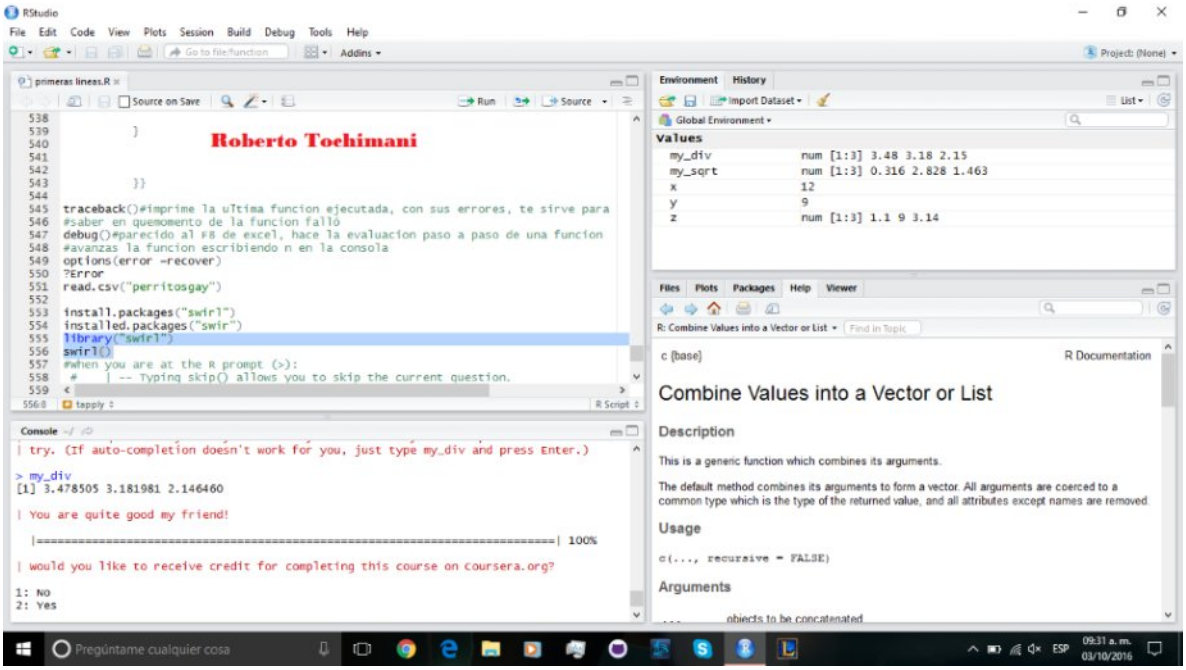
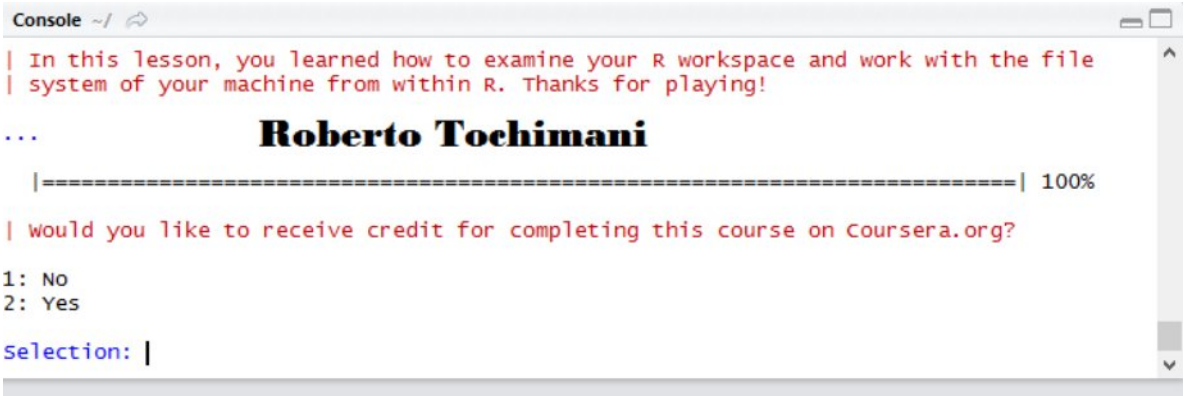
# 1: Basic Building Blocks




# 2: Workspace and Files



## 3: Sequences of Numbers

```
Console ~/

| If instead we want our vector to contain 10 repetitions of the vector (0, 1, 2),
| we can do rep(c(0, 1, 2), times = 10). Go ahead.

> rep(c(0, 1, 2), times = 10)
 [1] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2

| Great job!
```

**Roberto tochimani**

```
  |=================================================================================|  96%

| Finally, let's say that rather than repeating the vector (0, 1, 2) over and over
| again, we want our vector to contain 10 zeros, then 10 ones, then 10 twos. We can
| do this with the `each` argument. Try rep(c(0, 1, 2), each = 10).

> rep(c(0, 1, 2), each = 10)
 [1] 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2

| All that practice is paying off!

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

## 4: Vectors

```
Console ~/

| recycles, or repeats, 1:4 until it matches the length of LETTERS.

...

  |=================================================================================|  95%

| Also worth noting is that the numeric vector 1:4 gets 'coerced' into a character
| vector by the paste() function.

...
```

**Tochimani**

```
  |=================================================================================|  97%

| We'll discuss coercion in another lesson, but all it really means is that the
| numbers 1, 2, 3, and 4 in the output above are no longer numbers to R, but rather
| characters "1", "2", "3", and "4".

...

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

## 5: Missing Values

```
Console ~/

  |=================================================================      |  95%

| Let's do one more, just for fun. In R, Inf stands for infinity. What happens if
| you subtract Inf from Inf?

> Inf(Inf)
Error: attempt to apply non-function
> Inf[Inf]
[1] NA

| Almost! Try again. Or, type info() for more options.

| Type Inf - Inf. Can you guess the result?

> Inf-Inf
[1] NaN

| Excellent job!

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: No
2: Yes

Selection: |
```

**Tochimani**

## 6: Subsetting Vectors

```
Console ~/

  |=================================================================      |  95%

| Likewise, we can specify a vector of names with vect[c("foo", "bar")]. Try it out.

> vect[c("foo","bar")]
foo bar
 11   2

| You are doing so well!

  |=================================================================================  |  97%

| Now you know all four methods of subsetting data from vectors. Different
| approaches are best in different scenarios and when in doubt, try it out!

...

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection: |
```

**Roberto tochimani**

## 7: Matrices and Data Frames

```
Console ~/
  patient age weight bp rating test
1    Bill   1      5  9     13   17
2    Gina   2      6 10     14   18
3   Kelly   3      7 11     15   19
4    Sean   4      8 12     16   20

| All that practice is paying off!

  |=================================================================================
  |  97%

| In this lesson, you learned the basics of working with two very important and common
data
| structures -- matrices and data frames. There's much more to learn and we'll be cover
ing more
| advanced topics, particularly with respect to data frames, in future lessons.

...

  |=================================================================================
==| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: No
2: Yes

Selection:
```

**Roberto Tochimani**

## 8: Logic

```
Console ~/
  |=================================================================================  |  96%

| Which of the following evaluates to TRUE?

1: all(c(TRUE, FALSE, TRUE))
2: all(ints == 10)
3: any(ints == 2.5)
4: any(ints == 10)

Selection: 4

| Keep working like that and you'll get there!

  |=================================================================================  |  98%

| That's all for this introduction to logic in R. If you really want to see what you can do
| with logic, check out the control flow lesson!

...

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection: |
```

**Roberto Tochimani**

## 9: Functions

```
Console ~/

| You are really on a roll!

  |=================================================================================  |  98%

| We've come to the end of our lesson! Go out there and write some great functions!

...

  |==================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: No
2: Yes

Selection: |
```

**Roberto Tochimani**

## 10: lapply and sapply

```
Console ~/

  |=================================================================================  |  96%

| The only difference between previous examples and this one is that we are defining and using our own
| function right in the call to lapply(). Our function has no name and disappears as soon as lapply()
| is done using it. So-called 'anonymous functions' can be very useful when one of R's built-in
| functions isn't an option.

...

  |=================================================================================  |  98%

| In this lesson, you learned how to use the powerful lapply() and sapply() functions to apply an
| operation over the elements of a list. In the next lesson, we'll take a look at some close relatives
| of lapply() and sapply().

...

  |==================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

**Roberto Tochimani**

## 11: vapply and tapply

```
Console ~/

2: 1010.0
3: 119.0
4: 56.00
5: 5.00

Selection: 4

| Perseverance, that's the answer.

  |=================================================================================  |  96%

| In this lesson, you learned how to use vapply() as a safer alternative to sapply(), which is most
| helpful when writing your own functions. You also learned how to use tapply() to split your data
| into groups based on the value of some variable, then apply a function to each group. These
| functions will come in handy on your quest to become a better data analyst.

...

  |==================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

**Roberto Tochimani**

## 12: Looking at Data

```
Console ~/
...

  |=================================================================================|  92%

| str() is actually a very general function that you can use on most objects in R. Any time you want
| to understand the structure of something (a dataset, function, etc.), str() is a good place to
| start.
```

**Roberto Tochimani**

```
...

  |=================================================================================|  96%

| In this lesson, you learned how to get a feel for the structure and contents of a new dataset using
| a collection of simple and useful functions. Taking the time to do this upfront can save you time
| and frustration later on in your analysis.

...

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: No
2: Yes

Selection: |
```

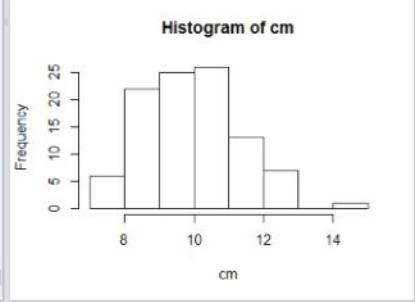
## 13: Simulation

```
Console ~/
| at work, but that's a lesson for another day!
```

**Roberto Tochimani**

```
....

  |=================================================================================|  94%

| All of the standard probability distributions are built into R, including exponential (rexp()),
| chi-squared (rchisq()), gamma (rgamma()), .... Well, you see the pattern.

...

  |=================================================================================|  97%

| Simulation is practically a field of its own and we've only skimmed the surface of what's possible.
| I encourage you to explore these and other functions further on your own.

...

  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: No
2: Yes

Selection:
```

```
cm        num [1:100] 10.2 9.6 8.8 9.4 8 8.4 8...
Files  Plots  Packages  Help  Viewer
Zoom  Export  Publish
```




## 14: Dates and Times

```
Console ~/
  |=================================================================================|  94%

| Use difftime(Sys.time(), t1, units = 'days') to find the amount of time in DAYS that has passed
| since you created t1.

> difftime(Sys.time(), t1, units = 'days')
Time difference of 0.006035344 days

| Keep working like that and you'll get there!

  |=================================================================================|  97%

| In this lesson, you learned how to work with dates and times in R. While it is important to
| understand the basics, if you find yourself working with dates and times often, you may want to
| check out the lubridate package by Hadley Wickham.

...
```

**Roberto Tochimani**

```
  |=================================================================================| 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection: |
```

# 15: Base Graphics