Intro to R - Data Types and Data Structures - 1

One should look for what is and not what he thinks should be. (Albert Einstein)

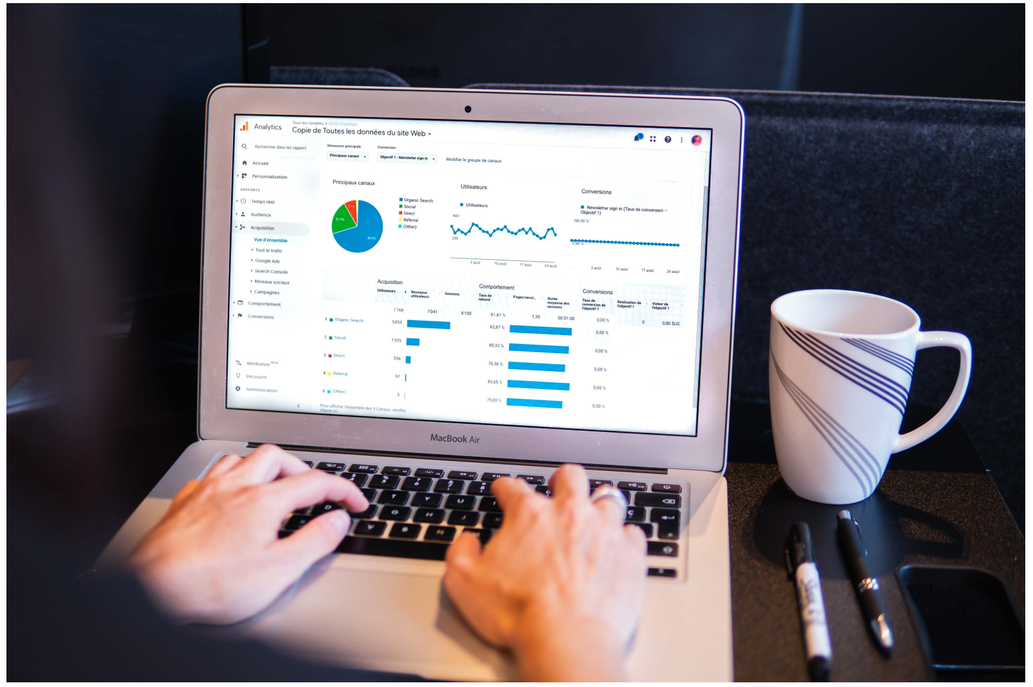
Data Types and Data Structures: Topic introduction

In this part of the course, we will cover the following concepts:

- Overview of R data types and data structures
- Lists, vectors, matrices, and dataframes

Warm Up

- Let’s have a short conversation to get the day going
- What kinds of data do you come into contact with most frequently?
- Leave an example in the virtual chat

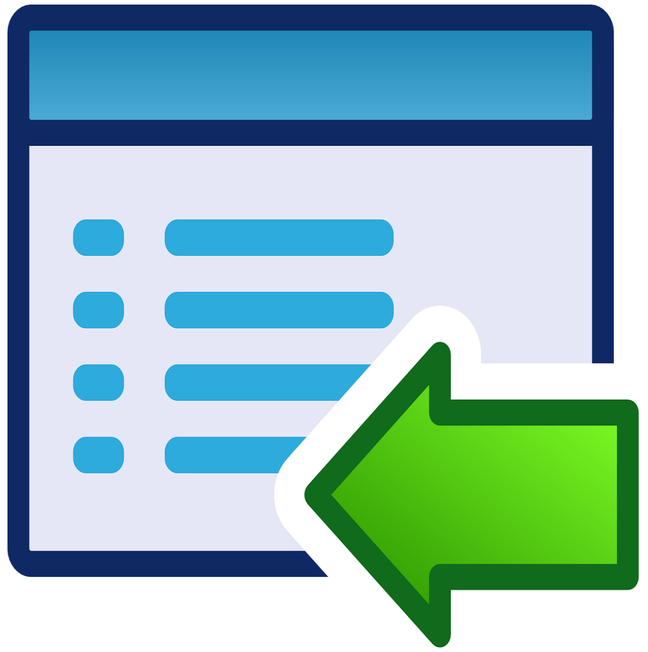


Module completion checklist

Objective	Complete
Distinguish data types (`integer`, `character`, `numeric`, and `logical`)	
Identify basic data structures in R	

Data type

- A data type is a set of data values with common characteristics, and based on which we can form expressions and functions
- The data type defines how data can be processed and how it can be stored
- For instance, a web page is a data type, insofar as all web pages share the following characteristics:
  - Address
  - Layout
  - Data
  - Integration with other web pages into a web site
  - Community which allows people to update its content
  - Web server storage



Basic data classes and types

- In R, we can refer to a few generic data classes based on how R handles objects, or more specific data types
- In most cases, there is no need to distinguish between the two
- The most important thing is to **convert** data to the specific type that meets your analytical or programming needs

Data class (high level)	Data type (low level)	Example
Integer	Integer	`-1, 5, or 1L, 5L`
Numeric	Double, float	`2.54`
Character	Character	`“Hello”`
Logical	Logical	`TRUE, FALSE`

Basic data classes: what we will use

- Depending on the tasks we need to perform, we may need to keep track of what type of data we are working with or return other properties of the data
- Let's get some practice with the following functions, which will return different results based on the data type

Item	Purpose
Value	Example of class
`typeof()`	Finds the type of the variable
`class()`	Returns the class of the variable
boolean function	Specific function that checks class and returns TRUE or FALSE
`attributes()`	Checks the metadata/attribute of the variable
`length()`	Checks the length of the object

Basic data classes: integer

- In R, integers are whole numbers that can be positive or negative
- **Note:** The **L** suffix explicitly marks any number as an integer

Item	Integer
Value	34, 34L
`typeof()`	integer
`class()`	integer
boolean function	`is.integer()`
`attributes()`	`NULL`
`length()`	1

```
# Create an integer type variable.
integer_var = 34L

# Check type of variable.
typeof(integer_var)
```

```
[1] "integer"
```

```
# Check class of variable.
class(integer_var)
```

```
[1] "integer"
```

```
# Check attributes of variable
attributes(integer_var)
```

```
NULL
```

```
# Check if the variable is integer.
is.integer(integer_var)
```

```
[1] TRUE
```

```
# Check length of variable
# (i.e. how many entries).
length(integer_var)
```

```
[1] 1
```

Basic data classes: numeric

- Decimal values are called numerics in R
- When R stores these values, it stores them as double-precision floating-point numbers
- Remember to use the **L** suffix to coerce them into integers if needed

Item	Numeric
Value	24.24
`typeof()`	double
`class()`	numeric
boolean function	`is.numeric()`
`attributes()`	`NULL`
`length()`	1

```
# Create a numeric class variable.
numeric_var = 24.24
typeof(numeric_var)
```

```
[1] "double"
```

```
# Check class of variable.
class(numeric_var)
```

```
[1] "numeric"
```

```
# Check attributes of variable
attributes(numeric_var)
```

```
NULL
```

```
# Check if the variable is numeric
is.numeric(numeric_var)
```

```
[1] TRUE
```

```
# Check length of variable
length(numeric_var)
```

```
[1] 1
```

Basic data classes: character

- In R, a sequence of characters can include letters, numbers, and/or symbols
- This data type is different from the others in that they are enclosed in single or double quotation marks
  - Both ends of the string must have the same type of quotation mark
- **Note:** if a character string does not begin with a # and/or is not enclosed in quotation marks, it will throw an error

```
# Create a character class variable
character_var = "Hello" # using double quotation
marks
character_var2 = 'Hello' # using single quotation
marks
```

Basic data classes: character (contd)

Item	Character
Value	“Hello”
typeof()	character
class()	character
boolean function	is.character()
attributes()	NULL
length()	1

```
# Create a character class variable.
character_var = "Hello"
```

```
# Check if the variable is character.
is.character(character_var)
```

```
[1] TRUE
```

```
# Check class of variable.
class(character_var)
```

```
[1] "character"
```

```
# Check metadata/attributes of variable.
attributes(character_var)
```

```
NULL
```

```
# Check length of variable
# (i.e. how many entries).
length(character_var)
```

```
[1] 1
```

Some useful character operations

- Data of the character type is often **challenging** to prepare for analysis, but the following operations can help

```
# Create another character class variable.
case_study = "JUmbLEd CaSE"

# Convert a character string to lower case.
tolower(case_study)
```

```
[1] "jumbled case"
```

```
# Convert a character string to upper case.
toupper(case_study)
```

```
[1] "JUMBLED CASE"
```

```
# Count number of characters in a string.
nchar(case_study)
```

```
[1] 12
```

```
# Compare to the output of the `length` command.
length(case_study)
```

```
[1] 1
```

```
# Get just a part of character string.
substr(case_study, #<- original string
       1,          #<- start index of substring
       7)          #<- end index of substring
```

```
[1] "JUmbLEd"
```

- Every character in the string is **indexed** with a numerical value beginning with **1**

Basic data classes: logical

Item	Logical
Value	`TRUE` or `FALSE`
`typeof()`	logical
`class()`	logical
boolean function	`is.logical()`
`attributes()`	`NULL`
`length()`	1

```
# Create a logical class variable.
logical_var = TRUE

# Check type of variable.
typeof(logical_var)
```

```
[1] "logical"
```

- If you're new to programming, the logical type may seem unusual

```
# Check if the variable is logical.
is.logical(logical_var)
```

```
[1] TRUE
```

```
# Check metadata/attributes of variable.
attributes(logical_var)
```

```
NULL
```

```
# Check length of variable
# (i.e. how many entries).
length(logical_var)
```

```
[1] 1
```

- However, it is one of the most useful types for creating **programmatic rules** based on whether an observation does or does not satisfy a condition

Basic data classes: summary & conversion

- Finally, let's discuss data type conversions

Item	Integer	Numeric	Character	Logical
Value	`34, 34L`	`24.24`	Hello	`TRUE` or `FALSE`
`typeof()`	integer	double	character	logical
`class()`	integer	numeric	character	logical
boolean function	`is.integer()`	`is.numeric()`	`is.character()`	`is.logical()`
`attributes()`	`NULL`	`NULL`	`NULL`	`NULL`
`length()`	1	1	1	1
To convert a variable to this type	`as.integer()`	`as.numeric()`	`as.character()`	`as.logical()`

Data type conversions

- Here's an example of how we can convert numeric variables to integers

```
# Create a numeric class variable.
numeric_var = 24.24
typeof(numeric_var)
```

```
[1] "double"
```

```
# Convert numeric variable to integer
a = as.integer(numeric_var)
print(a)
```

```
[1] 24
```

- Let's now try to convert a numeric variable to a character variable and look at the output

```
# Convert numeric variable to character
char_variable = as.character(numeric_var)
print(char_variable)
```

```
[1] "24.24"
```

- We observe that the variable is now enclosed in quotes which denotes that it's a character

Module completion checklist

Objective	Complete
Distinguish data types (`integer`, `character`, `numeric`, and `logical`)	✔
Identify basic data structures in R	

Basic data structures

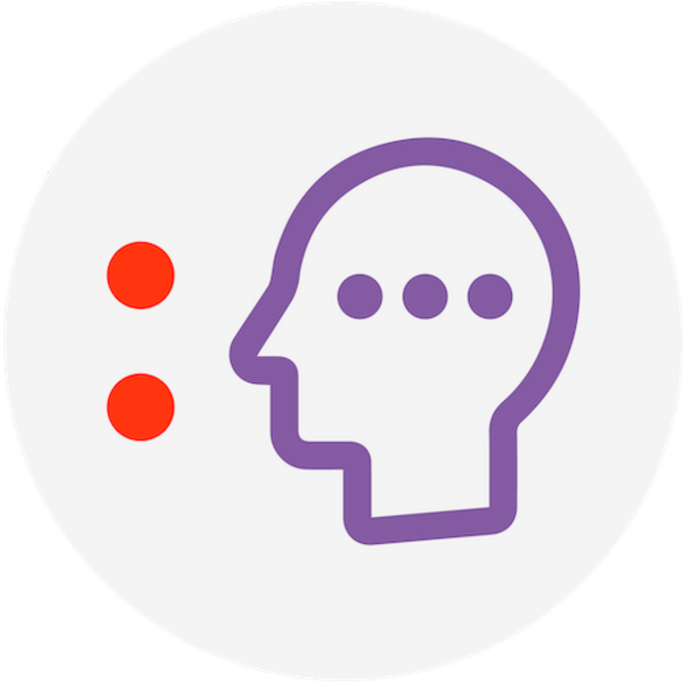
- So far, we have learned some of the most basic as well as most common data types
- Next, we are going to focus on groupings of one or more data types organized in various ways, otherwise known as data structure
- A data structure is a method for describing how data is organized based on what data types it can contain
- Certain operations and algorithms can only interpret certain data structures

Basic data structures (cont'd)

Data structure	Number of dimensions	Single data type	Multiple data types
Vector (Atomic vector)	1 (entries)	✔	✘
Vector (List)	1 (entries)	✔	✔
Matrix	2 (rows and columns)	✔	✘
Data frame	2 (rows and columns)	✔	✔

- We will explore each kind of data structure in detail in next module!

Knowledge check



Module completion checklist

Objective	Complete
Distinguish data types (`integer`, `character`, `numeric`, and `logical`)	✔
Identify basic data structures in R	✔

Congratulations on completing this module!

You are now ready to try Tasks 1-2 in the Exercise for this topic