Lecture-2

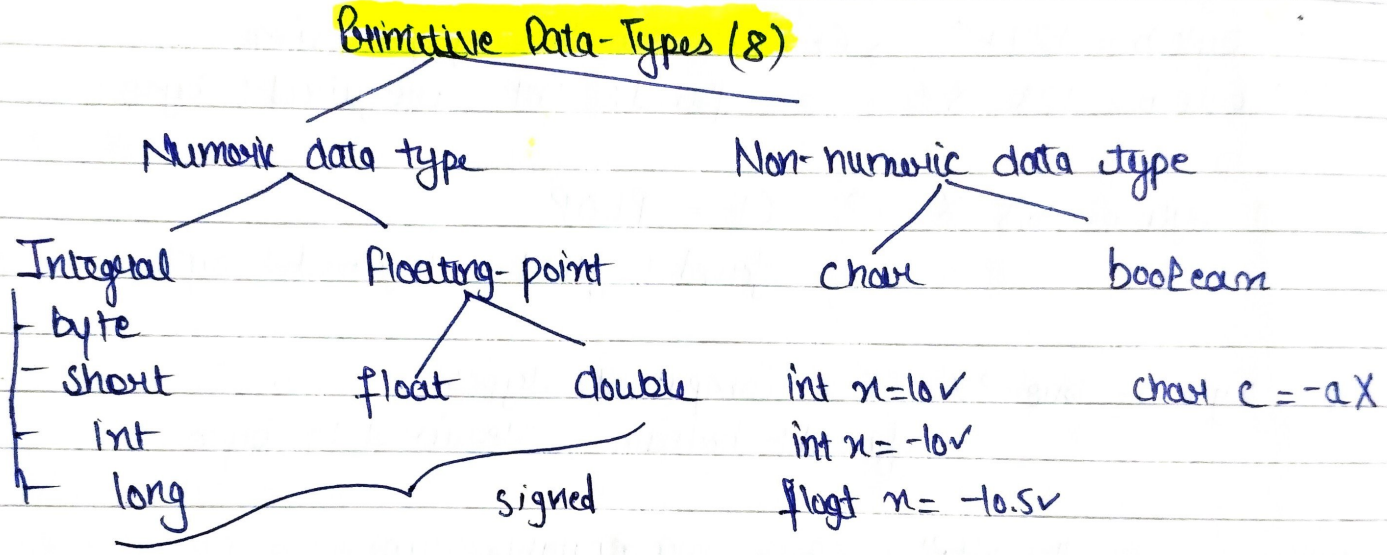
Data-Types

In Java, every variable and expression has some type.
Each and every data-type is clearly defined.
Every assignment should be checked by compiler for compatibility.
Because of above reasons, Java is strongly typed language.
bool x = 0 X int a = 10.5 X => but accetable in C

Is Java pure OOP language?

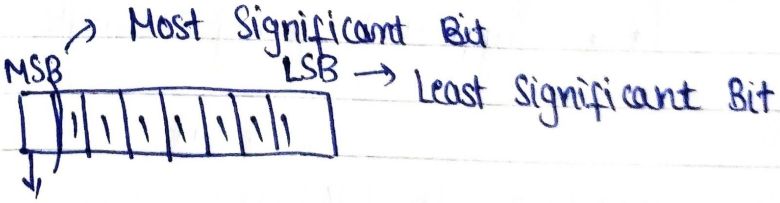
Java is not considered as pure object oriented programming language because several OOPs features are not supported by Java like operator overloading, multiple inheritance.

Moreover we are depending on primitive data types which are non-object.



⇒ Except boolean and char, remaining data-types are considered as signed data types because we can represent both +ve and -ve numbers.

byte



Size :- 1byte (8 bits)
MAX_VALUE :- +127
MIN_VALUE :- -128
Range :- -128 to +127

Signbit 0 → +ve, 1 → -ve

Max value it can store when all values in 7 bits are 1. So

$2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$

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

1. MSB acts as sign bit. 0 → positive numbers 1 → negative

2. +ve numbers will be represented directly in memory whereas -ve numbers are represented in 2's complimend form.

(i) byte b = 10 ✓
(ii) byte b = 127 ✓
(iii) byte b = 128 ✗ → CE :- Possible loss of precision
found : int required : byte

(iv) byte b = 10.5 ✗ → CE = PLOP
found : float required : byte

(v) byte b = true → ✗ CE = incompatable types
found :- boolean required :- byte

(vii) byte b = "Ankita" CE = incompatable types
found :- string required :- byte

When using byte is choice?

★ Byte is the best choice if we have to handle data in form of streams either by the file or from the network. (File supported form/ network supported form is byte).

Short

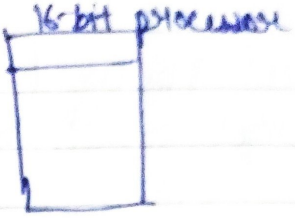
Size :- 2 bytes (16 bits)
Range :- -2^{15} to $2^{15}-1$
[-32768 to 32767]

This is the most rarely used data-type in Java.

When using short is a choice?

short data-type is best suitable for 16 bit processors But these processors are outdated now, hence corresponding short data-type is also almost outdated now.

* Earlier we used to have 16-bit processors → File reading was efficient with 16-bit data type (short).



int

The most commonly used datatype in Java is int.

Size :- 4 bytes (32 bits)

Range :- -2^{31} to 2^{31} [-2147483648 to 2147483647]

(i) int x = 2147483647 ✓

(ii) int x = 2147483648 ✗ CE :- integer number too long large (Because all numbers are treated as int)

(iii) int x = 2147483648L ✗ CE :- Possible loss of precision found :- long required :- int

(iv) int x = true ✗ CE :- incompatable types

long

Sometimes int may enough to hold long/big values then we should go for long data type.

Ex1 :- The amount of distance travelled by light in 1000 days; to hold this value int may not be enough.

long l = 1,26,000 x 60x60 x 24 x1000 miles.

Example2 The number of characters present in big file may exceed int range. Hence return type of f.length() method is long, but 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) represent integral values, if we want to represent floating values, we should go for floating-point data types.

Floating-point data types

float	double
1. If we want to have 5-6 decimal places of accuracy, then we should go for float.	→ If we want 14-15 decimal places of accuracy → go for double.
2. Float follows single precision	⇒ double precision.
3 size:- 4 bytes	⇒ size ⇒ 8 bytes
4. Range:- $-3.4e^{38}$ to $3.4e^{38}$	⇒ $-1.7e^{308}$ to $1.7e^{308}$

boolean

Size: Not Applicable [Virtual Machine dependent]
Range:- " " [but allowed values are true/false]

```
boolean b = true ✓
boolean b = 0 X   CE = incompatible type
                       found : int   required : boolean
```

boolean b = True ✗ CE = cannot find symbol
boolean b = "True" ✗ CE = incompatable data type

```
int x = 1;
if(x){
System.out.print("Hi");
}
```

In C++ ⇒ Hi
In Java ⇒ Error.

CE:
incompatable
data-type
found: int
required: boolean

```
while(1)
{ System.out.print("Hi");
}
```

↓,
In C++ it will print infinite times "Hi" but in Java it will give CE

char

⇒ language like C/C++ are ASCII code based and no. of allowed diff ASCII code values are <=256, To represent these 256 characters, 8 bits are enough hence size of char for these language is 8 bits | 1 byte. But in Java, number of Unicode characters are >256 and <= 65536. To represent these many characters 8 bit many not enough so we have to go for 16 bits | 2 byte.

Size :- 2 byte
Range :- 0 to 65535

char ch = null; CE :- incompatable data type

null is a default value for object reference and we can't apply it for primitive data-type. But if we still try to use it for primitive, it will give compile time error.

Summary of data type

	data-type	size	Range	Wrapper class	default value
1.	byte	1 byte	-2^7 to 2^7-1	Byte	0
2.	short	2 bytes	-2^{15} to $2^{15}-1$	Short	0
3.	int	4	-2^{31} to $2^{31}-1$	Integer	0
4.	long	8	-2^{63} to $2^{63}-1$	Long	0L
5.	float	4	$-3.4e38$ to $3.4e38$	Float	0.0
6	double	8	$-1.7e308$ to $1.7e308$	Double	0.0
7.	boolean	NA	NA	Boolean	false (For C/C++ - true)
8.	char	2	0 to 65535	Character	0 [represents space character]

~ Ankita Bansal