Challenges

Rohit Rawat

January 30, 2016

Challenge 2.1 Goal : to identify the data manipulation actions needed to perform a task

Look at this data:

country	year	pop	continent	lifeExp	gdpPercap
Afghanistan	1952	8425333	Asia	28.801	779.4453
Afghanistan	1957	9240934	Asia	30.332	820.8530
Afghanistan	2007	31889923	Asia	43.828	974.5803
Albania	1952	1282697	Europe	55.230	1601.0561
Albania	1967	1984060	Europe	66.220	2760.1969
Albania	1972	2263554	Europe	67.690	3313.4222
Albania	1977	2509048	Europe	68.930	3533.0039
Argentina	1952	17876956	Americas	62.485	5911.3151
Argentina	1957	19610538	Americas	64.399	6856.8562

All of this data is loaded into a `data.frame` called `gapminder`. Your boss wants you to compute the average life expectancy over all the years of data available for *Albania*. **Think of what other variables you will need to create to store intermediary results and the final result. Also think about the type each variable will have to be (data.frame, vector, scalar, string?)**. There are many correct answers to this problem. (no code is needed)

Solution You will want to maybe create a subset `data.frame` with only the rows for Algeria. You would then want to cut out life expectancy column into a `vector`. You would then compute the mean and store it in a `scalar`.

Challenge 2.2 Goal : write a basic script

Write and run your own script that will

- Set `x` to 25
- Set `y` to 15
- Calculate the sum and store it in a variable `z`
- print `z`

To start a new script, open the File menu -> New File -> R script. Save it as **`myAdder.R`** and run it.

Solution

```
# a simple script to add two numbers

# create two variables x and y
x <- 25
y <- 15

# calculate the sum z
z <- x + y

# print the result
print(paste('The sum of 25 and 15 is', z))
```

```
## [1] "The sum of 25 and 15 is 40"
```

Challenge 2.3 Goal : Wrapped function calls.

As we've seen in our print statements, we can use `paste` or `paste0` to concatenate strings.

1. Write a function called `fence` that takes two parameters called `original` and `wrapper` and returns a new string that has the `wrapper` character at the beginning and end of the `original`:

Example function call and output:

```
fence('name', '---')
---name---
```

Solution

```
# function to surround a piece of text with other text
fence <- function(original, wrapper) {
  result <- paste0(wrapper, original, wrapper)

  return(result)
}

# test function call
print(fence('Rohit', '***'))
```

```
## [1] "***Rohit***"
```