DATA TYPES

In JAVA, every variable & every expression has some type.

Each & every data type is clearly defined.

Every assignment should be checked by compiler for type compatability.

Because of above reasons we can conclude,

Date ___/___/_____ Page No.: _____

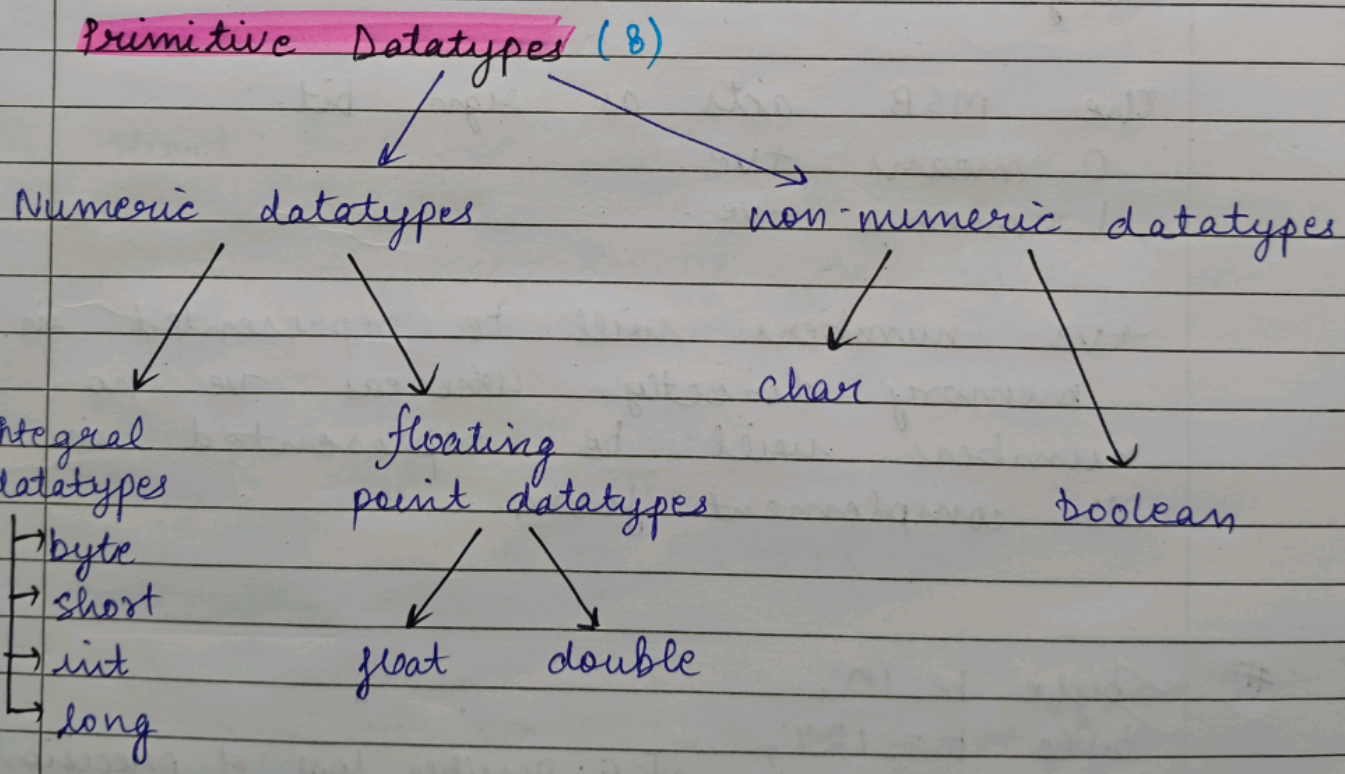
JAVA language is strongly typed programming language.

Ques → Is JAVA a pure OOP language?

* JAVA is not considered as pure OOP language bcoz several OOP feature are not satisfied by JAVA (like operator overloading, multiple inheritance, etc).

* Moreover, we are depending upon primitive datatypes which are non objects.

Primitive Datatypes (8)



Except boolean & char, remainning data types considered as signed datatype bcoz we can represent both +ve & -ve.

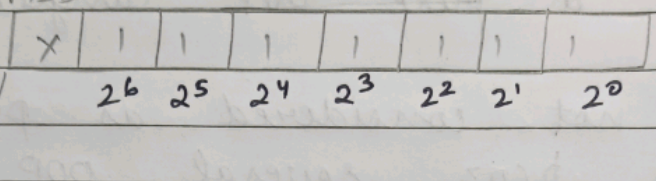
Date ___/___/_____ Page No.: _____

① **Byte :**

size 1 byte (8 bits)

Max-value : +127

Most significant bit ← MSB



sign bit

0 → +ve

1 → -ve

$64+32+16+8+4+2+1$
$= 127.$

min_value : -128 (two's complement)

Range = -128 to 127.

The MSB acts as sign bit.
0 means +ve.
1 means -ve.

+ve numbers will be represented in memory directly whereas -ve no. numbers will be represented in 2's complement form.

✓ byte b = 10;
✓ byte b = 127;
(X) byte b = 128; → [CE: Possible loss of precision
found: int
required: byte]
(X) byte b = 10.8; → [found: double
required: byte]

(X) byte b = true; → C.E : incompatible type
required : byte
found : boolean

(X) byte b = "durg"; incompatible type
found : Java.lang.String
required : byte

Streams → character
→ Byte

Byte is the best choice if we want to handle data in terms of streams either from the file or from the network.
(File & network support form is byte)

② short :

short is most rarely used data type in JAVA.

Size : 2 bytes (16 bits)
Range : -2^{15} to $2^{15}-1$
(-32768 to 32767)

Eg → ✓ short s = 32767;
(X) short s = 32768; → C.E → PLP
found : int
required : short

(X) short s = 10.5; → CE → incompatible type
found : double
required : short

Date ___/___/_____ Page No.: _____

Short data type is best suitable for 16 bit processor like 8085 but these processor are completely out dated. Hence, corresponding short datatype is also out dated data type.

(3) int :

In JAVA, most commonly used datatype

size : 4 bytes (32 bits)
Range : $(-2^{31}$ to $2^{31}-1)$
$[-2147483648$ to $2147483647]$

Eg → ✓ int x = 2147483647;

(X) int x = 2147483648;
C.E → integer number ~~too~~ large

(X) int x = 2147483648l;
C.E → possible loss of precision
found : long
required : int

30 | 11 | 24

(4) Sometimes int may not enough to hold big values, then we should go for long type.

Eg → the amt of distance travelled by light in 1000 day. To hold this

value may not enough.
So, we should go for long data type.

long l = 1,26,000 X 60 X 60 X 24 X 1000 miles;

② The number of characters present in a big file may exceed int range hence, the return type of length method is long not int.

long l = f.length();

Size: 8 bytes (64 bits)

Range: -2^{63} to $2^{63}-1$

NOTE:

All the above data types (byte, short, int, long) meant for representing integral value & if we want to represent floating point values then we should go for floating point DATA TYPES.

Floating - Point datatype

- Float
- Double

Date ___/___/_____ Page No.: _____

	Float	Double
Decimal point	→ 5 to 6	14 to 15
Accuracy	→ single precision	Double precision
Range	→ -1.7e38 to 1.7e38	-3.4e308 to 3.4e308
Size	→ 4 bytes	8 bytes

(1) If we want 5 to 6 decimal point of accuracy, then we should go for float.

If we want 14 to 15 decimal point of accuracy, then we should go for double.

(2) Float follows single precision.
Double follows double precision.

(3) Float size → 4 bytes
Double size → 8 bytes

(4) Float Range → -3.4e38 to 3.4e38
Double Range → -1.7e308 to 1.7e308

Boolean Datatype

Size → not applicable [Virtual Mission Dependent]

Range → not applicable [but allowed values are True | False]

Eg →

(✓) boolean b = true;

(X) boolean b = 0; → C.E → incompatible type
found : int
required : boolean

(X) boolean b = True; → C.E → cannot find symbol
symbol → variable True
location → class Test.

(X) boolean b = "True"; → C.E → incomptabile type
found : string
required : boolean

```
int x=0
if (x)
{ super ("Hi");
}
else
{ super ("Helo"); }
```

C.E → incompatible type
found : int
required : Boolean

```
while (1)
{
super ("Hello");
}
```

Date __/__/____ Page No.: _____

Char

size : 2 bytes
Range : 0 to 65535.

JAVA	C / C++
256 characters (ASCII) + Any worldwide language	ASCII ≤ 256 — 8 bits are enough → 1 byte
⩾ 256 ⩽ 65	a to z, A to z, 0 to 9, $, #, ↑, -, * ⋮
unicode char	
☐ ... ☐	
अ .... ह	
ಅ .... ಕ್ಷ	

Old language (like C, C++) are ASCII code based & the number of allowed different ASCII code characters are less than or equal to 256.
To represent these 256 characters 8 bits are enough.

Hence, the size of char in old languages is 1 byte.

But JAVA is unicode based & the number of different unicode characters are greater than 256 & less than or equal to 65536.
To represent these many characters

Date ___/___/_____ Page No.: _____

8 bits are not enough, compulsory we should go for 16 bits.

Hence, size of char in JAVA is 2 byte.

Summary of JAVA primitive DATA TYPES

DT	size (bytes)	Range	Wrapper class	Default value
Byte	1	-2^7 to 2^7-1	Byte	0
short	2	-2^{15} to $2^{15}-1$	Short	0
int	4	-2^{31} to $2^{31}-1$	integer	0
long	8	-2^{63} to $2^{63}-1$	long	0
float	4	-3.4e38 to 3.4e38	float	0.0
double	8	-1.7e308 to 1.7e308	double	0.0
boolean	N.A	N.A (allowed: T/F)	Boolean	false
char	2	0 to 65535	character	0 (space) character

<u>null</u> → default values for object type (reference) & we can't apply for primitive DATA types.