



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO®



Tecnológico Nacional De México

Instituto Tecnológico De Tijuana

Subdirección Académica

Departamento de Sistemas y Computación

Semestre Enero - Junio 2022

Ingeniería Informática

Minería De Datos

Introducción a git

Práctica 2

Perez Ortega Victoria Valeria

No.18210718

Díaz Ruiz Uriel

No.18210839

JOSE CHRISTIAN ROMERO HERNANDEZ

Tijuana, B.C. a 07 de Marzo de 2022.



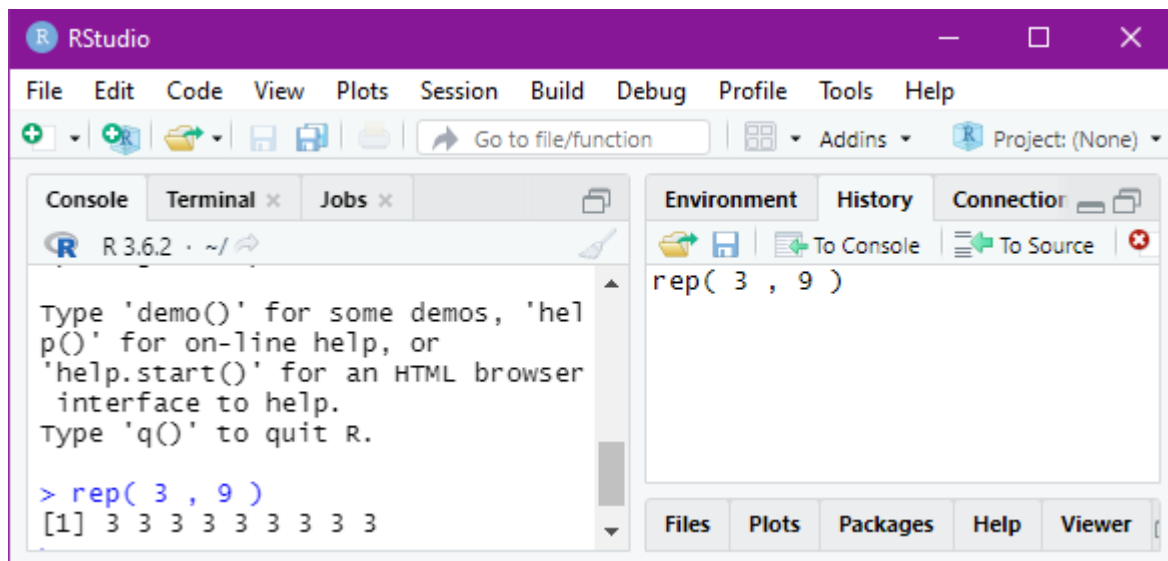
Funciones

Practique encontrar 20 funciones más en R y haga un ejemplo de ello.

1 Rep: esta función permite repetir un número n número de veces, primero colocamos el número y luego la cantidad.

Ejemplo:

```
rep( 3 , 9 )
```



2 Con esta función podemos hacer una operación, es un poco más compleja que la anterior.

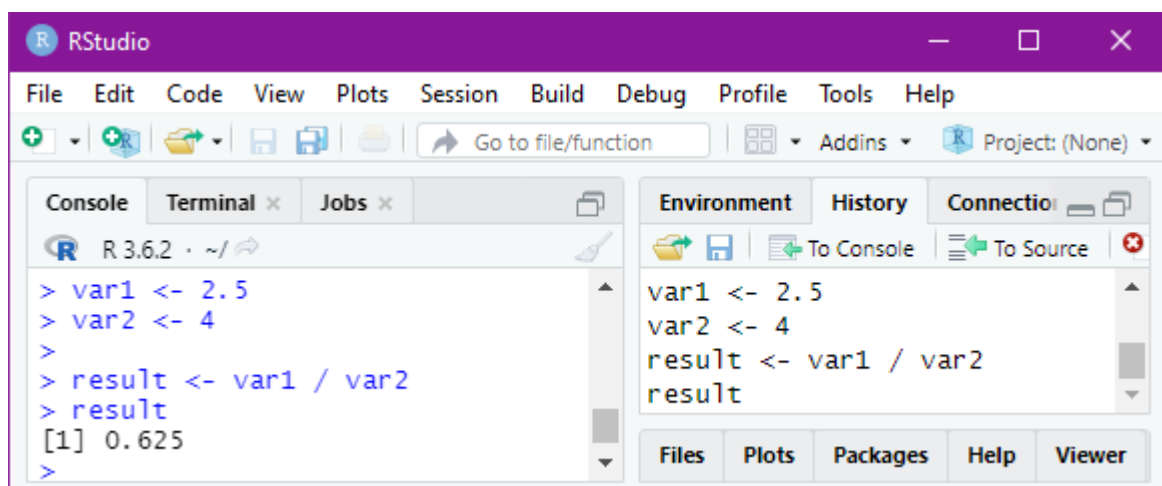
Ejemplo:

```
var1 <- 2.5
```

```
var2 <- 4
```

```
resultado <- var1 / var2
```

```
resultado
```





3. Con esta función podemos contar las veces que uno engaña, Primero comenzamos a partir de qué valor queremos que empiece a contar y luego el resto de la función.

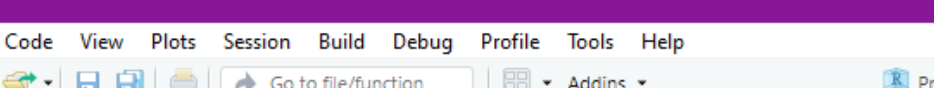
Ejemplo:

```
contador <- 0
```

```
mientras ( contador < 8 ){  
  imprimir( " Contador: " )  
  imprimir ( contador )  
  contador <- contador + 1  
}
```

The screenshot shows the RStudio environment. The console pane on the left displays the execution of a while loop. The code entered is: `counter <- 0`, followed by a while loop `while (counter < 8){` containing `print(" Counter: ")`, `print (counter)`, and `counter <- counter + 1`. The output shows the loop executing 8 times, printing the counter value from 0 to 7. The environment pane on the right shows the current state of the workspace, including variables like `rep(3 , 9)`, `var1 <- 2.5`, `var2 <- 4`, `resultado <- var1 / var2`, `resultado`, `var1 <- 2.5`, `var2 <- 4`, `result <- var1 / var2`, `result`, `counter <- 0`, and the while loop code itself.

```
RStudio  
File Edit Code View Plots Session Build Debug Profile Tools Help  
Go to file/function Addins Project: (None)  
Console Terminal x Jobs x  
R 3.6.2 · ~/   
> counter <- 0  
>  
> while ( counter < 8 ){  
+   print( " Counter: " )  
+   print (counter)  
+   counter <- counter + 1  
+ }  
[1] " Counter: "  
[1] 0  
[1] " Counter: "  
[1] 1  
[1] " Counter: "  
[1] 2  
[1] " Counter: "  
[1] 3  
[1] " Counter: "  
[1] 4  
[1] " Counter: "  
[1] 5  
[1] " Counter: "  
[1] 6  
[1] " Counter: "  
[1] 7  
> |  
Environment History Connections  
To Console To Source  
rep( 3 , 9 )  
var1 <- 2.5  
var2 <- 4  
resultado <- var1 / var2  
resultado  
var1 <- 2.5  
var2 <- 4  
result <- var1 / var2  
result  
counter <- 0  
while ( counter < 8 ){  
  print( " Counter: " )  
  print (counter)  
  counter <- counter + 1  
}  
Files Plots Packages Help Viewer
```



The screenshot shows the RStudio application window. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for saving, opening, and navigating files. The main workspace is divided into several panes. On the left, the 'Source' pane is active, showing a file named 'trunc.R'. The 'Console' pane at the bottom left displays the R prompt and the execution of the `trunc(4.523)` command, resulting in the output `[1] 4`. The 'Environment' pane on the right shows the current environment with variables `round(15.845,digits = 1)`, `rep(3 , 9)`, and `trunc(4.523)`. The 'Plots' pane is also visible on the right.



7. Sqrt: esta función te permite calcular la raíz cuadrada de un número.

Ejemplo:
`sqrt(26)`

The screenshot shows the RStudio interface. The console pane on the left displays the following commands and output:

```
R 3.6.2 ~/  
> trunc(2.987)  
[1] 2  
> sqrt(9)  
[1] 3  
> |
```

The environment pane on the right shows the following objects:

```
4 != 3  
trunc(2.987)  
sqrt(9)
```

8. Con esta función podemos hacer una operación, más sencilla y usando variables.

Ejemplo:
`A <- 20`
`B <- 6`
`R <- -A - B`
`R`

The screenshot shows the RStudio interface. The console pane on the left displays the following commands and output:

```
R 3.6.2 ~/  
> sqrt(9)  
[1] 3  
> A <- 20  
> B <- 6  
> R <- A - B  
> R  
[1] 14  
>
```

The environment pane on the right shows the following objects:

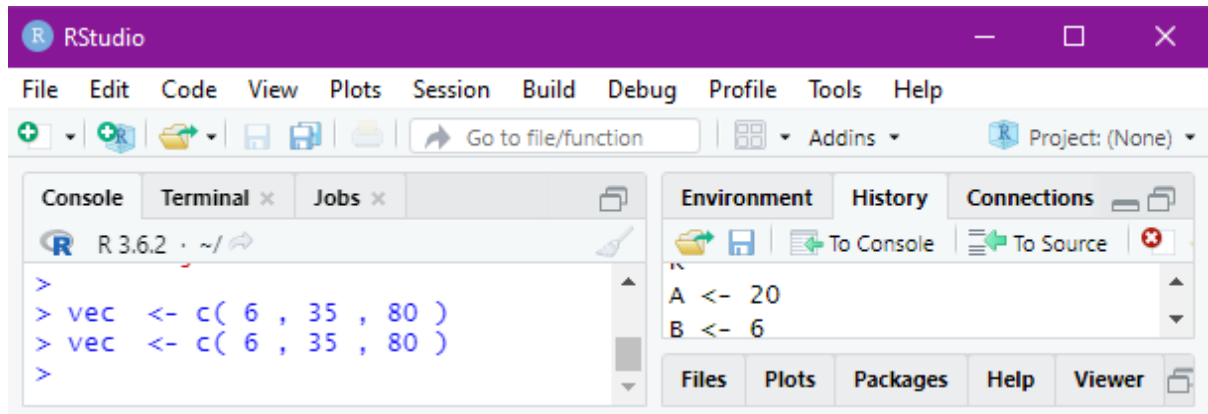
```
sqrt(9)  
A <- 20  
B <- 6  
R <- A - B  
R
```



9. C: esta función te permite combinar y crear un vector

Ejemplo:

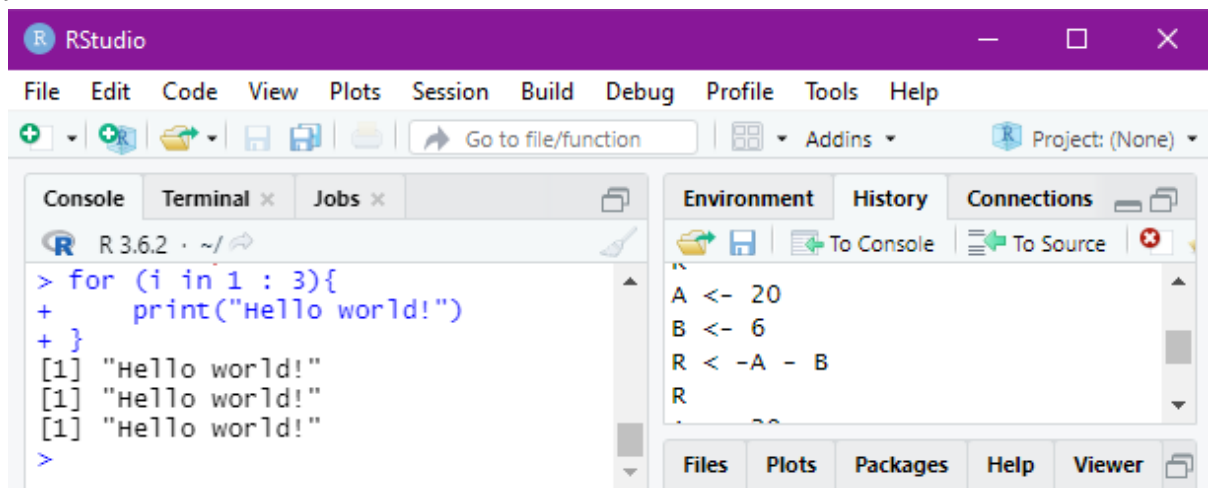
```
vec <- c( 6 , 35 , 80 )
```



10. Con esto imprimimos la frase uno quiere y la cantidad de veces que uno quiere

Ejemplo:

```
para ( yo en 1 : 3 ){  
  imprimir( " ¡Hola mundo! " )  
}
```





11 Pegar: esta función concatenar vectores después de convertirlos en caracteres.

Ejemplo:

```
mensaje <- paste ('Hola ', ' Mundo ')
```

```
R 3.6.2 ~/  
[1] "Hello world!"  
[1] "Hello world!"  
> mensaje <- paste ('Hello','world')  
> |
```

12 La siguiente función podemos imprimir el mensaje repetidamente y luego podemos enumerarlos

Ejemplo:

```
para (yo en 0 : 2 ){  
  imprimir (yo )  
  imprimir ( " Minería de datos " )  
}
```

```
R 3.6.2 ~/  
> for (i in 0:2){  
+   print (i)  
+   print("Data Mining")  
+ }  
[1] 0  
[1] "Data Mining"  
[1] 1  
[1] "Data Mining"  
[1] 2  
[1] "Data Mining"  
>
```



13 longitud: esta función muestra el número de elementos en un vector.

Ejemplo:

```
x <- c( 1 , 2 , 3 , 4 , 5 , 6 )
```

```
longitud ( x )
```

The screenshot shows the RStudio interface. The console pane on the left displays the following code and output:

```
R 3.6.2 ~/  
> x <- c(1,2,3,4,5,6)  
> length(x)  
[1] 6  
>  
> |
```

The environment pane on the right shows the current environment with the following objects:

```
x <- c(1,2,3,4,5,6)  
length(x)
```

14 Con esta función muestra que tipo de dato es.

Ejemplo:

```
tipo de( 2 )
```

The screenshot shows the RStudio interface. The console pane on the left displays the following code and output:

```
R 3.6.2 ~/  
[1] 4  
> typeof(2)  
[1] "double"  
>  
> |
```

The environment pane on the right shows the current environment with the following objects:

```
length(x)  
typeof(2)
```




15 Esta función genera un bucle hasta que decidimos detenerlo

Ejemplo:

```
mientras ( VERDADERO ){  
  print( " Hola buenos dias.... " )  
}
```

The screenshot shows the RStudio interface. The console on the left displays the output of a while loop: 15 lines of "[1] Hello good Morning....". The environment pane on the right shows the code for a for loop: `for(i in 1:5){ print("Hello world!") }`, followed by `message<-paste('Hello','world')`, `for(i in 0:5){ print(i) print("Data Mining") }`, `x<-c(1,2,3,4)`, `length(x)`, `typeof(2)`, `while(TRUE){ print("Hello good Morning....") }`, `stop`, and `while(TRUE){ print("Hello good Morning....") }`.

16 Con esta función podemos saber si el número que ingresamos es el doble o no.

Ejemplo:

```
es.doble( 10 )
```

The screenshot shows the RStudio interface. The console on the left displays the output of the `es.doble(10)` function: `[1] TRUE`. The environment pane on the right shows the code for the `length(x)` and `is.doble(10)` functions.



17 Seq: esta función te permite generar una secuencia

Ejemplo:
secuencia(3 , 9)

The screenshot shows the RStudio environment. In the Console pane on the left, the following commands and output are visible:

```
R 3.6.2 ~/  
[1] TRUE  
> seq(3,9)  
[1] 3 4 5 6 7 8 9  
> |
```


In the Environment pane on the right, the objects 'is.double(10)' and 'seq(3,9)' are listed. The 'seq(3,9)' object is highlighted, showing its value as a sequence of integers from 3 to 9.

18 Con esta función podemos saber si el número que ingresamos es un número entero o no.

Ejemplo:
es.entero(19)

The screenshot shows the RStudio environment. In the Console pane on the left, the following commands and output are visible:

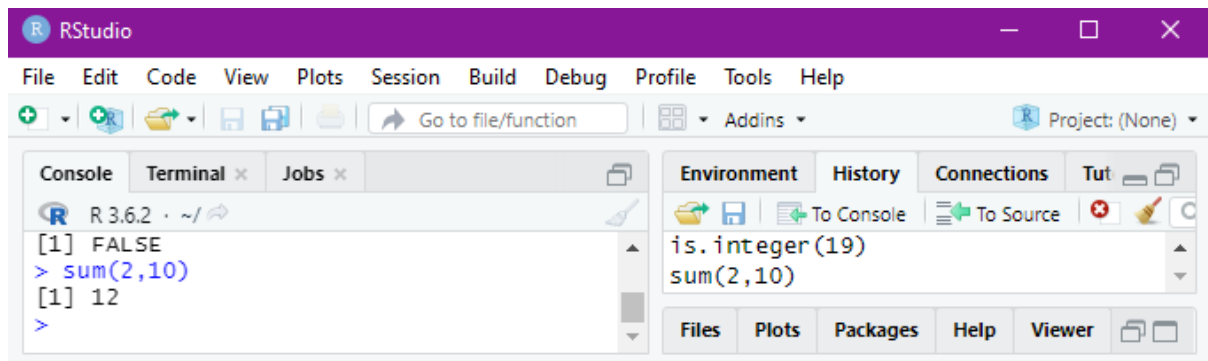
```
R 3.6.2 ~/  
[1] 3 4 5 6 7 8 9  
> is.integer(19)  
[1] FALSE  
>
```


In the Environment pane on the right, the objects 'seq(3,9)' and 'is.integer(19)' are listed. The 'is.integer(19)' object is highlighted, showing its value as FALSE.



19 Suma: esta función suma todos los valores dentro del rango que establezcas.

Ejemplo:
suma(2,10)



20 round: esta función redondea un número a los lugares decimales especificados.

Ejemplo:
redondo (15.845 , dígitos = 1)

