R Programming Basics (Part 1)

Zhaoqi Wang

2025-12-12

Contents

1 Download R and RStudio

- Download R and RStudio
  - https://www.r-project.org
  - https://posit.co/downloads/

2 Basics

2.1 R Language Syntax

```
"Hello World!"
## [1] "Hello World!"
```

```
print("hello world!")
## [1] "hello world!"
```

```
5 + 5
## [1] 10
```

```
2*1
## [1] 2
```

```
5/4
## [1] 1.25
```

```
2^16 #exponent
## [1] 65536
```

```
# Built-in Math Functions
max(5, 10, 15)
## [1] 15
```

```
min(5, 10, 15)
## [1] 5
```

```
sqrt(16)
## [1] 4
```

```
abs(-4.7)
## [1] 4.7
```

```
ceiling(1.4)
## [1] 2
```

```
floor(1.4)
## [1] 1
```

```
# mean(x, na.rm= FALSE )
# sd(x)
# median(x)
```

```
# Assignment
# Use the "<-" operator
# on the left is the name
# on the right is the data you want assigned to the name
x <- 1
y <- "Hey there"
name <- "John"
```

```
age <- 40
name
## [1] "John"
```

```
age
## [1] 40
```

```
print(name)
## [1] "John"
```

Scripts, You can work in scripts for interactive programming as we have in this lecture. To open a new script, click “File” » “New File” » “R Script” you can also run these .R script files.

Run Code in a Script Window You can also just run a single line of your script by moving your cursor to the line you want to run and clicking ctrl+enter together. You can run multiple lines by selecting all the lines you want to run, and clicking ctrl+enter together.

2.2 Concatenate Elements

```
text <- "awesome"
paste("R is", text)
## [1] "R is awesome"
```

```
text1 <- "R is"
text2 <- "awesome"
paste(text1, text2)
## [1] "R is awesome"
```

```
num1 <- 5
num2 <- 10
num1 + num2
## [1] 15
```

2.3 Variable Names

```
# Legal variable names:
myvar <- "John"
my_var <- "John"
myVar <- "John"
MYVAR <- "John"
myvar2 <- "John"

# Illegal variable names:
# 2myvar <- "John"
# my-var <- "John"
# my var <- "John"
# _my_var <- "John"
# my_v@ar <- "John"
# TRUE <- "John"
```

There are rules about naming objects.

```
# Your object name must start with a letter (upper or lower)
y1 <- 1
y_1 <- 1
```

```
y.1 <- 1
# if you update an object, perhaps append numbers with a "." or "_"
# it cannot start with a number
# 1y <- 1 (This is Wrong)
# there are a bunch of protected special characters.
# object named cannot include these characters ^ !@#$%^&*()
```

assign numeric value

```
# In this context, you technically could use the "=" symbol for assignment.
x = 1
x <- 1
# but in other contexts you cannot,
# and in some context you'll only want to use the "=" equal sign.
# so, for assignment, just stick wtih the "<-" operator.
```

2.4 Comments

```
# Note also that the "#" pound symbol allow for comments in your script.
# R will ignore any lines in the console or your scripts after a #, to the end of the line.
x <- 1 #so you can comment after code too
```

2.5 *In Class Applications*

In an R script (.r) files, practice the following two tasks:

Task 1

Use the "paste(text1, text2)" syntax we just learned to print the following sentence: "I find R interesting and I am willing to learn more about it." Note that you need to use four "texts" for the above four different colored parts.

Task 2

Calculate the following simple math and interpret the results: 3^4; 21 ÷ 4; 23 %/% 6; 4 %% 3.

3 Data Types

```
my_var <- 30 # my_var is type of numeric
my_var <- "Sally" # my_var is now of type character (aka string  )

# numeric - (10.5, 55, 787)
# integer - (1L, 55L, 100L, where the letter "L" declares this as an integer)
# complex - (9 + 3i, where "i" is the imaginary part)
# character (a.k.a. string) - ("k", "R is exciting", "FALSE", "11.5")
# logical (a.k.a. boolean) - (TRUE or FALSE)

x <- 10.5   # numeric
y <- 10L    # integer
z <- 1i     # complex
```

```
x <- 10.5
y <- 55
# Print the class name of x and y
```

```
class(x)
## [1] "numeric"
```

```
class(y)
## [1] "numeric"
```

```
# Type Conversion
as.numeric()
## numeric(0)
```

```
as.integer()
## integer(0)
```

```
as.complex()
## complex(0)
```

```
# how about is.numeric()?
```

4 If Else Statement

4.1 Booleans / Logical Values

```
10 > 9    # TRUE because 10 is greater than 9
## [1] TRUE
```

```
10 == 9   # FALSE because 10 is not equal to 9
## [1] FALSE
```

```
10 < 9    # FALSE because 10 is greater than 9
## [1] FALSE
```

```
a <- 10
b <- 9
a > b
## [1] TRUE
```

```
# Operator
# ==
# !=
# >
# <
# >=
# <=
```

4.2 If... Else

```
a <- 33
b <- 200

if (b > a) {
  print("b is greater than a")
}
## [1] "b is greater than a"
```

```r
if (b > a) {
  print ("b is greater than a")
} else {
  print("b is not greater than a")
}
## [1] "b is greater than a"
```

```r
a <- 33
b <- 33

if (b > a) {
  print("b is greater than a")
} else if (a == b) {
  print ("a and b are equal")
}
## [1] "a and b are equal"
```

```r
a <- 200
b <- 33

if (b > a) {
  print("b is greater than a")
} else if (a == b) {
  print("a and b are equal")
} else {
  print("a is greater than b")
}
## [1] "a is greater than b"
```

```r
a <- 200
b <- 33
c <- 500

if (a > b & c > a) {
  print("Both conditions are true")
}
## [1] "Both conditions are true"
```

```r
a <- 200
b <- 33
c <- 500

if (a > b | a > c) {
  print("At least one of the conditions is true")
}
## [1] "At least one of the conditions is true"
```

```r
x <- 41

if (x > 10) {
  print("Above ten")
  if (x > 20) {
    print("and also above 20!")
  } else {
    print("but not above 20.")
  }
} else {
  print("below 10.")
}
## [1] "Above ten"
## [1] "and also above 20!"
```

4.3 *In Class Applications*

Task 1

Variable `a` equals to the square root of 16654; and `b` equals to 122.

Use `if else` statement to write the following: if b > a, b is greater than a; otherwise, if a equals b, a and b are equal; otherwise else, a is greater than b.

Task 2

Variable `a` equals to the square root of 43435; and `b` equals to 210.

Use `if else` statement to write the following: if b > a or b == a, tell me that b is greater than or equal to a; otherwise, tell me a is greater than b.

5 For Loop and R Functions

5.1 Basic For Loop

```r
for (x in 1:10) {
  print(x)
}
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

```r

fruits <- list("apple", "banana", "cherry")

for (x in fruits) {
  print(x)
}
## [1] "apple"
## [1] "banana"
## [1] "cherry"
```

```r

dice <- c(1, 2, 3, 4, 5, 6)
for (x in dice) {
  print(x)
}
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
```

```
dice <- 1:6

for(x in dice) {
  if (x == 6) {
    print(paste("The dice number is", x, "Win!"))
  } else {
    print(paste("The dice number is", x, "Lose!"))
  }
}
## [1] "The dice number is 1 Lose!"
## [1] "The dice number is 2 Lose!"
## [1] "The dice number is 3 Lose!"
## [1] "The dice number is 4 Lose!"
## [1] "The dice number is 5 Lose!"
## [1] "The dice number is 6 Win!"
```

5.2 Nested Loops

```
adj <- list("red", "big", "tasty")
fruits <- list("apple", "banana", "cherry")
  for (x in adj) {
    for (y in fruits) {
      print(paste(x, y))
  }
}
## [1] "red apple"
## [1] "red banana"
## [1] "red cherry"
## [1] "big apple"
## [1] "big banana"
## [1] "big cherry"
## [1] "tasty apple"
## [1] "tasty banana"
## [1] "tasty cherry"
```

5.3 Functions

```
my_function <- function() { # create a function with the name my_function
  print("Hello World!")
}


my_function() # call the function named my_function
## [1] "Hello World!"
```

```r
my_function <- function(fname) {
  paste(fname, "Griffin")
}

my_function("Peter")
## [1] "Peter Griffin"
```

```r
my_function("Lois")
## [1] "Lois Griffin"
```

```r
my_function("Stewie")
## [1] "Stewie Griffin"
```

```r
my_function <- function(x) {
  return (5 * x)
}

print(my_function(3))
## [1] 15
```

```r
print(my_function(5))
## [1] 25
```

```r
print(my_function(9))
## [1] 45
```

```r
Nested_function <- function(x, y) {
  a <- x + y
  return(a)
}

Nested_function(Nested_function(2,2), Nested_function(3,3))
## [1] 10
```

5.4 *In Class Applications*

The "value" takes value from 60 to 65. For each value in "value", print the following:

- "The value is 60 not 65."
- "The value is 61 not 65."
- "The value is 62 not 65."
- "The value is 63 not 65."
- "The value is 64 not 65."
- "The value is 65, finally!"