223,372,036,854,775,808 or -2^{63}
Range	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
Syntax	long variable_Name = value
Example	long b = 5000

float Data Types

Size	4 bytes
Max Value	$3.40282347 \times 10^{38}$
Min Value	$1.40239846 \times 10^{-45}$
Range	$1.40239846 \times 10^{-45}$ to $3.40282347 \times 10^{38}$
Syntax	float variable_Name = valueF
Example	float b = 10.5f

double Data Types

Size	8 bytes
Max Value	1.7976931348623157 x 10^{308}
Min Value	4.9406564584124654 x 10^{-324}
Range	4.9406564584124654 x 10^{-324} to 1.7976931348623157 x 10^{308}
Syntax	double variable_Name = value
Example	double b = 10.5d

char Data Types

Size	2 byte
Max Value	65535
Min Value	0
Range	0 to 65535
Syntax	char variable_Name = value
Example	char initialName = ‘a’

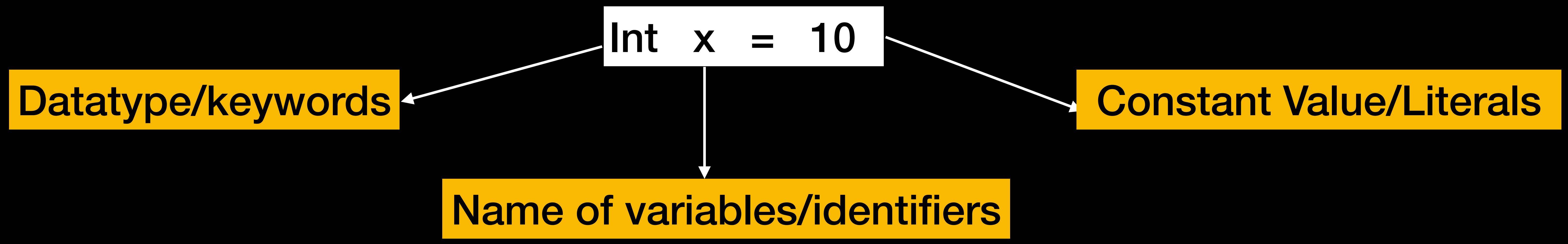
boolean Data Types

Size	NA
Max Value	NA
Min Value	NA
Range	NA
Value	true or false
Example	boolean isPass = true

Float	Double
Size 4 byte	**Size - 8 byte**
If we want 5 to 6 decimal place of accuracy than we should go for float	If we want 14 to 15 decimal place of accuracy than we should go for double

Literals

A constant value which can be assign to the variable is called “Literal”



Int x = 10

Datatype/keywords

Name of variables/identifiers

Constant Value/Literals

Integral Literals

- Decimal Literals (Base 10)
- Octal Literals (Base 8)
- Hexadecimal Literals (Base 16)
- Binary literals

Char Literals

- Single quote
- Char literal as Integral literal
- Unicode Representation
- Escape Sequence

Floating - Point Literals

- Decimal Literals (Base 10)

Boolean Literals

String Literals

Literals

Integral Literals

Decimal Literals (Base 10)

Octal Literals (Base 8)

Hexadecimal Literals (Base 16)

Binary literals

Floating - Point Literals

Decimal Literals (Base 10)

Char Literals

Single quote

Char literal as Integral literal

Unicode Representation

Unicode Representation

Boolean Literals

String Literals

Literals

Integral Literals

Decimal Literals (Base 10)

Allow digit are 0 to 9 . Ex : int x=10

Octal Literals (Base 8)

Allow digit are 0 to 7 . The octal number should be prefix with ZERO(0)

Ex : int x=010

Literals

Integral Literals

Hexadecimal Literals (Base 16)

Allow digit are 0 to 9 . And characters are “a to f” or “A to F”

Prefix with “ox” or “oX”

NOTE : Since JAVA is case-sensitive language but here, this is the only place where this rule is **not valid**

Binary literals

From JAVA 1.7 onwards , Integral Literals can be specified even in binary also .

Allow digit are 0 & 1 but literals value should be prefix with “0b” or “0B”

Ex: int x=0b1111

Literals

Char Literals

Single quote

Char Literals can be represented as Single Character with single quote. Ex : char c=‘a’

Char literal as Integral literal

Char Literals can be represented as Integer Literals which represent unicode of that character

Can be specify as Integral literals either in Decimal or Octal or Hexa Decimal form

Range should be 0 to 65535

Ex: char ch = 97

Literals

Char Literals

Unicode Representation

A Char literals can be represented in unicode representation which is notify as “\U- - - - ” . Four digit hex decimal no.

Ex- **char ch=‘\u0061’**

Escape Sequence

It can be specify as Escape character as below

Ex: char ch = \n

Escape character	Meaning
\n	New Line
\t	Horizontal Tab
\r	Carriage Return
\b	Back Space
\f	Form feed
\’	Single Quote
\”	Double Quote
\\	Back Slash

Literals

Floating - Point Literals

Decimal Literals (Base 10)

Every floating point literals is by default double type & hence we would not assign directly to float variable. But we can specify explicitly floating point literals, as the float type by specifying with ‘f’ or ‘F’

Ex: float f= 3.1414 F;

We can specify floating point literals explicitly as double type by suffixing with ‘d’ or ‘D’.

Ex**- double d=3.1414D;**

Escape Sequence

Literals

Boolean Literals

The only possible value for the Boolean is “true” OR “false”.

Ex: boolean b= true;

String Literals

Any Sequence of character with in “…” is called String literals.

Ex- **String s=“JAVA”;**

Operators

Increment & Decrement operators [+]

Difference between b++ and b+1

String concatenation operators [+]

Relational Operators [> , < , >= , <=]

Equality Operators [== , !=]

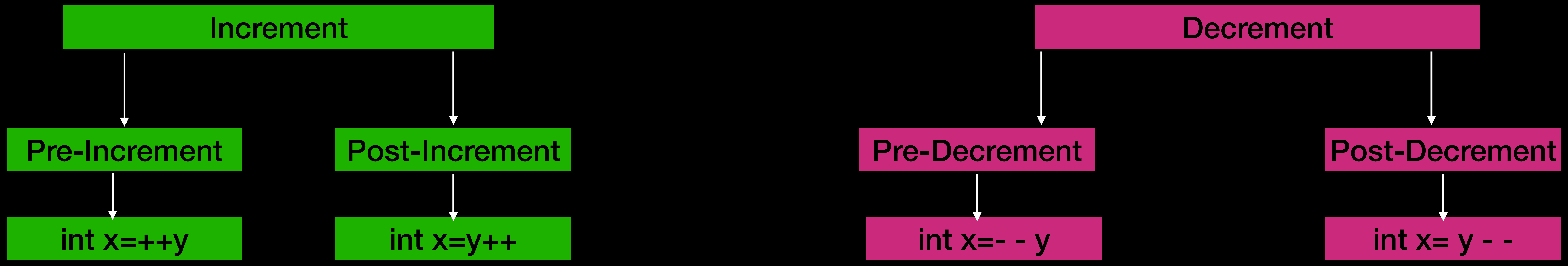
Bit-wise Operators [& , | , ^]

Bit-wise Complement Operators [~]

Boolean Complement Operators [!]

Short-Circuit Operators [&& , ||]

Operators



Increment

Pre-Increment

int x=++y

Post-Increment

int x=y++

Decrement

Pre-Decrement

int x=- - y

Post-Decrement

int x= y - -

Operators

Difference between b++ and b+1

Operators

String concatenation operators [+]

Operators

Relational Operators [> , < , >= , <=]

Operators

Equality Operators [== , !=]

Operators

Bit-wise Operators [& , | , ^]

Operators

Bit-wise Complement Operators [~]

Operators

Boolean Complement Operators [!]

Operators

Short-Circuit Operators [&& , ||]