Data type (basic type) refers to type and size of data associated with variables and functions.

Data type is used for declaration of memory location of variable which determines the features of data.

In Kotlin, everything is an object, which means we can call member function and properties on any variable.

```
val myNum = 5                // Int
val myDoubleNum = 5.99       // Double
val myLetter = 'D'           // Char
val myBoolean = true         // Boolean
val myText = "Hello"         // String
```

Kotlin built in data type are categorized as following different categories:

Number
Character
Boolean
Array
String

(1) Number Types

Number types of data are those which hold only number type data variables.

It is further categorized into different Integer and Floating point.

Data Type	Bit Width (Size)	Data Range
Byte	8 bit	-128 to 127
Short	16 bit	-32768 to 32767
Int	32 bit	-2,147,483,648 to 2,147,483,647
Long	64 bit	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
Float	32 bit	1.40129846432481707e-45 to 3.40282346638528860e+38
Double	64 bit	4.94065645841246544e-324 to 1.79769313486231570e+308

(a) Byte

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:

Example:-

```
val myNum: Byte = 100
println(myNum)
```

(b) Short

The Short data type can store whole numbers from -32768 to 32767:

Example:-

```
val myNum: Short = 5000
 println(myNum)
```

(c) Int

The Int data type can store whole numbers from -2147483648 to 2147483647:

Example:-

```
val myNum: Int = 100000
println(myNum)
```

(Note) Difference Between Int and Long

A whole number is an Int as long as it is up to 2147483647.

If it goes beyond that, it is defined as Long:

Example:-

```
val myNum1 = 2147483647  // Int
val myNum2 = 2147483648  // Long
```

(d) Long

The Long data type can store whole numbers from -9223372036854775807 to 9223372036854775807.

This is used when Int is not large enough to store the value.

Optionally, you can end the value with an "L":

Example:-

```
val myNum: Long = 15000000000L
println(myNum)
```

(e)Floating Point Types

Floating point types represent numbers with a decimal, such as 9.99 or 3.14515.

The Float and Double data types can store fractional numbers:

Float Example:-

```
val myNum: Float = 5.75F
println(myNum)
```

Scientific Numbers

A floating point number can also be a scientific number with an "e" or "E" to indicate the power of 10:

Example:-

```
val myNum1: Float = 35E3F
val myNum2: Double = 12E4
println(myNum1)
println(myNum2)
```

(NOTE) Use Float or Double?

The precision of a floating point value indicates how many digits the value can have after the decimal point.

The precision of Float is only six or seven decimal digits,

while Double variables have a precision of about 15 digits.

Also note that you should end the value of a Float type with an "F".

(d)double

```
 val myNum: Double = 19.99
println(myNum)
```

(2) Character (Char) Data Type

Characters are represented using the keyword Char.

Char types are declared using single quotes (' ').

Example:-

```
val myGrade: Char = 'B'
 println(myGrade)
```

Data Type	Bit Width (Size)	Data Range
Char	4 bit	-128 to 127

(NOTE) Unlike Java, you cannot use ASCII values to display certain characters.
The value 66 would output a "B" in Java, but will generate an error in Kotlin:
In Kotlin:-

```
val myLetter: Char = 66
println(myLetter) // Error
```

in java:-

```
val myLetter: Char = 66
println(myLetter) //No Error
```

(3) Boolean Data Types :-
Boolean data is represented using the type Boolean.
It contains values either true or false.

Data Type	Bit Width (Size)	Data Value
Boolean	1 bit	true or false

(4) Array:-

Arrays in Kotlin are represented by the Array class.
Arrays are created using library function arrayOf() and Array() constructor.
Array has get (), set() function, size property as well as some other useful member functions.

Creating Array using library function arrayOf()
The arrayOf() function creates array of wrapper types.
The item value are passed inside arrayOf() function like arrayOf(1,2,3) which creates an array[1,2,3].

The elements of array are accessed through their index values (array[index]). Array index are start from zero.

```
val id = arrayOf(1,2,3,4,5)
val firstId = id[0]
val lasted = id[id.size-1]
```

Creating Array using Array() constructor
Creating array using Array() constructor takes two arguments in Array() constructor:

First argument as a size of array, and
Second argument as the function, which is used to initialize and return the value of array element given its inde

(5) String:-

String in Kotlin is represented by String class. String is immutable,
which means we cannot change the elements in String.
String declaration:

```
val text ="Hello, JavaTpoint"
```

Types of String

String are categorize into two types. These are:

1. Escaped String: Escape String is declared within double quote (" ") and
may contain escape characters like '\n', '\t', '\b' etc.

```
val text1 ="Hello, JavaTpoint"
//or
val text2 ="Hello, JavaTpoint\n"
//or
val text3 ="Hello, \nJavaTpoint"
```

2. Raw String: Row String is declared within triple quote (""" """).
It provides facility to declare String in new lines and contain multiple lines.
Row String cannot contain any escape character.

```
val text1 ="""   Welcome  To JavaTpoint   """
```