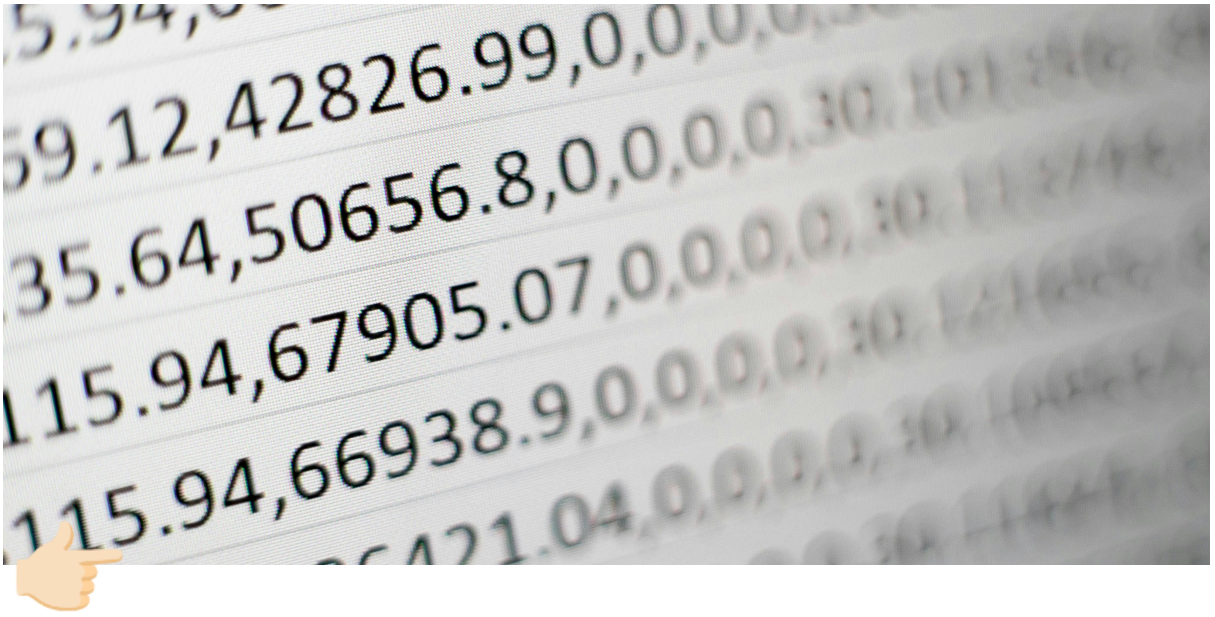
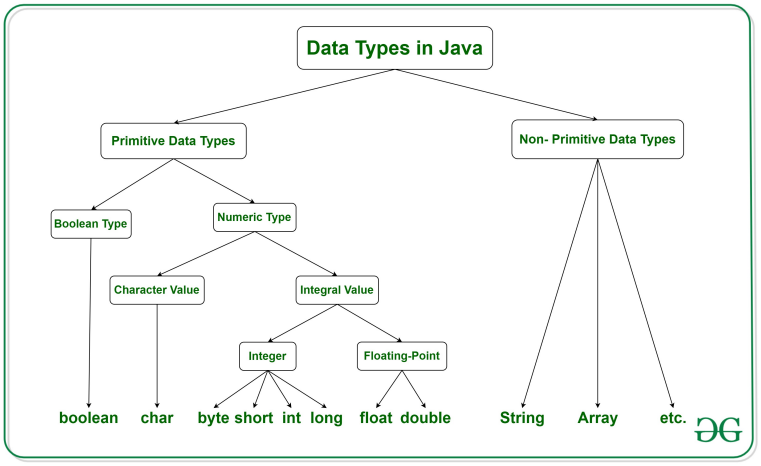


Data Types in Java

Java is statically typed and also a strongly typed language because, in Java, each type of data is predefined as part of the programming language and all constants or variables defined for a given program must be described with one of the Java data types.

Data types specify the different sizes and values that can be stored in the variable.

1. **Primitive data types:** The primitive data types include boolean, char, byte, short, int, long, float and double.
2. **Non-primitive data types:** The non-primitive data types include Classes, Interfaces, and Arrays.



Primitive Data Types

In Java language, primitive data types are the building blocks of data manipulation. These are the most basic data types available in Java language.

Java is a statically-typed programming language. It means, all variables must be declared before its use. That is why we need to declare variable's type and name.

Type	Size	Default	Range of values
boolean	1 bit	false	true, false
byte	8 bits	0	-128 to 127
char	16 bits	\u0000	ASCII values 0 to 255
short	16 bits	0	-32768 to 32767
int	32 bits	0	-2147483648 to 2147483647
long	64 bits	0	-9223372036854775808 to 9223372036854775807
float	32 bits	0.0	upto 7 decimal digits
double	64 bits	0.0	upto 16 decimal digits

Boolean Data Type

Boolean data type represents only one bit of information **either true or false.**

```
// Synatx
boolean isJavaFun = true;
```

Numbers

Primitive number types are divided into two groups:

Integer types stores whole numbers, positive or negative (such as 123 or -456), without decimals. Valid types are `byte`, `short`, `int` and `long`.

Floating point types represents numbers with a fractional part, containing one or more decimals. There are two types: `float` and `double`.

Byte Data Type

The `byte` data type can store whole numbers from -128 to 127. This can be used instead of `int` or other integer types to save memory when you are certain that the value will be within -128 and 127:

```
// Syntax
byte byteVar;
byte a = 10;
byte b = -20;
```

Size: 1 byte (8 bits)

Short Data Type

The `short` data type can store whole numbers from -32768 to 32767. Similar to byte, use a short to save memory in large arrays.

```
// Syntax
short myNum = 5000;
```

Int Data Type

The `int` data type can store whole numbers from -2147483648 to 2147483647. In general, and in our tutorial, the `int` data type is the preferred data type when we create variables with a numeric value.

```
//Syntax
int myNum = 100000;
```

Long Data Type

The `long` data type can store whole numbers from -9223372036854775808 to 9223372036854775807. This is used when int is not large enough to store the value. Note that you should end the value with an "L"

```
// Synatx
long myNum = 15000000000L;
```

Float Data Type

The float data type is a single-precision 32-bit IEEE 754 floating-point. Use a float (instead of double) if you need to save memory in large arrays of floating-point numbers.

```
// Syntax
float myNum = 5.75f;
```

Double Data Type

The double data type is a double-precision 64-bit IEEE 754 floating-point.

```
// Syntax
double myNum = 19.99d;
```

Characters

Char Data Type

The `char` data type is used to store a **single** character. The character must be surrounded by single quotes, like 'A' or 'c':

```
char myGrade = 'B';
```

Non-Primitive Data Type or Reference Data Types

Non-primitive data types are called **reference types** because they refer to objects.

1. Strings

The `String` data type is used to store a sequence of characters (text). String values must be surrounded by double quotes.

Strings are defined as an array of characters. The difference between a character array and a string in Java is, that the string is designed to hold a sequence of characters in a single variable whereas, a character array is a collection of separate char-type entities.

```
String name = "Aditya Singh";
```

2.Class

3. Object

4. Interface →(Interfaces are used to achieve abstraction in Java)

5. Array

The main difference between **primitive** and **non-primitive** data types are:

- Primitive types are predefined (already defined) in Java. Non-primitive types are created by the programmer and is not defined by Java (except for `String`).
- Non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.
- A primitive type has always a value, while non-primitive types can be `null`.
- A primitive type starts with a lowercase letter, while non-primitive types starts with an uppercase letter.