INTRODUCTION TO RSTUDIO

Adelaide Code Club

24/03/2022

OUTLINE FOR TODAY

- What is R and RStudio?
- Basics of R
  - Operators
  - Variables
  - Vectors
  - Functions
- Practise using R

WHAT IS R AND RSTUDIO

- R is a programming language for statistical computing.
- R is powerful, and fast.
- If you open R, just get a console.
- RStudio provides
  - a text editor
  - history of commands
  - an "environment" where you can see the objects and elements you have created
  - help functions
  - a panel to view plots.

R BASICS

- Operators
- Variables
- Vectors
- Other data formats
- Functions

OPERATORS

- Arithmetic operators – return a number
  - Regular operators `(+ - * / ^)`
  - `x %%% y` – modulus (x mod y)
  - `x %/% y` – integer division

OPERATORS

- Arithmetic operators – return a number
  - Regular operators `(+ - * / ^)`
  - `x %%% y` – modulus (x mod y)
  - `x %/% y` – integer division

Regular logical operators – return a logical `(True/False)`

- Regular `(<, <=, >, >=, ==, !=)`
- `!x` – not x
- `x|y` – x OR y
- `x & y` – x AND y

OPERATORS

- Arithmetic operators – return a number
  - Regular operators `(+ - * / ^)`
  - `x %%% y` – modulus (x mod y)
  - `x %/% y` – integer division

Regular logical operators – return a logical `(True/False)`

- Regular `(<, <=, >, >=, ==, !=)`
- `!x` – not x
- `x|y` – x OR y
- `x & y` – x AND y

Test it out!

VARIABLES

- A variable is a value that you assign a name to.
  - The names can be whatever you want, they cant have spaces but can include _ or ‘ . ’. `( x , sample.one , s_1)`

VARIABLES

- A variable is a value that you assign a name to.
  - The names can be whatever you want, they cant have spaces but can include _ or ‘ . ’. `( x , sample.one , s_1)`
- Assign values using ‘<–’ or ‘=‘

```
var <- 40
```

VARIABLES

- A variable is a value that you assign a name to.
  - The names can be whatever you want, they cant have spaces but can include _ or ‘ . ’. `( x , sample.one , s_1)`
- Assign values using ‘<–’ or ‘=‘

```
var <- 40
```

- You can return the value of the variable to typing its name

VARIABLES

- A variable is a value that you assign a name to.
  - The names can be whatever you want, they cant have spaces but can include _ or ‘ . ’. `( x , sample.one , s_1)`
- Assign values using ‘<–’ or ‘=‘

```
var <- 40
```

- You can return the value of the variable to typing its name
- Can use this variable as you would that value

```
## make percentage:  (# character allows you to add comments to code)
var / 100
```

VARIABLES CONTINUED…

- Can also use variables to create other variables

```
var.2 <- var /100
```

VARIABLES CONTINUED…

- Can also use variables to create other variables

```
var.2 <- var /100
```

- If you change the value of variable, it won't automatically change values of variables that's used it.
- If `var = 80, var.2` wont be 0.8 unless reassign value.

VECTORS

- A vector is a collection of numbers. Each individual number is referred to as an element. (A single value is technically a vector of length 1.)

VECTORS

- A vector is a collection of numbers. Each individual number is referred to as an element. (A single value is technically a vector of length 1.)
- To create a vector need to use `c()` which means to combine.

```
vec <- c(1, 2, 3, 4, 5)
```

This can also be done with `vec = c(1:5)`

VECTORS

- A vector is a collection of numbers. Each individual number is referred to as an element. (A single value is technically a vector of length 1.)
- To create a vector need to use `c()` which means to combine.

```
vec <- c(1, 2, 3, 4, 5)
```

  This can also be done with `vec = c(1:5)`

- You can often use vectors as you would a single number

```
vec * 10
```

VECTORS

- A vector is a collection of numbers. Each individual number is referred to as an element. (A single value is technically a vector of length 1.)
- To create a vector need to use `c()` which means to combine.

```
vec <- c(1, 2, 3, 4, 5)
```

  This can also be done with `vec = c(1:5)`

- You can often use vectors as you would a single number

```
vec * 10
```

- You can also have vectors of character strings

```
char_vec<- c("one", "two", "three", "four")
```

INDEXING VECTORS

- Indexing allows you to extract certain values from a vector. Index using [ ]

INDEXING VECTORS

- Indexing allows you to extract certain values from a vector. Index using [ ]
- ** R is a `1`-based system, meaning the first value is value 1. Other languages can be 0-based meaning the first value is value 0 and value 1 is actually the second value.

INDEXING VECTORS

- Indexing allows you to extract certain values from a vector. Index using [ ]
- ** R is a `1`-based system, meaning the first value is value 1. Other languages can be 0-based meaning the first value is value 0 and value 1 is actually the second value.

```
Extract a value → vec[1]
```

INDEXING VECTORS

- Indexing allows you to extract certain values from a vector. Index using [ ]
- ** R is a 1-based system, meaning the first value is value 1. Other languages can be 0-based meaning the first value is value 0 and value 1 is actually the second value.

```
Extract a value → vec[1]

Reassign a value → vec[2] <- 20
```

INDEXING VECTORS

- Indexing allows you to extract certain values from a vector. Index using [ ]
- ** R is a 1-based system, meaning the first value is value 1. Other languages can be 0-based meaning the first value is value 0 and value 1 is actually the second value.

```
Extract a value → vec[1]

Reassign a value → vec[2] <- 20

Extract a slice of a vector → vec_s[2:4]
```

OTHER DATA FORMATS

- lists
- matrix
- data.frames

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`
- `str()` – describes the structure of an object

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`
- `str()` – describes the structure of an object
- `length()` – will return the number of elements

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`
- `str()` – describes the structure of an object
- `length()` – will return the number of elements
- `max()`

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`
- `str()` – describes the structure of an object
- `length()` – will return the number of elements
- `max()`
- `min()`

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof()` – determines the type of any object
- `str()` – describes the structure of an object
- `length()` – will return the number of elements
- `max()`
- `min()`
- `mean()`

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof()` – determines the type of any object
- `str()` – describes the structure of an object
- `length()` – will return the number of elements
- `max()`
- `min()`
- `mean()`
- `sum()`

FUNCTIONS

- Functions enable you to perform a specific task without you having to code it yourself.
  - You are also able to create your own functions
- `typeof() – determines the type of any object`
- `str()` – describes the structure of an object
- `length()` – will return the number of elements
- `max()`
- `min()`
- `mean()`
- `sum()`

If you don't know have to use a function type `?function_name` in the console

EXERCISE

- Download R script from Adelaide Code Club git repository.
- To do this, go to the terminal and move into the directory using cd
  - On my machine this is cd ~/2022_Adelaide_Code_Club
- Once in the directory use ‘git pull’ to download a copy of the files added to the repository.
- We will be using the file intro_to_r.R

R SCRIPTS

- To execute a line of code in an R script use:
  - Cmd + enter for macs
  - Ctrl + enter for windows
  - This runs the line the cursor is in and then moves the cursor to the next line
- To execute the file:
  - Use the run button at the top of the script window
  - cmd + shift + enter (mac)
  - ctrl + shift + enter (windows)

SUMMARY

- R and Rstudio
- Create variables and vectors
- Index vectors
- Utilise basic functions : `typeof()`, `str()`, `length()`, `max()`, `min()`, and `mean()`