ICO 187 ANÁLISIS DE DATOS

Clase 25: Programación en R usando RStudio

Año: 2021
Profesor: Sebastián Egaña

1. Clase anterior

En la clase anterior, introdujimos aspectos relacionados con la utilización de R, Rstudio y la programación asociada. Ahora nos abocaremos a intentar programar cosas simples.

Aclaración: Las líneas de este documento que empiezan con ## [1] corresponde al output de la consola de R, y se generan debido a que se corre un código dentro del documento.

2. Programación básica en R

2.1. Hello World!

Un ejercicio básico es poder darle la instrucción al computador de imprimir (print) la frase "Hello World!" (Hola Mundo!). Veamos como hacer esto en R.

El siguiente código puede realizar la tarea en R:

```
print("Hello world!")
```

```
## [1] "Hello world!"
```

Veamos una estructura distinta, generando un objeto llamado print y después imprimiendolo en la consola:

```
print <- print("Hello world!")
```

```
## [1] "Hello world!"
```

```
print
```

```
## [1] "Hello world!"
```

2.2. ¿Cómo programar un dado?[1]

Pensemos en que elementos necesitamos para replicar un dado en la realidad.

1. Necesitamos la existencia de algun elemento que contenga las seis caras del dado.
2. Necesitamos alguna función que nos permita obtener una de las caras del dado.
3. Debemos procupar que el número obtenido cumpla con ser obtenido a través de muestreo con reemplazo.

Con esto podríamos tener un dado.

Para poder cumplir con el punto 1, podemos generar un elemento que contenga números desde el 1 al 6.

```
caras <- 1:6

caras
```

```
## [1] 1 2 3 4 5 6
```

Para el punto 2, debemos buscar un comando que pueda extraer una muestra del elemento que posee los números (que corresponden a las caras del dado). Esto lo puede realizar el comando sample.

Recordar que se puede buscar información sobre el comando anteponiendo signo de interrogación.

```
?sample
```

Ahora lo utilizamos, seteando la semilla para todos obtener el mismo resultado. Setear semilla, refiere a establecer el punto desde donde nace la generación aleotaria de números; al dejar establecida la "semilla", nos aseguramos de que a todos nos genere el mismo valor en un primer intento.

```
set.seed(10)

sample(x = caras, size = 1, replace = TRUE)
```

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

Podríamos ahora nombrar al objeto, para que quede en el ambiente de R.

```
dice <- sample(x = caras, size = 1, replace = TRUE)
```

Fijarse que al nombrarlo, no se genera un número aleatorio, lo único que se genera es el elemento.

Para "tirar el dado", deberíamos llamar al elemento ahora

```
set.seed(10)
dice
```

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

3. Vectores, matrices, lista y dataframes/tibbles

Pasaremos ahora a utilizar elementos u objetos de R, que por lo general poseen más de un dato.

[1] Ejemplo en base al libro Hands on Programming

3.1. Pregunta 1[2]

A. Genere un vector que contenga la siguiente secuencia de números de 1 al 10.

Existen tres formas para poder realizar esto

- Forma 1.

```
seq(1,10,1)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

- Forma 2.

```
1:10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

- Forma 3.

```
seq(10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

- Forma 4.

```
c(1,2,3,4,5,6,7,8,9,10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Notas:

- La sintaxis de la forma 1 contiene 3 argumentos:

seq(from = 1, to = 10, by = 1)

¿Cómo podemos buscar información sobre la función seq?

```
?seq
```

B. Genere el siguiente vector: "year_2015", "year_2016", "year_2017", "year_2018"

```
year <- 2015:2018
paste0("year_", year)
```

```
## [1] "year_2015" "year_2016" "year_2017" "year_2018"
```

3.2. ¿Cómo seleccionar valores de un vector?

Por lo general, existen múltiples formas de seleccionar valores de un vector. Lo más común es seleccionarlos a través de su ubicación.

Genermos un elemento considerando el vector anterior, pero haciendo una variación de la función paste0

[2]Preguntas en base al curso generado por Victor Macías para la Universidad de Chile

```r
v_year <- paste0("year", sep = "_", year)
```

Ahora seleccionemos el primer elemento del vector:

```r
v_year[1]
```

```
## [1] "year_2015"
```

Veamos que pasa cuando ponemos el signo "-" delante.

```r
v_year[-1]
```

```
## [1] "year_2016" "year_2017" "year_2018"
```

Dicho símbolo, actua como negación, esto en base a la lógica clásica.

En el caso de querer selecionar el último valor del vector, podríamos poner el número de dicho valor. Sabemos que el vector tiene 4 valores, por lo tanto podríamos cambiar el 1 por 4:

```r
v_year[4]
```

```
## [1] "year_2018"
```

Otra forma, podría ser combinandolo con la función length

```r
v_year[length(v_year)]
```

```
## [1] "year_2018"
```

Esta última forma es muy utilizada, debido a que genera la posibilidad de reacomodar la posición del último valor, debido a que depende del largo del vector para determinar su valor. Veamos que arroja al utilizarla:

```r
length(v_year)
```

```
## [1] 4
```

En el caso de querer seleccionar varios valores, podemos realizarlo de la siguiente manera:

```r
v_year[c(1,3)]
```

```
## [1] "year_2015" "year_2017"
```

Y aplica lo mismo en caso de la negación:

```r
v_year[-c(1,3)]
```

```
## [1] "year_2016" "year_2018"
```

O también

```
v_year[c(-1,-3)]
```

```
## [1] "year_2016" "year_2018"
```

3.3. Pregunta 2

El vector x, contiene las edades de 5 estudiantes

```
x = c("Pedro"=17, "Ana"=NA, "Maya"=23, "Max"=NA, "Paula"=20)
```

A. Defina un vector y que excluya los missing values (NA).

```
y=x[-c(2,4)] #forma 1
y
```

```
## Pedro  Maya Paula
##    17    23    20
```

```
y=x[c(-2,-4)] # forma 2
y
```

```
## Pedro  Maya Paula
##    17    23    20
```

```
y=x[!is.na(x)] # forma 3
y
```

```
## Pedro  Maya Paula
##    17    23    20
```

```
y = na.omit(x) # forma 4
y
```

```
## Pedro  Maya Paula
##    17    23    20
## attr(,"na.action")
## Ana Max
##   2   4
## attr(,"class")
## [1] "omit"
```

B. Asigne las edades de Pedro y Ana a un vector z

```
z <- x[c("Pedro","Ana")] #forma 1
z
```

```
## Pedro   Ana
##    17    NA
```

```r
z <- x[c(1,2)] #forma 2
z
```

```
## Pedro   Ana
##    17    NA
```

C. Calcule el promedio aritmético de las edades de los 5 estudiantes

Veamos un primera manera:

```r
mean(x)
```

```
## [1] NA
```

¿Qué cree que pasa en este caso? ¿Cuál es el error?

Veamos una manera alternativa de solucionar esto:

```r
mean(x, na.rm = TRUE)
```

```
## [1] 20
```

Recuerdo que la media muestral se define como:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{1}$$

Veamos la manera de realizar lo mismo, pero programando nosotros la función para el promedio:

1. Primero deberíamos sumar todos los valores del vector.
2. Dividir la suma de dichos números, sobre el total de valores del vector.

¿Recuerda algunas funciones vistas que podamos utilizar?

Veamos un primer intento:

```r
sum(x)/length(x)
```

```
## [1] NA
```

Probemos esto sobre un elemento sin valores vacíos:

```r
sum(y)/length(y)
```

```
## [1] 20
```

Con esto ya sabemos el problema. Intentemos lidiar con la posibilidad de tener valores NA dentro de nuestro vector. Veamos como realizar la suma:

```r
sum(x[!is.na(x)])
```

```
## [1] 60
```

Y como realizar el conteo:

```
length(x[!is.na(x)])
```

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

Combinemos ambas cosas:

```
sum(x[!is.na(x)])/length(x[!is.na(x)])
```

```
## [1] 20
```

4. Fechas Relevantes

Unidad	Evaluación	Ponderación	Fecha
Unidad I	~~Evaluación diagnóstica~~		25/03/2021
	~~Evaluación Individual Participación~~	(5 %)	05/04/2021
	~~Evaluación Grupal~~	(15 %)	27/04/2021 - 04/05/2021
	~~Evaluación Individual - Sumativa I~~	(30 %)	11/05/2021
Unidad II	Evaluación Grupal	(15 %)	20/06/2021
	Evaluación Grupal - Sumativa II	(15 %)	27/06/2021
Unidad III	Evaluación Formativa		27/06/2021
	Evaluación Grupal Sesión I- Sumativa III	(20 %)	08/07/2021
	Evaluación grupal Sesión II- Sumativa III	(20 %)	13/07/2021