# Pull the Right Strings with stringr: Exercises

By providing a set of wrappers to existing functions, the stringr package allows for simple, consistent and efficient manipulations of strings in R. Even though there are some more basic packages that offer strings-related functions, you might find yourself in need for a more complete and straightforward solution for handling strings in R.

With a simple and consistent syntax, stringr provides some very convenient functions around pattern matching, characters manipulation, whitespace handling and more. The full reference of the package can be found here.

Please find below a set of exercises that will help you practice a variety of stringr functions. The focus is on practical operations that data analysts are required to perform on a daily basis. Answers to the exercises are available here. And, don't forget to check out our other exercise sets on the stringr package by following the stringr tag.

For the following exercises we will use this data:

addresses <- c("14 Pine Street, Los Angeles", "152 Redwood Street, Seattle", "8 Washington Boulevard, New York")

products <- c("TV ", " laptop", "portable charger", "Wireless Keybord", " HeadPhones ")

long_sentences <- stringr::sentences[1:10]

```
field_names <- c("order_number", "order_date", "customer_email", "product_title", "amount")

employee_skills <- c("John Bale (Beginner)", "Rita Murphy (Pro)", "Chris White (Pro)", "Sarah Reid (Medium)")
```

**Exercise 1**
Normalize the addresses vector by replacing capitalized letters with lower-case ones.

**Exercise 2**
Pull only the numeric part of the addresses vector.

**Exercise 3**
Split the addresses vector into two parts: address and city. The result should be a matrix.

**Exercise 4**
Now try to split the addresses vector into three parts: house number, street and city. The result should be a matrix.
Hint: use a regex lookbehind assertion

**Exercise 5**
In the long_sentences vector, for sentences that start with the letter "T" or end with the letter "s", show the first or last word respectively. If the sentence both starts with a "T" and ends with an "s", show both the first and the last words. Remember that the actual last character of a sentence is usually a period.

**Learn more** about string manipulation with stringr in the online course Learn R by Intensive Practice.

**Exercise 6**
Show only the first 20 characters of all sentences in the long_sentences vector. To indicate that you removed some characters, use two consecutive periods at the end of each sentence.

**Exercise 7**
Normalize the products vector by removing all unnecessary whitespaces (both from the start, the end and the middle), and by capitalizing all letters.

**Exercise 8**
Prepare the field_names for display, by replacing all of the underscore symbols with spaces, and by converting it to the title-case.

**Exercise 9**
Align all of the field_names to be with equal length, by adding whitespaces to the beginning of the relevant strings.

**Exercise 10**
In the employee_skills vector, look for employees that are defined as “Pro” or “Medium”. Your output should be a matrix that have the employee name in the first column, and the skill level (without parenthesis) in the second column. Employees that are not qualified should get missing values in both columns.

---

# Stringr Basic Functions: Exercises

The more ubiquitous data becomes, the number of standards and ways the data can get to you in a messy state both increase. I’ve found that many projects I’ve worked on, to my surprise, turned out to need a substantial amount of text processing

skills.

Base R has some powerful tools to manipulate strings, but specialized packages are also gaining momentum. stringr's simple and consistent syntax, makes it a strong alternative to base R functions. It doesn't hurt that it is written by R rockstar Hadley Wickham, who is also the author of other packages in wide use, such as dplyr, ggplot2.

Solutions are available here.

**Exercise 1**
Load (and install) the stringr and gapminder package. For a warm up, make a new data.frame based on the gapminder data with one row per country and two columns. Name the country and the continent it is classified to.  Name it simply df.

**Exercise 2**
Use a stringr function to find out what the average length of the country names are, as they appear in the data-set.

**Exercise 3**
Extract the first and last letter of each country's name. Make a frequency plot for both. Here you can use base-R function table.

**Exercise 4**
What countries have the word "and" as part of their name?

**Exercise 5**
Delete all instances of "," and "." from the country names.


**Exercise 6**

Use str_dup and str_c to generate the vector c("mouse likes cat very much", "mouse likes cat very very much", "mouse likes cat very very very much").

**Exercise 7**
Imagine you are creating an app to explore the Gapminder data; the tool you are using can only accommodate country names of 12 characters. Therefore, you decide to shorten the names from the right, such that if the country name is longer than 12 characters, you trim it to 11 and add a full stop. Example: “United States” becomes “United Stat.”. Use str_trunc(), then find a way to reach the same result without it.

**Exercise 8**
sentences is a character vector of 720 sentences that loads to your environment when you load the stringr package. Extract all two-character words from it and plot their frequency.

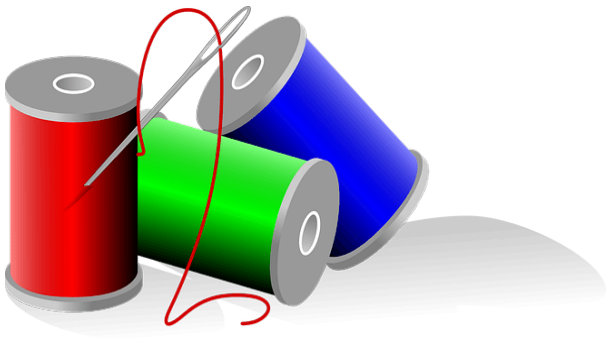
**Exercise 9**
Convert the names to lower case and count what characters are the most common in the country names overall.

**Exercise 10**
Only one country has “x” in its name, congrats Mexico! “A” is the most used character. What is the country that takes this the furthest and has the most “a”s in its name?

(Image by Jan)

---

# Hacking strings with stringr



This is first of the set of exercise on string manipulation with stringr

Answers to the exercises are available here.

If you obtained a different (correct) answer than those listed on the solutions page, please feel free to post your answer as a comment on that page.

**Exercise 1**

use a stringr function to merge this 3 strings .

```
x <- "I AM SAM. I AM SAM. SAM I AM"
y <- "THAT SAM-I-AM! THAT SAM-I-AM! I DO NOT LIKE THAT SAM-I-AM!"
z <- ""DO WOULD YOU LIKE GREEN EGGS AND HAM?"
```

**Exercise 2**

Now use a vector which contains x,y,z and NA and make it a single sentence using paste ,do the same by the same function you used for exercise1 .Can you spot the difference .

**Exercise 3**

Install the babynames dataset ,find the vector of length of the babynames using stringr functions. You may wonder nchar can do the same so why not use that ,try finding out the difference and let me know in the comments.

**Exercise 4**

We often use substr to get part of the string ,in stringr world there exist a much powerful function which does almost

the same thing . Create a string name with your name .
Use str_sub to get the last character and the last 5 characters .

**Exercise 5**

In mtcars dataset rownames, find all cars of the brand Merc .

**Learn more** about Text analysis in the online course Text Analytics/Text Mining Using R. In this course you will learn how create, analyse and finally visualize your text based data source. Having all the steps easily outlined will be a great reference source for future work.

**Exercise 6**

Use the same mtcars rownames ,find the total number of times “e” appears in that .

**Exercise 7**

Suppose you have a string like this
j <- "The_quick_brown_fox_jumps_over_the_lazy_dog"
split it in words using a stringr function

**Exercise 8**

On the same string I need the first word splitted but the rest intact ,help me to achieve that

**Exercise 9**

Now for on the same string J
a> I want the first “_” replaced by “–”
b> I want all the “_” replaced by “–”

**Exercise 10**

Many of the times ,you don’t want NA to appear when you do some string manipulation but its sometimes necessary to replace NA as a character(rather than remove it) ,stringr

provides a useful tool for that.
Now if I have a vector like this ,
na_string_vec <-
c(“The_quick_brown_fox_jumps_over_the_lazy_dog”,NA)

How can you turn the NA into a character string .