Lab 1

David Murillo

09/11/2022

Run the following two lines of code in the console.

```
c(1, 2, 3)
```

```
## [1] 1 2 3
```

```
"c(1, 2, 3)"
```

```
## [1] "c(1, 2, 3)"
```

Q1 (2 pts.): Explain why the outputs of the two lines are different.

Answer: the first line is making a numeric vector, and the second line is making a character vector.

Run the following two lines of code in the console and consider the differences:

```
c_1 = c(1, 2, 3)
c_2 = "c(1, 2, 3)"
```

Q2 (1 pt.): Is c_1 a variable, or a function? How do you know?

Answer: c_1 is a variable numeric with three observations, I know because we are making a vector numeric with la function c().

Q3 (1 pt.): Is c_2 a variable, or a function? How do you know?

Answer: c_2 is a variable only with one observation, I know because all la character is between ” “.

Q4 (1 pt.): If c_1 and c_2 have different values, why?

Answer: Yes, c_1 is a numeric vector with three observations, and c_2 is a character with one observation.

Create a numeric vector of length 6 called my_vec. It should contain the integers from 1 to 6. Build a matrix using the following code:

```
my_vec <- c(1:6)
mat_1 = matrix(my_vec, nrow = 3)
```

Q5 (1 pt.): What are the dimensions of the matrix (i.e. how many rows and columns)?

Answer: The matrix contain three rows and two colums.

Q6 (2 pts.): Write R code to retrieve the element of mat_1 that has a value of 3.

Answer: mat_1[3,1]

You will use my_vec from the previous question again.

Create a matrix mat_2 that has two rows and three columns using my_vec. Do not use the c() or rep() functions.

Create a matrix mat_3 that has three rows and two columns using my_vec. Do not use the c() or rep() functions.

```
mat_2 <- matrix(my_vec, nrow = 2, ncol = 3)
mat_3 <- matrix(my_vec, nrow = 3, ncol = 2)
```

Q7 (1 pt.): Paste the code you used to create mat_2.

mat_2 <- matrix(my_vec, nrow = 2, ncol = 3)

Q8 (1 pt.): Paste the code you used to create mat_3.

mat_3 <- matrix(my_vec, nrow = 3, ncol = 2)

Q9 (1 pt.): Did R use rows or columns to recycle/distribute the values in my_vec? Answer: Yes

Q10 (1 pt.): Using my_vec, create a matrix, mat_4. mat_4 must have a total number of elements that is not a multiple of 3.

```
mat_4 <- matrix(my_vec[my_vec == 2 | my_vec == 4 | my_vec == 5])
matrix(my_vec[!my_vec  %% 3 == 0])
```

```
##      [,1]
## [1,]    1
## [2,]    2
## [3,]    4
## [4,]    5
```

```
mat_4
```

```
##      [,1]
## [1,]    2
## [2,]    4
## [3,]    5
```

Q11 (1 pt.): How did R handle the recycling/distributing of values of my_vec in mat_4?

Answer: I tried different options to filter the total number of elements that is not a multiple of 3, with the ! and %% R give the number 1,2,4 and 5. R recycle the value of the my_vec and put in a matrix of three rows and one colum with my other option matrix(my_vec[my_vec == 2 | my_vec == 4 | my_vec == 5]).

Create a list, named my_list_1 with following three elements:

first element is numeric: 5.2 second element is a string “five point two” third element is a vector of all integers from 0 to 5. Do you recall how to do this from the DataCamp course?

```
my_list_1 <- list(5.2, "five point two", c(0:5) )
```

Name the elements in my_list_1:

“two” “one” “three” NOTE: Yes they are out of order!!!

```
names(my_list_1) <- c("two", "one", "three")
```

Run the following 8 lines of code.

```
my_list_1[[1]]
```

```
## [1] 5.2
```

```
my_list_1[[as.numeric("1")]]
```

```
## [1] 5.2
```

```
my_list_1[["1"]]
```

```
## NULL
```

```
my_list_1[["one"]]
```

```
## [1] "five point two"
```

```
my_list_1$one
```

```
## [1] "five point two"
```

```
my_list_1$"one"
```

```
## [1] "five point two"
```

```
#my_list_1$1
#error
my_list_1$"1"
```

```
## NULL
```

Q12 (8 pts.): For each of the 8 lines, answer the following: A. Did the line return a 1: value, 2: error, or 3: NULL? B. What type of subsetting operation was used (or attempted)? C. If it did not return an error describe, in ordinary English, a plausible explanation of how R could have performed the subsetting.

Answer: first line: value, filter argument of list in the first position, R take the argument of the list on the first position.

second line: value, filter argument called "1" as numeric, R take the argument called "1" as numeric

third line: NULL, argument as string called 1, R is filter the argument called "1" as string

fourth line: value, the argument called "one", R is filter the argument called one as string

fifth line: value, the argument called one, R take the argument called one

sixth line: value, the argument called one, R take the argument called "one" as string

seventh line: error, the argument called 1 as numeric, R take the argument called 1 as numeric

eighth line NULL, the argument called 1 as string, R take the argument called "1" as string.

Q13 (2 pts.): Identify which lines produced the string output "five point two" and explain why.

Answer: In the lines 4, 5, and 6. The result is because in the line 4, we are filter the argument called one in our list. In the line 5, also we are selecting the argument called one in our list. And in the line 6, we are calling the argument called one, but this time as a string. Because all the criteria how we called the argument one are correct, we got a result.

Q14 (1 pt.): Identify which lines produced NULL output and explain why.

Answer: In the lines 3, and 8. In the line 3 we are filter the argument called 1 in our list, we do not have argument called 1 because that we got a NULL result, in the line 8, we are called the argument called 1 as string, we do not have a argument called 1, because that we got a NULL.