

java -version : → whether java is installed or not.

javac -version
↳ x

Will learn java

↳ 1. java

javac

↳ Compile
↳ execute / Interpret

↑ class
↑ Bytecode

JDK

↓

JRE

JVM

javac

Env. variable to configure javac

→ Edit System Env. Variable.

→ Env. Variables < user → Specific user

< system → for All the System Users.

→ path → Edit → NEW → paste the path of JDK folder

C:\Program Files\Java\jdk1.8.0_271\bin

→ OK → OK → OK

→ Open New cmd and run command

javac -version

→ java 1.8

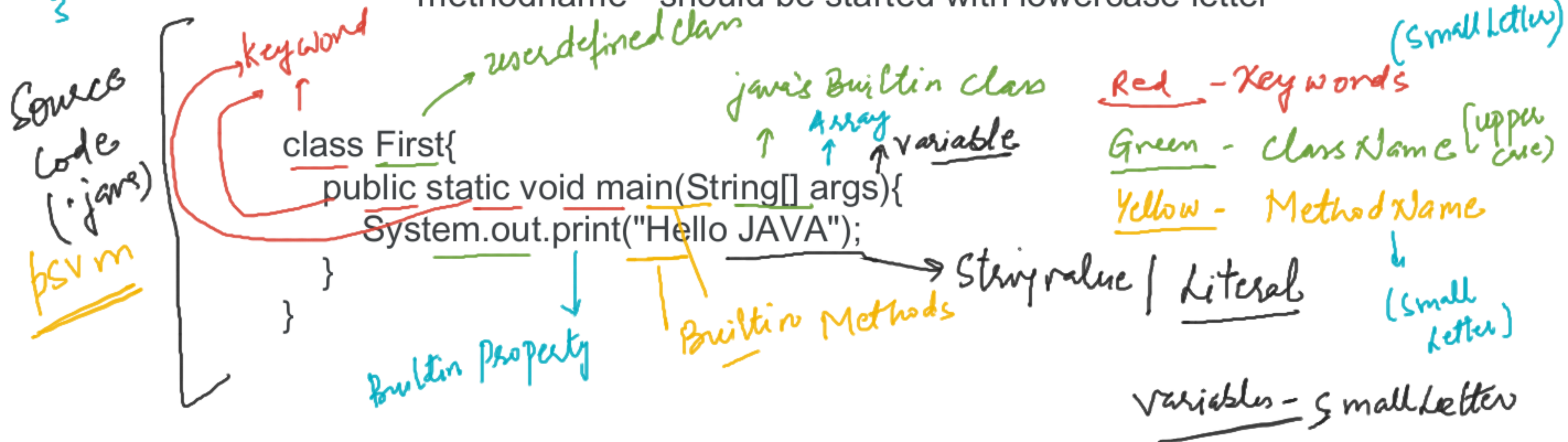
Notepad ++] →
Notepad

Name Convention: relevant / proper name

Creating any class, fileName, methodName

Classname - First letter of class Name should be in uppercase and
File name: is also starts with uppercase letter

methodname - should be started with lowercase letter

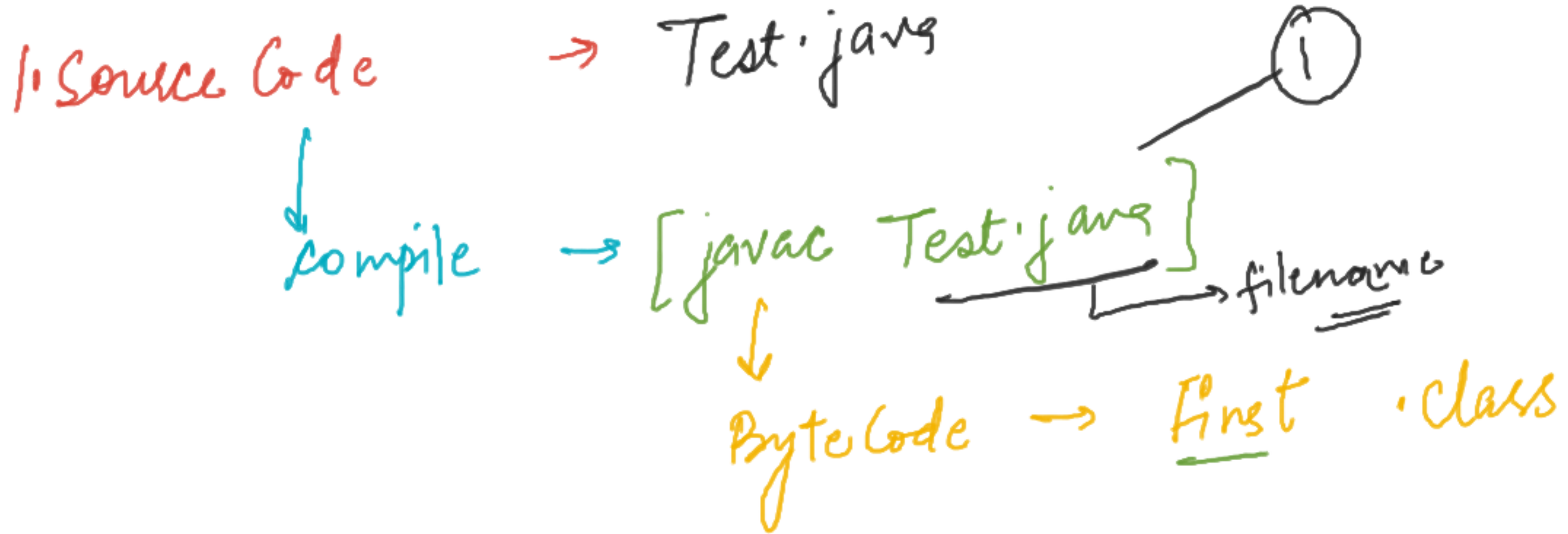


(i) `ClassName` and `FileName` is different - Yes, it compiles & executes the code successfully.

(ii) `ClassName` & `Filename` is Same \rightarrow Yes, "

(iii) If we write 'public' keyword in front of class, in case `ClassName`
and `FileName` is diff. \rightarrow Error at compile-time.
We need to keep the file & `ClassName` same, if we are using `public` keyword.

Program Execution Flow



JVM starts the execution from main() method.

```
public class Test{  
    public static void main(String[] args){  
        System.out.print("Hello JAVA");  
    }  
}
```

→ Creating a class
↓
To print the statement

→ main method from where JVM starts the execution

int a = 10
 ↓
 a++ ✓ post increment
 (+1)

System(a); // 11

pre ← ++a; // +1 = 12
 a++ 13

System(a);

int a = 12
a++; // 13 M/m

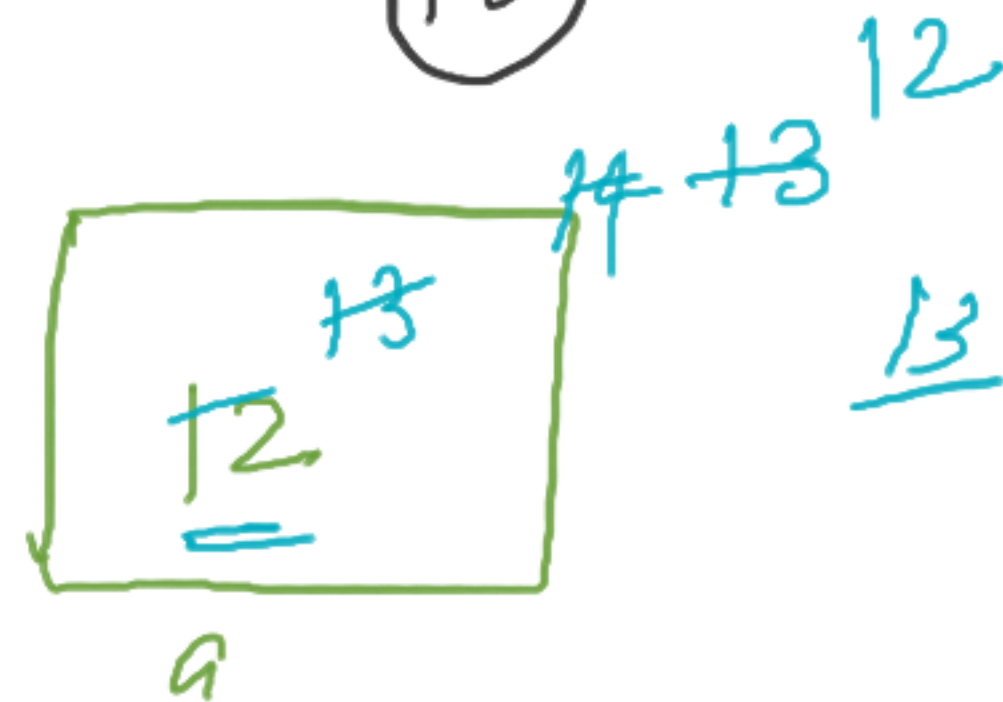
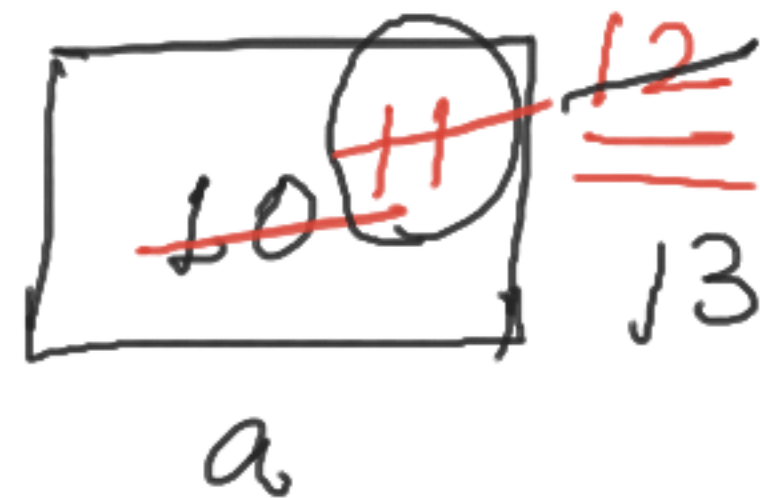
++a; // 14

a--;

--a;

a++;

System(a); // 13



int x = 1;

1 2

x

Syso(x++);

①

②

post increment

2 Action

Action 1 Print value

Syso(x); → ②

int y = 2;

y++; // 3

++y; // 4

✓ Syso(y++); // 4

++y; // 6

✓ Syso(y); // 6

M | m 5 6

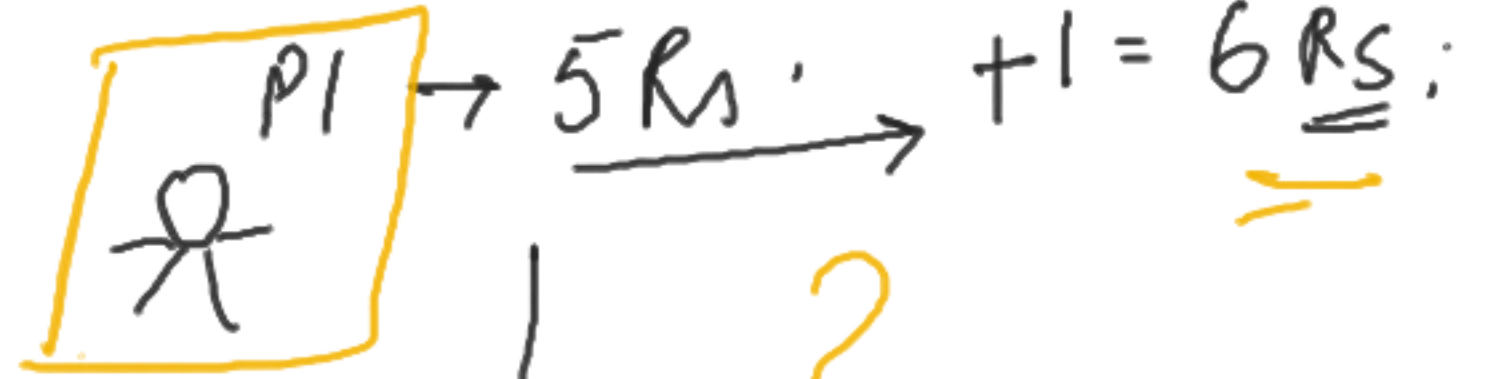
2 3 4

y

Pre-Increment

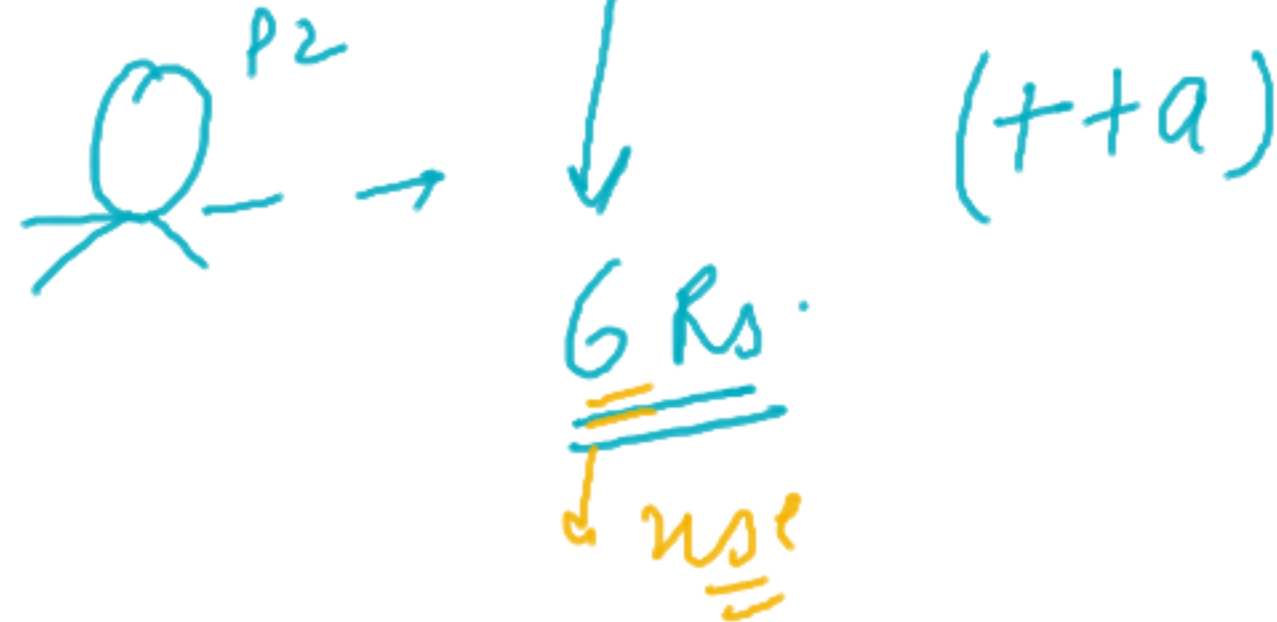
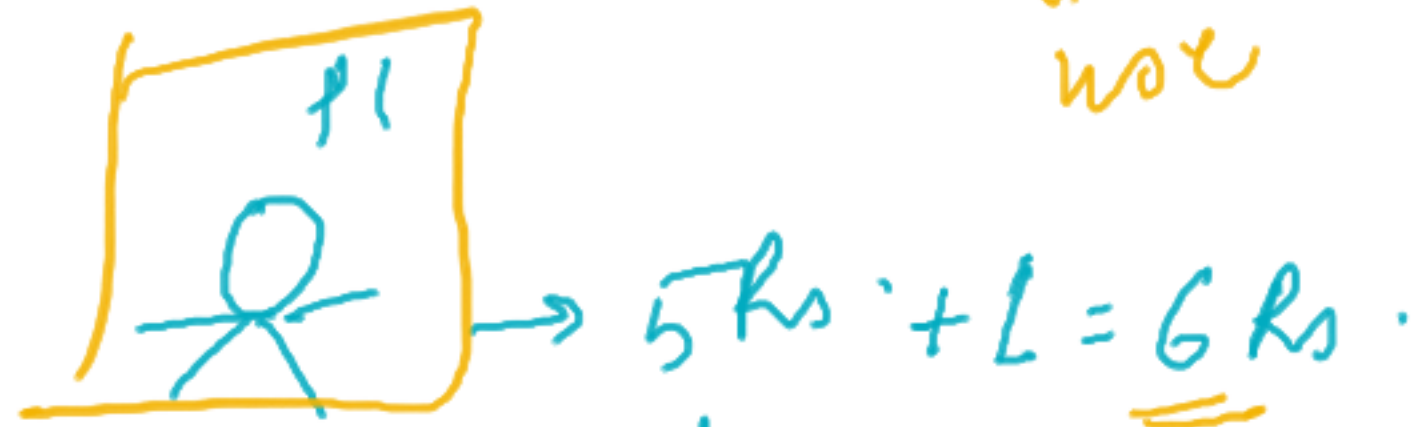
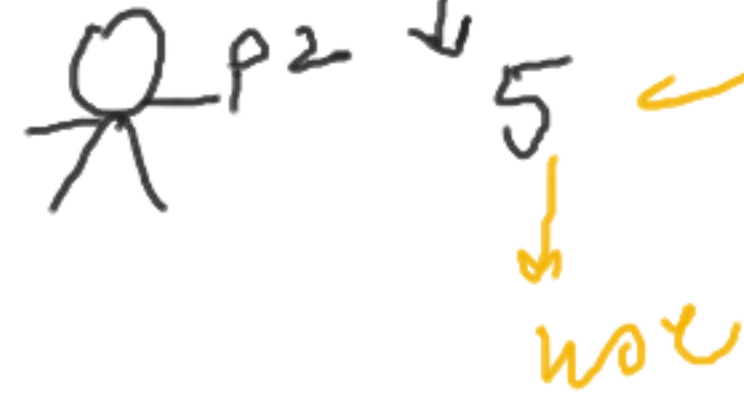
int a = 2;

sys0(++a); → (3)



sys0(a⁺⁺);

sys0(Rs
6ms)
6



(++a)

Ex- pre-post increment

int x = 10;

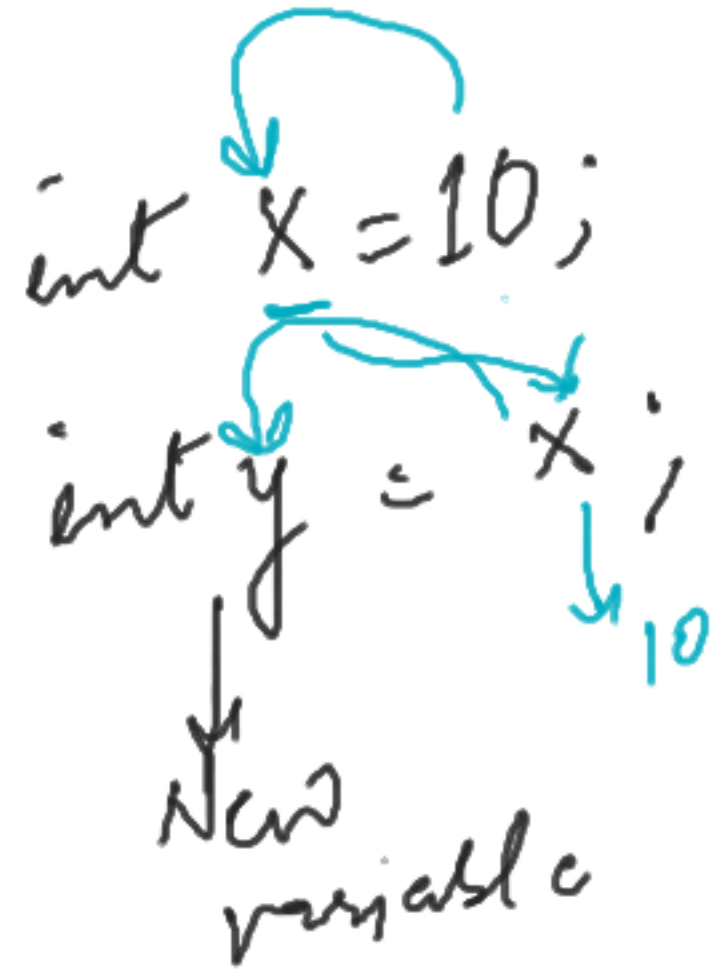
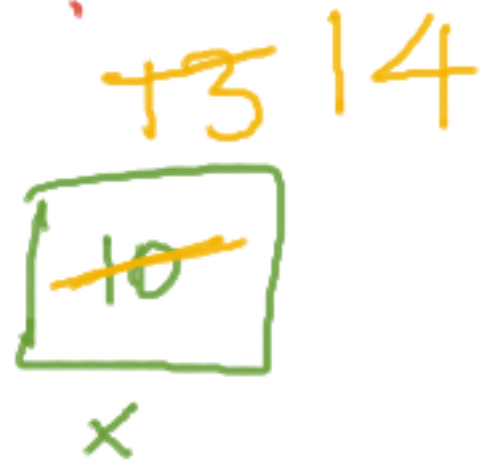
x++; // 11

++x; // 12

✓ Syso(x++); // 12

✓ Syso(++x); // 14

✓ Syso(x); // 14

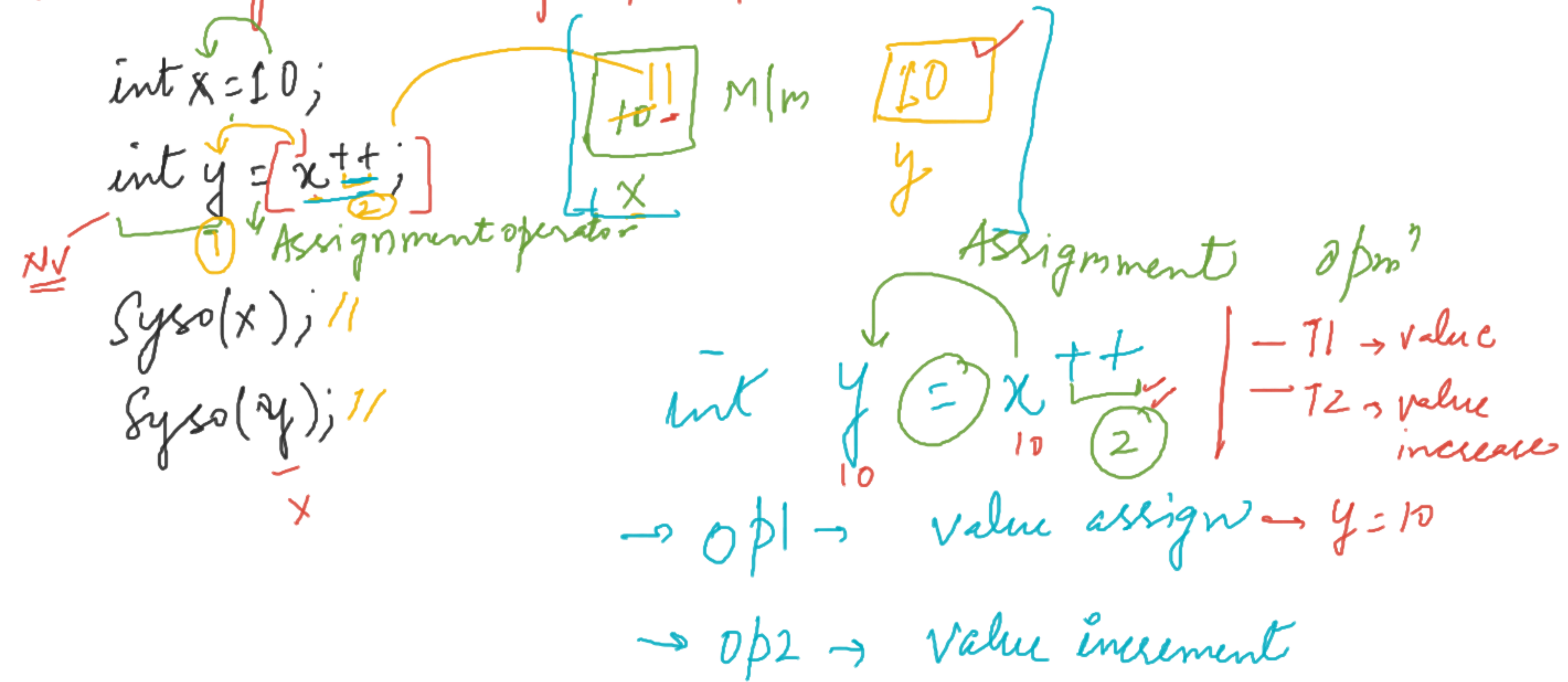


{ y = 10
x = 10 }

declaration → int x, int y

init → (x)

III) assign the value of pre-post increment in a variable.



Data Types

JVM

↓
(interpreter)

→ Primitive DT

→ Non primitive DT

→ that can store a single value in a variable

→ that can hold multiple values in a variable.

1 Bit → 1 0 1 0
 ↓
 Bit

1 Byte → 8 bits

→ primitive DT

byte
short
int
long

} numeric integers

float
double

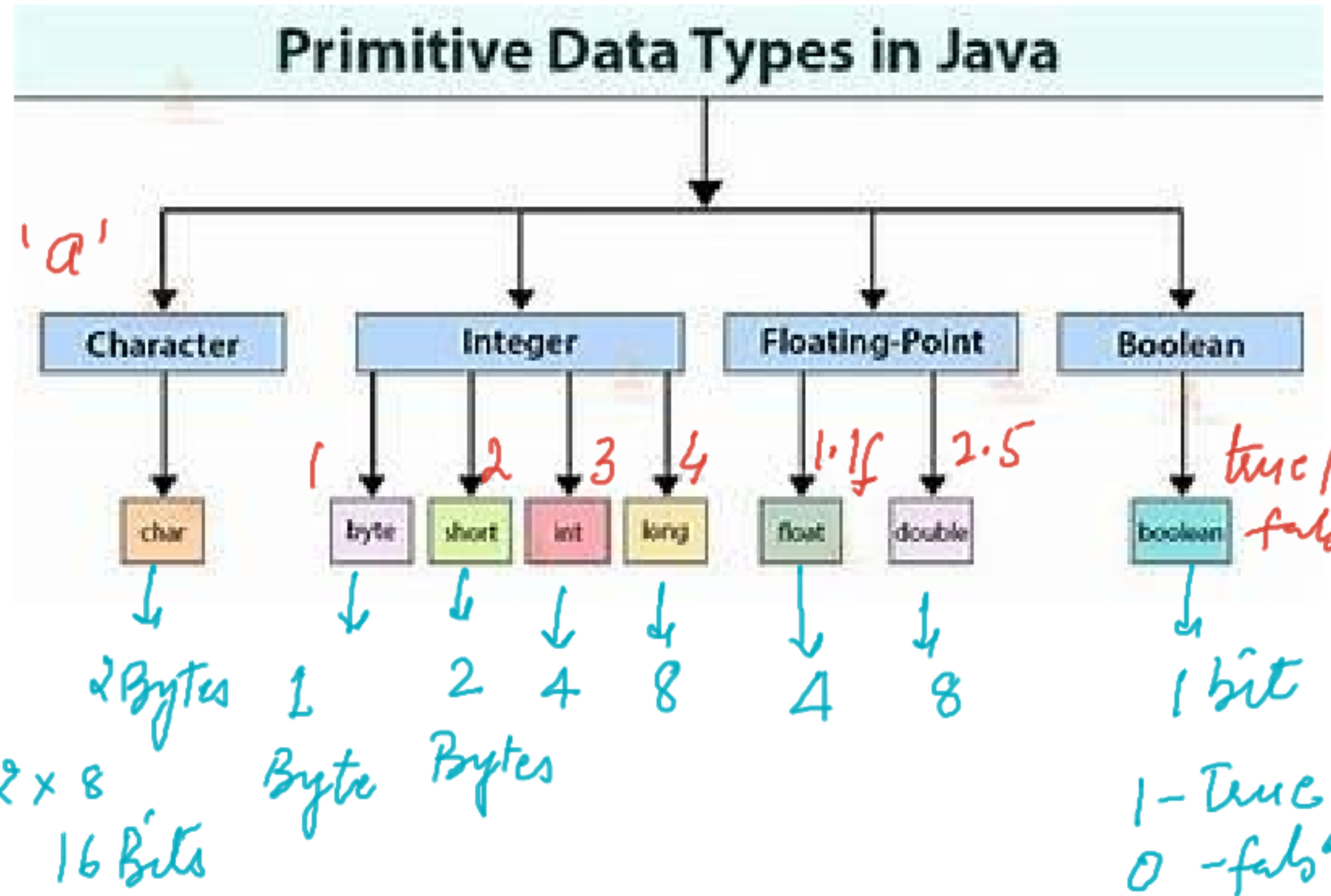
} decimal (.)

boolean → true, false
char → single char
a, b, c

eps.

Binary dump

	128	64	32	16	8	4	2	1
1					0	0	0	1
2					0	0	1	0
3			0	0	1	0	0	0



byte b = 10;

0 0 0 0 1 0 1 0
128 64 32 16 8 4 2 1

Primitive Data Types - Summary

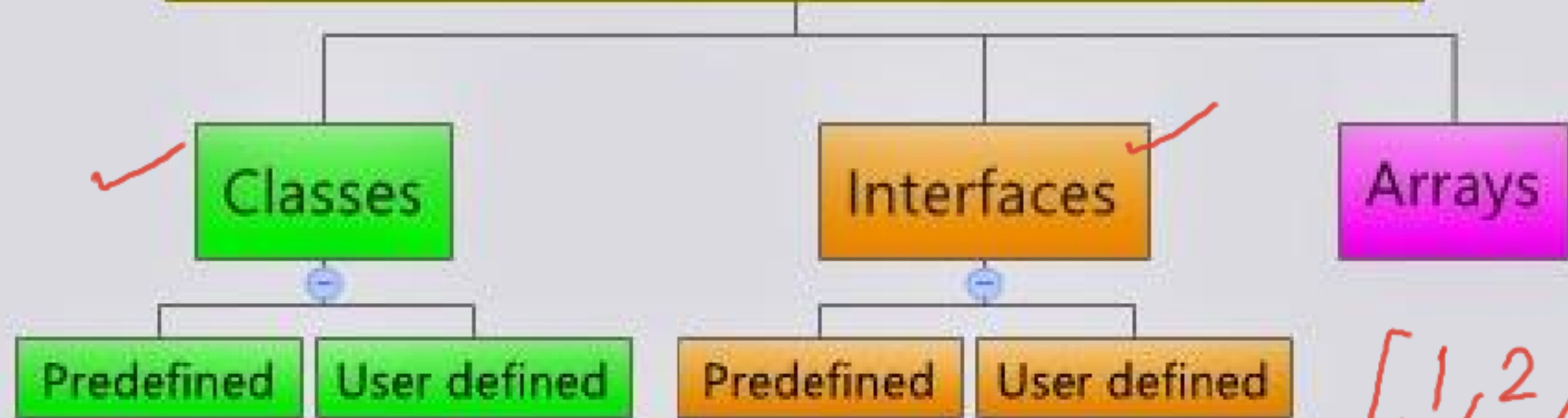
Data Type	Keyword	Size(bit)	Size(byte)	Min	Max
character	char	16	2	0	65535
byte	byte	8	1	-128	127
short	short	16	2	-32768	32767
integer	int	32	4	-2e31	-2e31 -1
float	float	32	4 x 8	1.4e-45	3.4e+38
long	long	64	8 x 8	-2e63	-2e63-1
double	double	64	8	4.9e-324	1.8e+308
boolean	boolean	NA	NA	false	true

(II) Non-primitive Data Type:

String x = "Test" → can hold multiple values.

Collection
of chrs.

Non-Primitive Data Types



var
↑

int[]

a = { 1, 2, 3, 4 }

String
Arrays =

a = [1, 2, 3, 4]

Type Casting :-

Conversion/ storing the value of one datatype into another datatype.

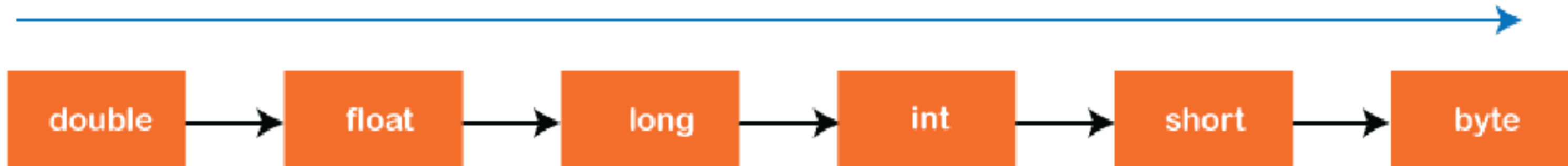
Star Cast

→ Role
↓

①

Narrowing Type Casting

(Explicit Casting)



(2)

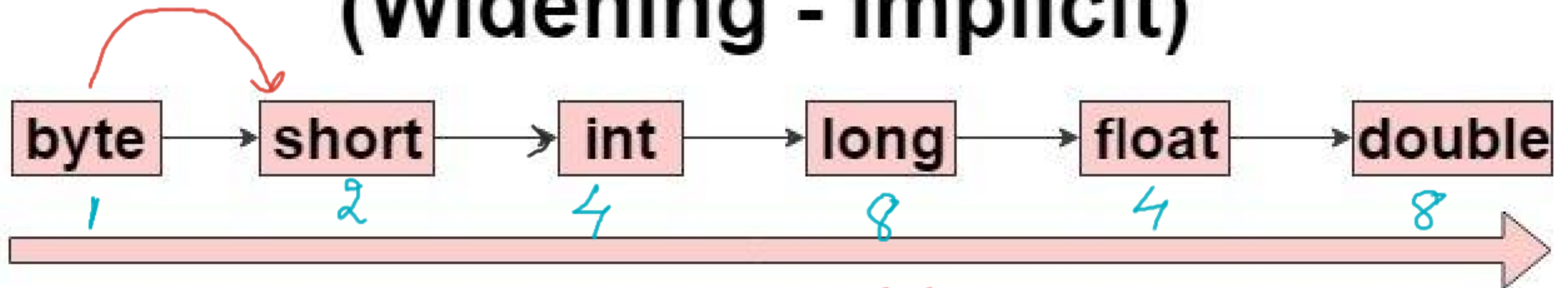
Widening Type Casting

(Implicit Casting)

Type Casting in Java

Automatic Type Conversion

(Widening - implicit)



Narrowing (explicit)

