IntermediateR

Saul Díaz Infante Velasco

2/8/23

Table of contents

Preface

Course Description

Intermediate R is the next stop on your journey in mastering the R programming language. In this R training, you will learn about conditional statements, loops, and functions to power your own R scripts. Next, make your R code more efficient and readable using the apply functions. Finally, the utilities chapter gets you up to speed with regular expressions in R, data structure manipulations, and times and dates. This course will allow you to take the next step in advancing your overall knowledge and capabilities while programming in R.

Introduction

Grade Rubric

The course encloenses 81 excercise and 14 videos that results in 6950 xp.

name	XP
Equality	100
Greater and less than	100
Compare Vectors	100
Compare Matrices	100
& and |	100
& and | (2)	100
Blend it all together	100
The is statement	100
Add and else	100
Customize furhter: else if	100
Take Control	100

1 Conditionals and Control Flow

In this chapter, you'll learn about relational operators for comparing R objects, and logical operators like "and" and "or" for combining TRUE and FALSE values. Then, you'll use this knowledge to build conditional statements.

1.1 Equality

The most basic form of comparison is equality. Let's briefly recap its syntax. The following statements all evaluate to `TRUE` (feel free to try them out in the console).

```
3 == (2 + 1)
"intermediate" != "r"
TRUE != FALSE
"Rchitect" != "rchitect"
```

Notice from the last expression that R is case sensitive: "R" is not equal to "r". Keep this in mind when solving the exercises in this chapter!

Instructions `100 XP`

- In the editor on the right, write R code to see if `TRUE` equals `FALSE`.
- Likewise, check if `-6 * 14` is not equal to `17 - 101`.
- Next up: comparison of character strings. Ask R whether the strings "useR" and "user" are equal.
- Finally, find out what happens if you compare logicals to numerics: are `TRUE` and 1 equal?

ex_001.R

```
# Comparison of logicals
TRUE == FALSE

# Comparison of numerics
```

```r
-6 * 14 != 17 -101
# Comparison of character strings
"useR" == "user"

# Compare a logical with a numeric
TRUE == 1
```

1.2 Greater and less than

Apart from equality operators, Filip also introduced the less than and greater than operators: $<$ and $>$. You can also add an equal sign to express less than or equal to or greater than or equal to, respectively. Have a look at the following R expressions, that all evaluate to FALSE:

```r
(1 + 2) > 4
"dog" < "Cats"
TRUE <= FALSE
```

Remember that for string comparison, R determines the greater than relationship based on alphabetical order. Also, keep in mind that `TRUE` is treated as 1 for arithmetic, and `FALSE` is treated as 0. Therefore, `FALSE < TRUE` is `TRUE`.

Instructions `100 XP`

- Write R expressions to check whether:
- `-6 * 5 + 2` is greater than or equal to `-10 + 1`.
- "raining" is less than or equal to "raining dogs".
- `TRUE` is greater than `FALSE`.

ex_002.R

```r
# Comparison of numerics

-6 * 5 + 2 >= -10 +1

# Comparison of character strings
"raining" <=  "raining dogs"
```

```
# Comparison of logicals

TRUE > FALSE
```

1.3 Compare vectors

You are already aware that R is very good with vectors. Without having to change anything about the syntax, R's relational operators also work on vectors.

Let's go back to the example that was started in the video. You want to figure out whether your activity on social media platforms have paid off and decide to look at your results for LinkedIn and Facebook. The sample code in the editor initializes the vectors linkedin and facebook. Each of the vectors contains the number of profile views your LinkedIn and Facebook profiles had over the last seven days.

1.3.1 Instructions `100 XP`

Using relational operators, find a logical answer, i.e. `TRUE` or `FALSE`, for the following questions:

- On which days did the number of LinkedIn profile views exceed 15? When was your
- LinkedIn profile viewed only 5 times or fewer? When was your LinkedIn profile
- visited more often than your Facebook profile?

ex003.R

```
# The linkedin and facebook vectors have already been created for you
# The linkedin and facebook vectors have already been created for you
linkedin <- c(16, 9, 13, 5, 2, 17, 14)
facebook <- c(17, 7, 5, 16, 8, 13, 14)

# Popular days

linkedin > 15
# Quiet days
linkedin <= 5

# LinkedIn more popular than Facebook
linkedin > facebook
```

1.4 Compare matrices

R's ability to deal with different data structures for comparisons does not stop at vectors. Matrices and relational operators also work together seamlessly!

Instead of in vectors (as in the previous exercise), the LinkedIn and Facebook data is now stored in a matrix called `views`. The first row contains the LinkedIn information; the second row the Facebook information. The original vectors `facebook` and `linkedin` are still available as well.

1.4.1 Instructions `100 XP`

Using the relational operators you've learned so far, try to discover the following:

- When were the views exactly equal to 13? Use the views matrix to return a logical matrix.
- For which days were the number of views less than or equal to 14? Again, have R return a logical matrix.

ex_004.R

```
# The social data has been created for you
linkedin <- c(16, 9, 13, 5, 2, 17, 14)
facebook <- c(17, 7, 5, 16, 8, 13, 14)
views <- matrix(c(linkedin, facebook), nrow = 2, byrow = TRUE)

# When does views equal 13?

views == 13
# When is views less than or equal to 14?

views <= 14
```

1.5 & and |

Before you work your way through the next exercises, have a look at the following R expressions. All of them will evaluate to TRUE:

```
TRUE & TRUE
FALSE | TRUE
```

```
5 <= 5 & 2 < 3
3 < 4 | 7 < 6
```

Watch out: `3 < x < 7` to check if `x` is between 3 and 7 will not work; you'll need `3 < x & x < 7` for that.

In this exercise, you'll be working with the `last` variable. This variable equals the value of the `linkedin` vector that you've worked with previously. The `linkedin` vector represents the number of LinkedIn views your profile had in the last seven days, remember? Both the variables `linkedin` and last have been pre-defined for you.

Instructions 100 XP

Write R expressions to solve the following questions concerning the variable last:

- Is `last` under 5 or above 10?
- Is `last` between 15 and 20, excluding 15 but including 20?

ex_005.R

```
# The linkedin and last variable are already defined for you
linkedin <- c(16, 9, 13, 5, 2, 17, 14)
last <- tail(linkedin, 1)

# Is last under 5 or above 10?
last < 5 | last > 10

# Is last between 15 (exclusive) and 20 (inclusive)?
last > 15 | last < 20
```

1.6 & and | (2)

Like relational operators, logical operators work perfectly fine with vectors and matrices.

Both the vectors `linkedin` and `facebook` are available again. Also a matrix `-views-` has been defined; its first and second row correspond to the `linkedin` and `facebook` vectors, respectively. Ready for some advanced queries to gain more insights into your social outreach?

Instructions 100 XP

- When did LinkedIn views exceed 10 and did Facebook views fail to reach 10 for a particular day? Use the `linkedin` and `facebook` vectors.
- When were one or both of your LinkedIn and Facebook profiles visited at least 12 times?
- When is the `views` matrix equal to a number between 11 and 14, excluding 11 and including 14?

ex_006.R

```
# The social data (linkedin, facebook, views) has been created for you

# linkedin exceeds 10 but facebook below 10
linkedin > 10 & facebook < 10


# When were one or both visited at least 12 times?
linkedin >= 12 | facebook >= 12

# When is views between 11 (exclusive) and 14 (inclusive)?

views > 11 & views <= 14
```

1.7 Blend it all together

With the things you've learned by now, you're able to solve pretty cool problems.

Instead of recording the number of views for your own LinkedIn profile, suppose you conducted a survey inside the company you're working for. You've asked every employee with a LinkedIn profile how many visits their profile has had over the past seven days. You stored the results in a data frame called `li_df`. This data frame is available in the workspace; type `li_df` in the console to check it out.

Instructions 100 XP

- Select the entire second column, named `day2`, from the `li_df` data frame as a vector and assign it to `second`.

- Use `second` to create a logical vector, thatcontains `TRUE` if the corresponding number of views is strictly greater than 25 or strictly lower than 5 and FALSE otherwise. Store this logical vector as `extremes`.

- Use `sum()` on the `extremes` vector to calculate the number of `TRUE`s in `extremes` (i.e. to calculate the number of employees that are either very popular or very low-profile). Simply print this number to the console.

ex_007.R

```
# li_df is pre-loaded in your workspace

# Select the second column, named day2, from li_df: second
second <- li_df$day2

# Build a logical vector, TRUE if value in second is extreme: extremes
extremes <- second > 25 | second < 5

# Count the number of TRUEs in extremes
print(sum(extremes))
```

1.8 The if statement

Before diving into some exercises on the if statement, have another look at its syntax:

```
if (condition) {
  expr
}
```

Remember your vectors with social profile views? Let's look at it from another angle. The `medium` variable gives information about the social website; the `num_views` variable denotes the actual number of views that particular `medium` had on the last day of your recordings. Both variables have been pre- defined for you.

Instructions `100 XP`

- Examine the `if` statement that prints out "Showing LinkedIn information" `if` the `medium` variable equals `"LinkedIn"`. -Code an `if` statement that prints "You are popular!" to the console if the `num_views` variable exceeds 15.

ex_008.R

```
# Variables related to your last day of recordings
medium <- "LinkedIn"
num_views <- 14

# Examine the if statement for medium
if (medium == "LinkedIn") {
  print("Showing LinkedIn information")
}

# Write the if statement for num_views

if (num_views > 15) {
  print("You are popular!")
}
```

1.9 Add an else

You can only use an else statement in combination with an if statement. The else statement does not require a condition; its corresponding code is simply run if all of the preceding conditions in the control structure are FALSE. Here's a recipe for its usage:

```
if (condition) {
  expr1
} else {
  expr2
}
```

It's important that the else keyword comes on the same line as the closing bracket of the if part!

Both if statements that you coded in the previous exercises are already available to use. It's now up to you to extend them with the appropriate else statements!

Instructions `100 XP`

Add an else statement to both control structures, such that

- "Unknown medium" gets printed out to the console when the if-condition on medium does not hold.
  -R prints out "Try to be more visible!" when the if-condition on num_views is not met.

ex_009.R

```
# Variables related to your last day of recordings
medium <- "LinkedIn"
num_views <- 14
# Control structure for medium
if (medium == "LinkedIn") {
  print("Showing LinkedIn information")
} else{
  print("Unknown medium" )
}

# Control structure for num_views
if (num_views > 15) {
  print("You're popular!")
} else{
  print( "Try to be more visible!" )
}
```

1.10 Customize further: else if

The `else if` statement allows you to further customize your control structure. You can add as many `else if` statements as you like. Keep in mind that R ignores the remainder of the control structure once a condition has been found that is TRUE and the corresponding expressions have been executed. Here's an overview of the syntax to freshen your memory:

```
if (condition1) {
  expr1
} else if (condition2) {
  expr2
} else if (condition3) {
  expr3
} else {
  expr4
}
```

Again, It's important that the else if keywords comes on the same line as the closing bracket of the previous part of the control construct!

Instructions `100 XP`

Add code to both control structures such that:

- R prints out "Showing Facebook information" if `medium` is equal to "Facebook". Remember that R is case sensitive!
- "Your number of views is average" is printed if `num_views` is between 15 (inclusive) and 10 (exclusive). Feel free to change the variables `medium` and `num_views` to see how the control structure respond. In both cases, the existing code should be extended in the `else if` statement. No existing code should be modified.

ex_010.R

```
# Variables related to your last day of recordings
medium <- "LinkedIn"
num_views <- 14

# Control structure for medium
if (medium == "LinkedIn") {
  print("Showing LinkedIn information")
} else if (medium == "Facebook") {
  # Add code to print correct string when condition is TRUE
  print("Showing Facebook information" )
} else {
  print("Unknown medium")
}

# Control structure for num_views
if (num_views > 15) {
  print("You're popular!")
} else if (num_views <= 15 & num_views > 10) {
  # Add code to print correct string when condition is TRUE
  print("Your number of views is average")
} else {
  print("Try to be more visible!")
}
```

1.11 Else if 2.0

You can do anything you want inside if-else constructs. You can even put in another set of conditional statements. Examine the following code chunk:

```
if (number < 10) {
  if (number < 5) {
    result <- "extra small"
  } else {
    result <- "small"
  }
} else if (number < 100) {
  result <- "medium"
} else {
  result <- "large"
}
print(result)
```

Have a look at the following statements:

(1) If number is set to 6, "small" gets printed to the console.
(2) If number is set to 100, R prints out "medium".
(3) If number is set to 4, "extra small" gets printed out to the console.
(4) If number is set to 2500, R will generate an error, as result will not be defined.

Select the option that lists all the true statements.

> Run the code or a handwrite test (1, 3).

1.12 Take control!

In this exercise, you will combine everything that you've learned so far: relational operators, logical operators and control constructs. You'll need it all!

We've pre-defined two values for you: `li` and `fb`, denoting the number of profile views your LinkedIn and Facebook profile had on the last day of recordings. Go through the instructions to create R code that generates a 'social media score', sms, based on the values of `li` and `fb`.

Instructions 100 XP

Finish the control-flow construct with the following behavior:

- If both `li` and `fb` are 15 or higher, set `sms` equal to double the sum of `li` and `fb`.
- If both `li` and `fb` are strictly below 10, set `sms` equal to half the `sum` of `li` and `fb`.
- In all other cases, set sms equal to `li + fb`.
- Finally, print the resulting `sms` variable.

ex_011.R

```
# Variables related to your last day of recordings
li <- 15
fb <- 9

# Code the control-flow construct
if (li >= 15 & fb >= 15) {
  sms <- 2 * (li + fb)
} else if (li < 10 & fb < 10) {
  sms <- 0.5 * (li + fb)
} else {
  sms <- li + fb
}

# Print the resulting sms to the console
print(sms)
```

2 Summary

In summary, this book has no content whatsoever.

References