

Actividad 1 - Análisis de Conceptos.

Métodos Numéricos.

Ingeniería en Desarrollo de Software

Tutor: Miguel Ángel Rodríguez Vega.

Alumnos: Carlos Ariel Nicolini.

Fecha: 16/10/2023

Índice

Introducción	3
Descripción	4
Justificación	5
Desarrollo	6
• Descarga de Rstudio	6
• Carga de Valores_numericos.R	8
• Ejecucion de Valores_numericos.R.....	9
Conclusión	35
Referencias.....	36

Introducción

En el mundo real al aplicar las matemáticas nos podemos encontrar a menudo con problemas que no pueden ser resuelto de manera exacta, las cuales debemos abordar con algún procedimiento numérico para poder solucionar.

Un método numérico es un procedimiento mediante el cual se puede obtener de manera aproximada una solución realizando cálculos aritméticos y lógicos. El procedimiento consiste en una lista de instrucciones que especifican una secuencia de operaciones algebraicas y algoritmos, que resultan en una aproximación de la solución de dicho problema. El resultado dependerá del algoritmo utilizado y las herramientas con las que se resuelva. El objetivo principal del análisis numérico es encontrar soluciones aproximadas para problemas complejos.

Descripción

En esta actividad utilizaremos el programa Rstudio para interpretar un archivo con instrucciones, donde aprenderemos conceptos básicos de los métodos numéricos.

Instalaremos el programa Rstudio, descargaremos un archivo con instrucciones, las cuales ejecutaremos en el programa y se analizarán los resultados.

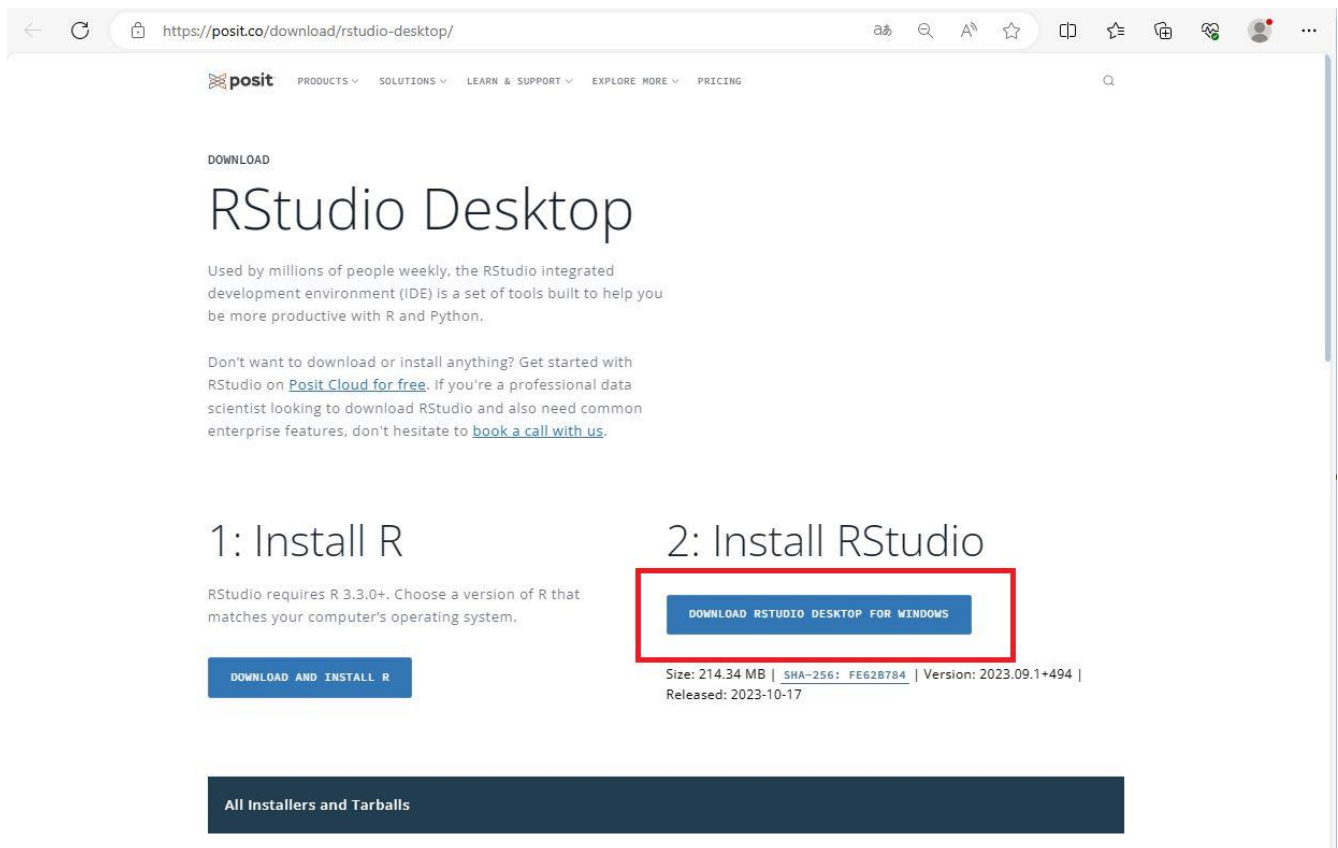
Justificación

En esta actividad es necesario realizarla para entender los funcionamientos de los métodos numéricos, como aplicarlos y también el uso del Rstudio, el cual de mano con los métodos numéricos nos van a ayudar a resolver de la manera más aproximada problemas de compleja solución.

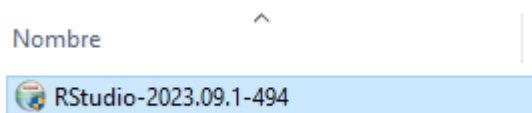
Desarrollo

Descarga de Rstudio

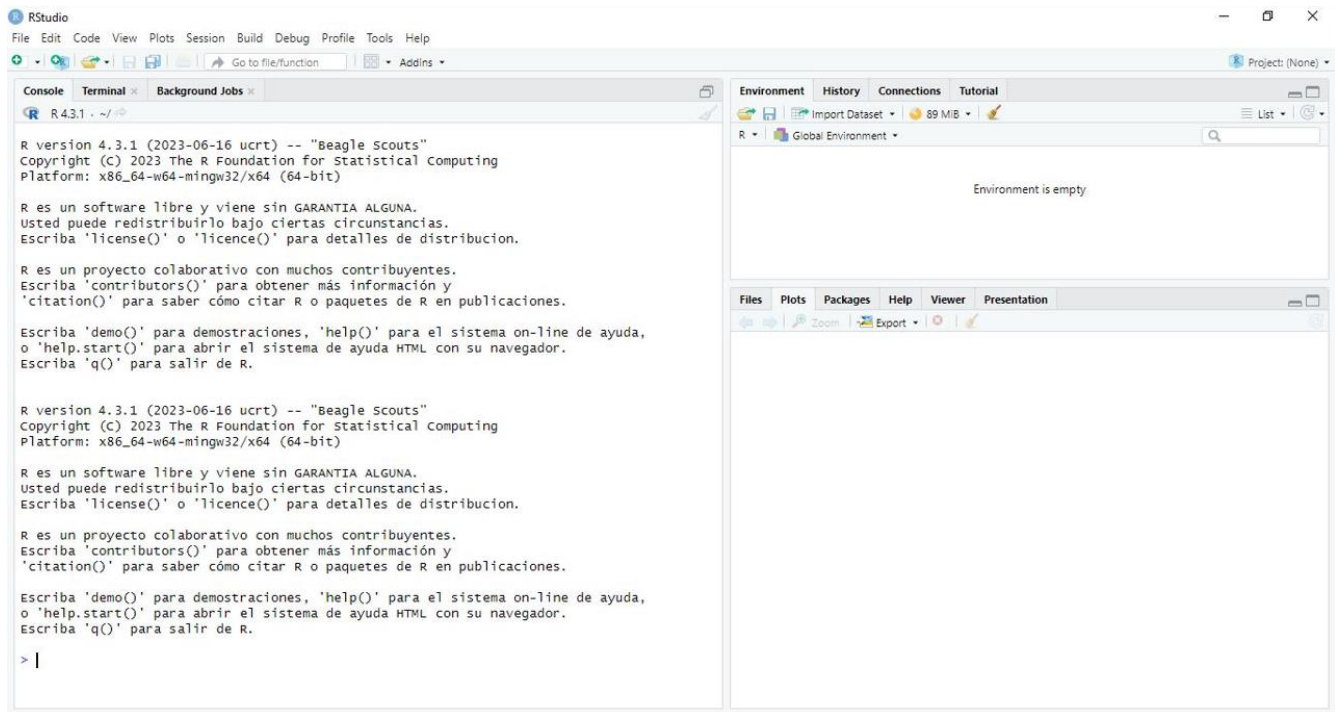
Ingresamos al siguiente enlace (<https://www.rstudio.com/products/rstudio/download/>),
descargamos el software Rstudio en el boton señalado con rojo en la imagen.



Una vez descargado el software, lo ubicamos en la carpeta donde se descargo, lo ejecutamos e instalamos el programa.



Al terminar la instalacion, ejecutamos el Rstudio y con estos pasos ya finalizamos la instalacion requerida para este punto.



Carga de valores_numericos.R

En esta parte del ejercicio ingresamos al siguiente enlace (http://umi.edu.mx/coppel/IDS/plataforma/files/programasenr/Valores_numericos.R), la cual nos abra un documentos con pasos a ejecutar. Descargamos esa pagina y la guardamos como un archivo txt en nuestro computadora. Una vez descargado, abrimos el programa Rstudio, elegimos en la barra de herramientas del Rstudio la opcion File y en el panel desplegable elegimos la opcion Open File, Elegimos el documento descargado (valores_numericos.R) y nos abra en el programa el documento descargado.

```

1 ##### VALORES NUMERICOS #####
2
3 ## DÍGITOS ENTEROS Y FRACCIONES ##
4
5
6 # Manejo de la representación numérica con lenguaje R a través de
7 # funciones, las representaciones pueden ser con:
8 # el signo igual(=) o con las teclas (<=)
9 # Para representar cualquier cantidad y a una variable se escribe:
10 a = 6384671
11
12 # Para que R, nos regrese el valor almacenado en la variable a, solo
13 # escribimos a y Enter
14 a
15
16 # Si lo que deseamos es guardar un número fraccionario, escribimos:
17 b <- 0.5342198
18
19 # Y para desplegar el valor, se escribe b y Enter
20 b
21
22 # También, se pueden realizar operaciones con los valores y obtener el
23 # resultado
24 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
25
26

```

```

R version 4.3.1 (2023-06-16 ucrt) -- "Beagle Scouts"
Copyright (c) 2023 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribución.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> |

```


Ejecución de valores_numericos.R

En este punto del ejercicio ejecutaremos los comando que vienen en el archivo valores_numericos.R y daremos una breve explicacion.

Con el siguiente comando asignados un valor a la variable a (puede ser con el signo = o con <-).

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Displays the script 'Valores_numericos.R.txt' with the following content:


```
1 ##### VALORES NUMERICOS #####
2
3 ## DÍGITOS ENTEROS Y FRACCIONES ##
4
5
6 # Manejo de la representación numérica con lenguaje R a través de
7 # funciones, las representaciones pueden ser con:
8 # el signo igual(=) o con las teclas (<-)
9 # Para representar cualquier cantidad y a una variable se escribe:
10 a = 6384671
11
12 # Para que R, nos regrese el valor almacenado en la variable a, solo
13 # escribimos a y Enter
14 a
15
16 # Si lo que deseamos es guardar un número fraccionario, escribimos:
17 b <- 0.5342198
18
19 # Y para desplegar el valor, se escribe b y Enter
20 b
21
22 # También, se pueden realizar operaciones con los valores y obtener el
23 # resultado
24 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
25
```
- Environment Pane:** Shows the 'Global Environment' with a table of values:

Variable	Value
a	6384671
- Console:** Shows the R version information and the execution of the script:


```
R 4.3.1 ~ /
R version 4.3.1 (2023-06-16 ucrt) -- "Beagle Scouts"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

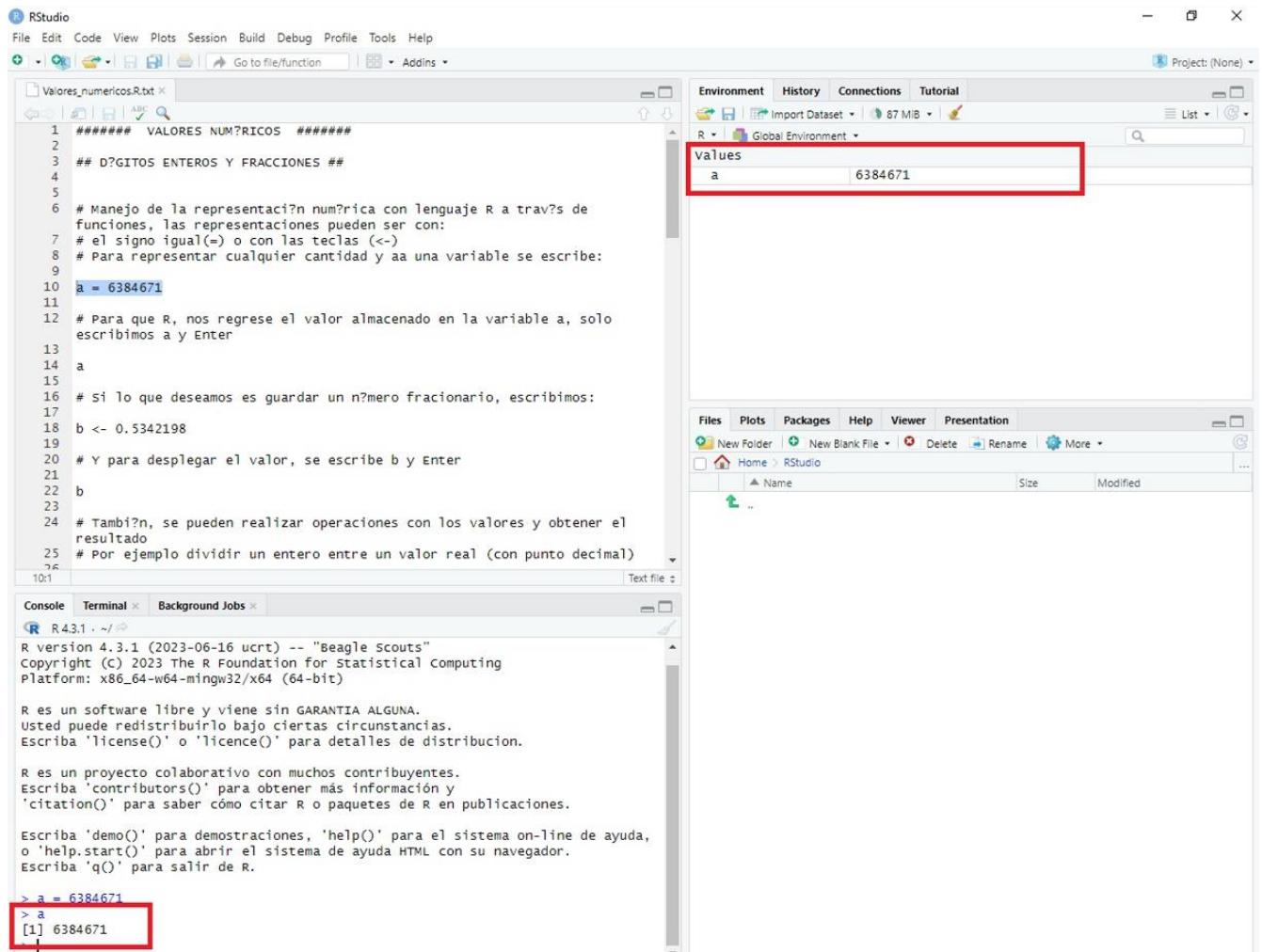
R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribución.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
```

Si escribimos `a` y damos enter, R nos regresara el valor almacenado en la variable `a`.

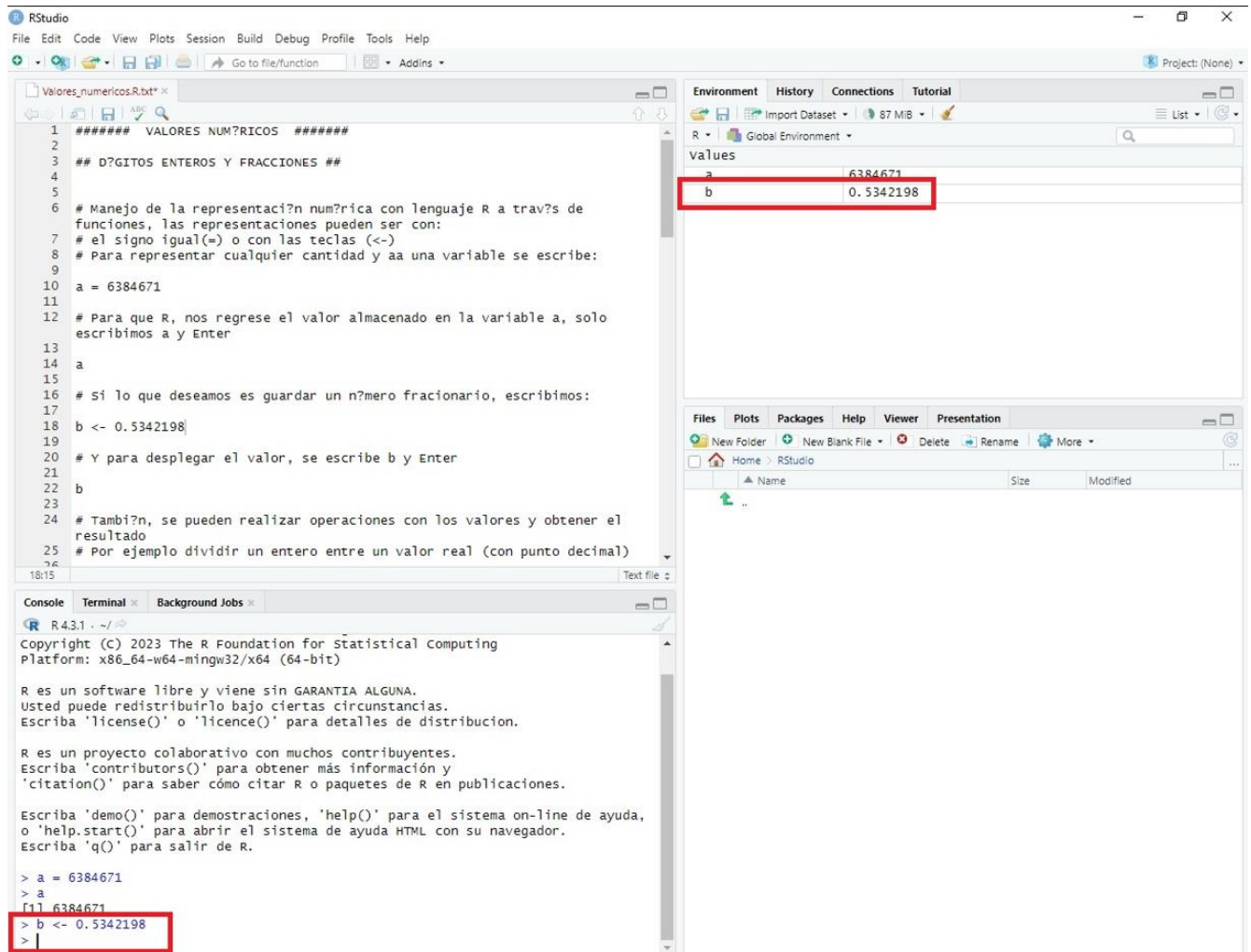


The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains an R script file named `Valores_numericos.R.txt`. The script includes comments in Spanish and the following code:

```
1 ##### VALORES NUMERICOS #####
2
3 ## DÍGITOS ENTEROS Y FRACCIONES ##
4
5
6 # Manejo de la representación numérica con lenguaje R a través de
7 # funciones, las representaciones pueden ser con:
8 # el signo igual(=) o con las teclas (<-)
9 # Para representar cualquier cantidad y a una variable se escribe:
10 a = 6384671
11
12 # Para que R, nos regrese el valor almacenado en la variable a, solo
13 # escribimos a y Enter
14 a
15
16 # Si lo que deseamos es guardar un número fraccionario, escribimos:
17 b <- 0.5342198
18
19 # Y para desplegar el valor, se escribe b y Enter
20 b
21
22
23 # También, se pueden realizar operaciones con los valores y obtener el
24 # resultado
25 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
```
- Environment Pane:** Located on the right, it shows the `Global Environment` with a table of values. The variable `a` is listed with its value `6384671`. This section is highlighted with a red rectangle.
- Console:** At the bottom, it shows the R session output. It includes the R version information (4.3.1) and the execution of the commands from the script. The final output shows the command `> a` resulting in `[1] 6384671`. This output is also highlighted with a red rectangle.

Para guardar un valor fraccionario escribimos la letra de la variable en la que se va a guardar, seguido por los signos <- y el valor a guardar.



The screenshot shows the RStudio interface. The script editor on the left contains R code for assigning values to variables 'a' and 'b'. The Environment pane on the right shows the current values of these variables. The console at the bottom shows the execution of the code.

```
1 ##### VALORES NUMERICOS #####
2
3 ## DÍGITOS ENTEROS Y FRACCIONES ##
4
5
6 # Manejo de la representación numérica con lenguaje R a través de
7 # funciones, las representaciones pueden ser con:
8 # el signo igual(=) o con las teclas (<-)
9 # Para representar cualquier cantidad y a una variable se escribe:
10
11 a = 6384671
12
13 # Para que R, nos regrese el valor almacenado en la variable a, solo
14 # escribimos a y Enter
15
16 a
17
18 # Si lo que deseamos es guardar un número fraccionario, escribimos:
19
20 b <- 0.5342198
21
22 # Y para desplegar el valor, se escribe b y Enter
23
24 b
25
26 # También, se pueden realizar operaciones con los valores y obtener el
27 # resultado
28
29 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
```

Environment pane:

Variable	Value
a	6384671
b	0.5342198

Console:

```
R 4.3.1 ~/>
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribución.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> |
```

Para mostrar el valor de la variable **b**, escribimos **b** y presionamos enter.

The screenshot displays the RStudio environment with the following components:

- Editor:** Contains a script file named `Valores_numericos.R.txt` with the following R code:

```
1 ##### VALORES NUMERICOS #####
2
3 ## DÍGITOS ENTEROS Y FRACCIONES ##
4
5
6 # Manejo de la representación numérica con lenguaje R a través de
7 # funciones, las representaciones pueden ser con:
8 # el signo igual(=) o con las teclas (<-)
9 # Para representar cualquier cantidad y a una variable se escribe:
10 a = 6384671
11
12 # Para que R, nos regrese el valor almacenado en la variable a, solo
13 # escribimos a y Enter
14 a
15
16 # Si lo que deseamos es guardar un número fraccionario, escribimos:
17
18 b <- 0.5342198
19
20 # Y para desplegar el valor, se escribe b y Enter
21
22 b
23
24 # También, se pueden realizar operaciones con los valores y obtener el
25 # resultado
26 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
```
- Environment:** Located on the right, it shows the current environment with two variables:

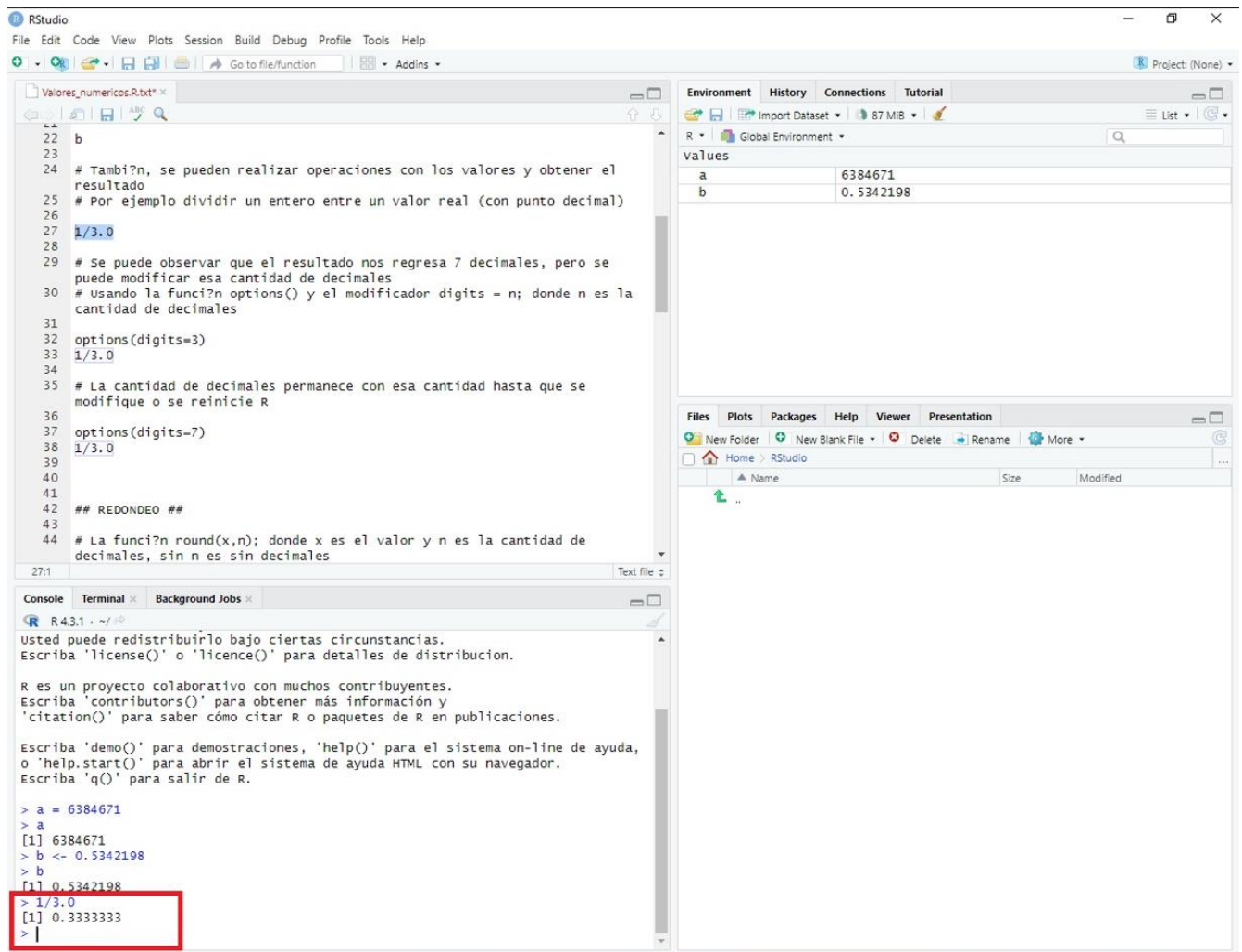
Variable	Value
a	6384671
b	0.5342198

The row for variable **b** is highlighted with a red box.
- Console:** Located at the bottom, it shows the execution history:

```
> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> |
```

The last two lines of the console output are highlighted with a red box.

Para dividir un numero entero entre un valor real (con punto decimal), lo representamos de la siguiente manera.



The screenshot shows the RStudio interface. The script editor on the left contains R code for dividing an integer by a real number. The console at the bottom shows the execution of these commands. The Environment pane on the right shows the values of variables 'a' and 'b'.

Script Editor Code:

```
22 b
23
24 # Tambi?n, se pueden realizar operaciones con los valores y obtener el
25 # resultado
26 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
27 1/3.0
28
29 # Se puede observar que el resultado nos regresa 7 decimales, pero se
30 # puede modificar esa cantidad de decimales
31 # Usando la funci?n options() y el modificador digits = n; donde n es la
32 # cantidad de decimales
33 options(digits=3)
34 1/3.0
35
36 # La cantidad de decimales permanece con esa cantidad hasta que se
37 # modifique o se reinicie R
38 options(digits=7)
39 1/3.0
40
41
42 ## REDONDEO ##
43
44 # La funci?n round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
```

Console Output:

```
R 4.3.1 ~ /
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener m?s informaci?n y
'citation()' para saber c?mo citar R o paquetes de R en publicaciones.

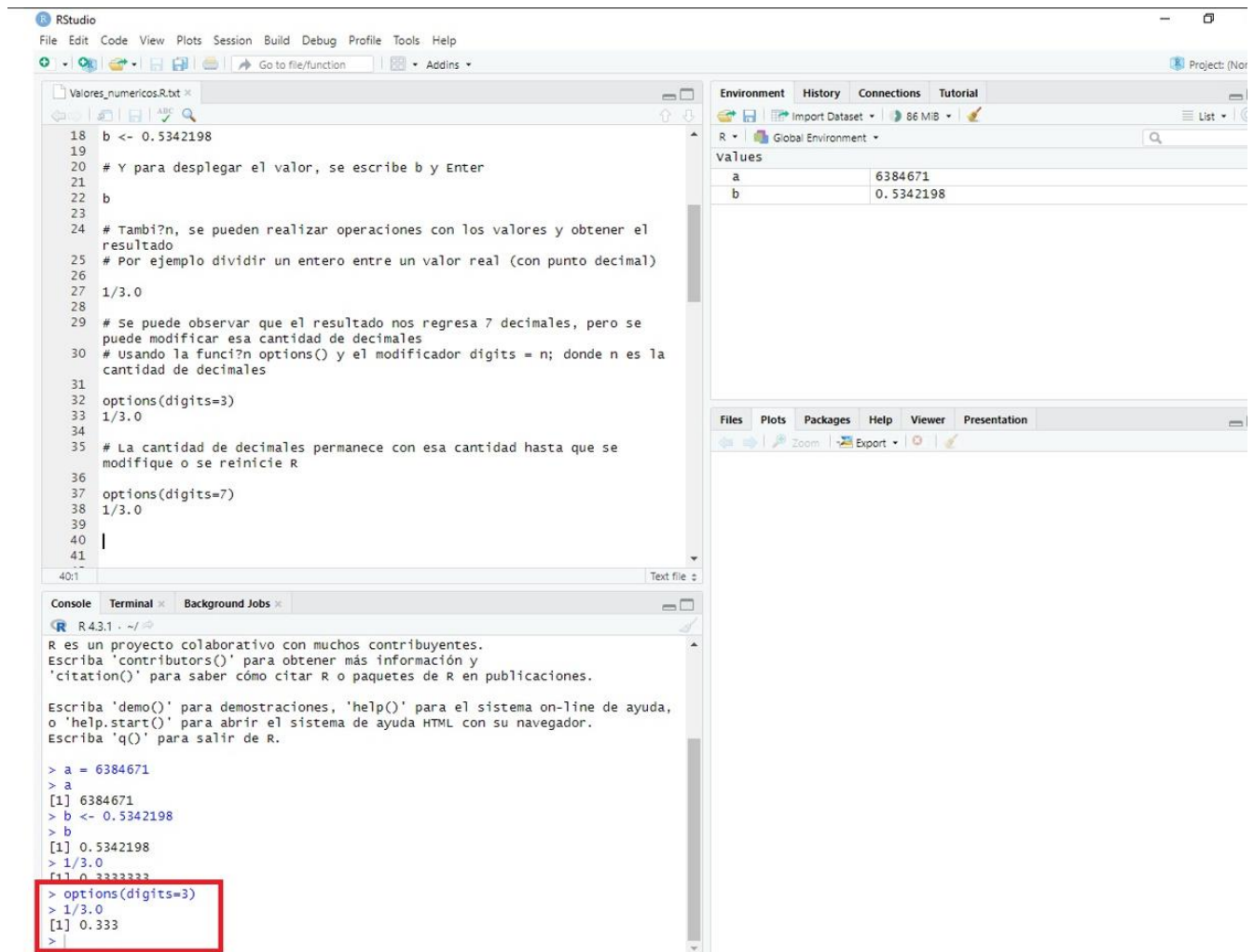
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> |
```

Environment Pane:

values	
a	6384671
b	0.5342198

Para que el resultado se muestre con 3 dígitos, se puede usar la función **options(digits=n)** donde **n** es la cantidad de decimales a mostrar, en este caso **3**.



The screenshot shows the RStudio interface. The script editor on the left contains the following R code:

```
18 b <- 0.5342198
19
20 # Y para desplegar el valor, se escribe b y Enter
21 b
22
23 # Tambi n, se pueden realizar operaciones con los valores y obtener el
24 # resultado
25 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
26 1/3.0
27
28 # Se puede observar que el resultado nos regresa 7 decimales, pero se
29 # puede modificar esa cantidad de decimales
30 # usando la funci n options() y el modificador digits = n; donde n es la
31 # cantidad de decimales
32 options(digits=3)
33 1/3.0
34
35 # La cantidad de decimales permanece con esa cantidad hasta que se
36 # modifique o se reinicie R
37 options(digits=7)
38 1/3.0
39
40
41
```

The Environment pane on the right shows the following values:

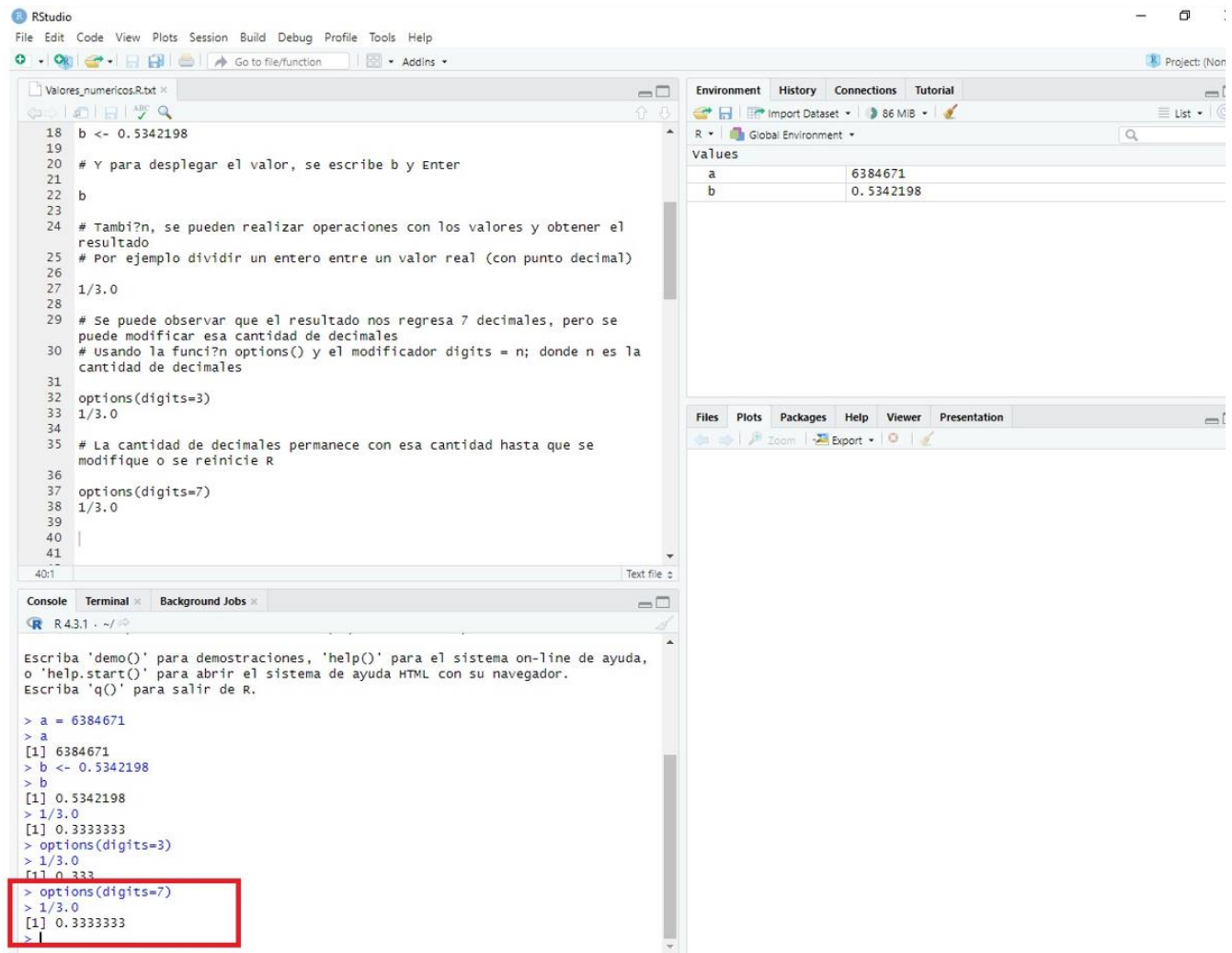
Variable	Value
a	6384671
b	0.5342198

The Console at the bottom shows the following output:

```
> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> options(digits=3)
> 1/3.0
[1] 0.333
>
```

The last three lines of the console output are highlighted with a red box.

Con esta modificacion regresamos a los 7 decimales.



The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for setting variables and controlling decimal output.
- Environment:** Shows the current environment with variables 'a' and 'b'.
- Console:** Shows the execution history, with the last few lines highlighted by a red box.

Source Editor Code:

```
18 b <- 0.5342198
19
20 # y para desplegar el valor, se escribe b y Enter
21
22 b
23
24 # Tambien, se pueden realizar operaciones con los valores y obtener el
25 # resultado
26 # Por ejemplo dividir un entero entre un valor real (con punto decimal)
27 1/3.0
28
29 # Se puede observar que el resultado nos regresa 7 decimales, pero se
30 # puede modificar esa cantidad de decimales
31 # Usando la funci?n options() y el modificador digits = n; donde n es la
32 # cantidad de decimales
33 options(digits=3)
34 1/3.0
35
36 # La cantidad de decimales permanece con esa cantidad hasta que se
37 # modifique o se reinicie R
38 options(digits=7)
39 1/3.0
40
41
```

Environment:

Values	
a	6384671
b	0.5342198

Console:

```
R 4.3.1 ~ /
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> options(digits=3)
> 1/3.0
[1] 0.333
> options(digits=7)
> 1/3.0
[1] 0.3333333
>
```

Con la función `round(x,n)` ; donde `x` es el valor y `n` la cantidad de decimales, lo redondeamos

The screenshot shows the RStudio interface with a script editor on the left, an environment pane on the top right, and a console at the bottom.

Script Editor (Valores_numericos.R.txt):

```

31 cantidad de decimales
32 options(digits=3)
33 1/3.0
34
35 # La cantidad de decimales permanece con esa cantidad hasta que se
36 # modifique o se reinicie R
37 options(digits=7)
38 1/3.0
39
40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
46 # Si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'mero
50 # nos proporcionar ese n'mero de decimales solicitado y redondeado con
51 # el siguiente d'gito.
52 round(97.5684197, 2)
53

```

Environment Pane:

values	
a	6384671
b	0.5342198

Console (R 4.3.1):

```

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

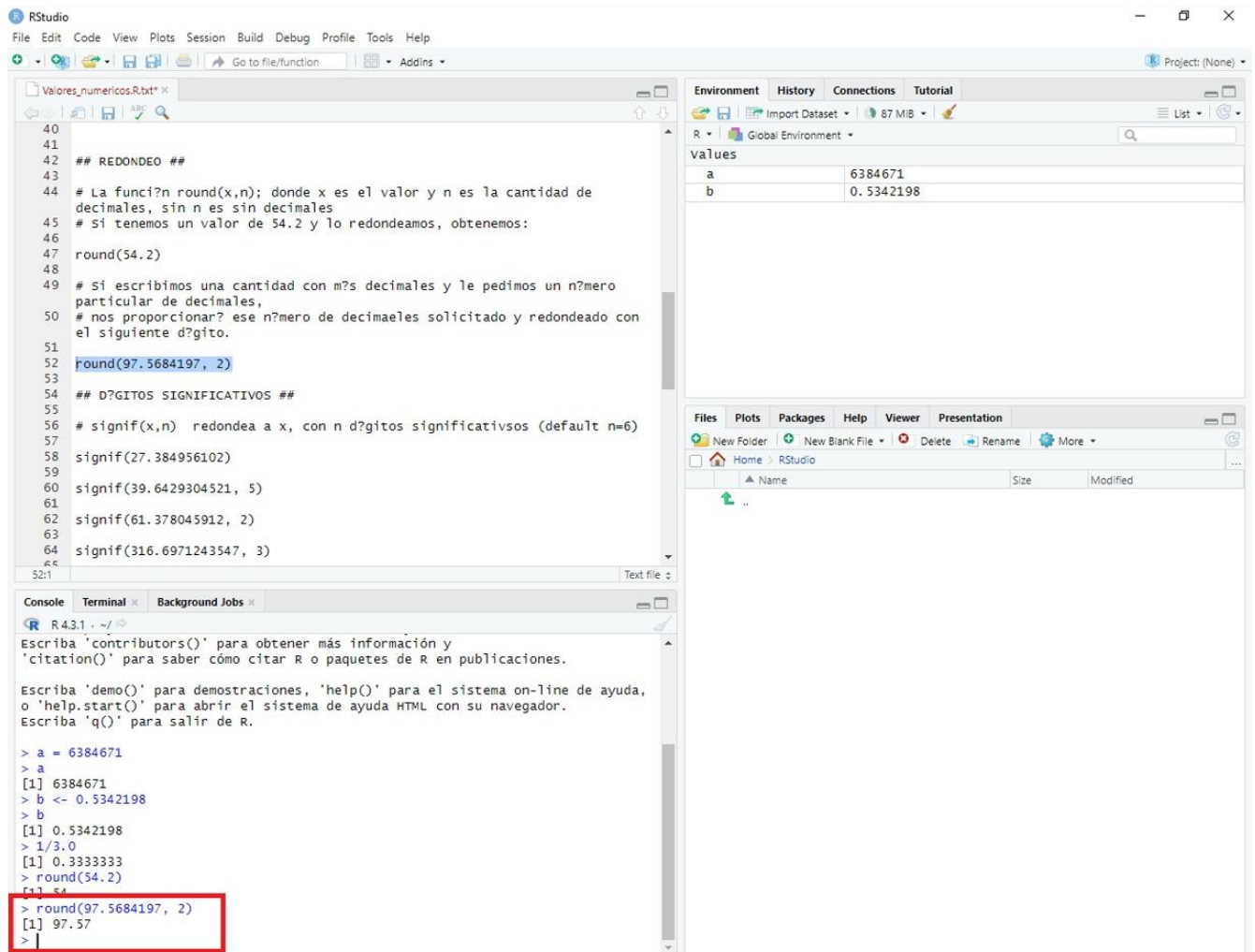
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
>

```

The console output for `round(54.2)` is highlighted with a red box, showing the result `[1] 54`.

Con esta funcion redondearemos un numero, la cual nos entregara el numero de decimales con el siguiente dígito en este caso 2



The screenshot shows the RStudio interface with a script editor on the left, an environment pane on the right, and a console at the bottom.

Script Editor (Valores_numericos.R.txt):

```
40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
46 # Si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'mero
50 # particular de decimales,
51 # nos proporcionar' ese n'mero de decimales solicitado y redondeado con
52 # el siguiente d'gito.
53 round(97.5684197, 2)
54
55 ## D'GITOS SIGNIFICATIVOS ##
56
57 # signif(x,n) redondea a x, con n d'gitos significativos (default n=6)
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)
```

Environment Pane:

values	
a	6384671
b	0.5342198

Console:

```
R 4.3.1 ~ /
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
>
```

La función `signif(x,n)` redondea a `x` con dígitos significativos (default `n=6`).

The screenshot shows the RStudio interface with a script editor on the left, an environment pane on the top right, and a console at the bottom.

Script Editor (Valores_numericos.R.txt):

```

40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
46 # Si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'mero
50 # nos proporcionar' ese n'mero de decimales solicitado y redondeado con
51 # el siguiente d'gito.
52 round(97.5684197, 2)
53
54 ## D'GITOS SIGNIFICATIVOS ##
55
56 # signif(x,n) redondea a x, con n d'gitos significativos (default n=6)
57
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)

```

Environment Pane:

values	
a	6384671
b	0.5342198

Console:

```

R 4.3.1 ~ / ~
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
>

```

La función `signif(x,n)` redondea a `x` con dígitos significativos, en este caso se indica que a 5.

The screenshot shows the RStudio interface with a script editor, environment pane, and console.

Script Editor (Valores_numéricos.R.txt):

```

40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
46
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'mero
50 # nos proporcionar ese n'mero de decimales solicitado y redondeado con
51 # el siguiente d'gito.
52
53 round(97.5684197, 2)
54
55 ## DÍGITOS SIGNIFICATIVOS ##
56
57 # signif(x,n) redondea a x, con n dígitos significativos (default n=6)
58
59 signif(27.384956102)
60
61 signif(39.6429304521, 5)
62
63 signif(61.378045912, 2)
64
65 signif(316.6971243547, 3)
66

```

Environment Pane:

Global Environment	
a	6384671
b	0.5342198

Console:

```

R 4.3.1 ~ /
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
>

```

La función `signif(x,n)` redondea a `x` con dígitos significativos, en este caso se indica que a 2.

The screenshot shows the RStudio interface with a script editor on the left, an environment pane on the top right, and a console at the bottom.

Script Editor (Valores_numericos.R.txt):

```

40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
46 # Si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'ero
50 # particular de decimales,
51 # nos proporcionar' ese n'ero de decimales solicitado y redondeado con
52 # el siguiente d'gito.
53 round(97.5684197, 2)
54
55 ## D'GITOS SIGNIFICATIVOS ##
56
57 # signif(x,n) redondea a x, con n d'gitos significativos (default n=6)
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)
65
66

```

Environment Pane:

values	
a	6384671
b	0.5342198

Console:

```

R 4.3.1 ~ /
> a = 6384671
> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.64293
> signif(61.378045912, 2)
[1] 61
>

```

The console output for `signif(61.378045912, 2)` is highlighted with a red box, showing the result `[1] 61`.

La función `signif(x,n)` redondea a `x` con dígitos significativos, en este caso se indica que a 3.

The screenshot shows the RStudio interface with a script editor on the left and a console at the bottom. The script editor contains R code that demonstrates the use of the `round` and `signif` functions. The console shows the execution of these functions, with the final result of `signif(316.6971243547, 3)` highlighted by a red box.

```

40
41
42 ## REDONDEO ##
43
44 # La función round(x,n); donde x es el valor y n es la cantidad de
45 # decimales, sin n es sin decimales
46 # Si tenemos un valor de 54.2 y lo redondeamos, obtenemos:
47 round(54.2)
48
49 # Si escribimos una cantidad con m's decimales y le pedimos un n'ero
50 # nos proporcionar' ese n'ero de decimales solicitado y redondeado con
51 # el siguiente d'gito.
52 round(97.5684197, 2)
53
54 ## D'GITOS SIGNIFICATIVOS ##
55
56 # signif(x,n) redondea a x, con n d'gitos significativos (default n=6)
57
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)
65

```

Console output:

```

> a
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> |

```

La siguiente función `e <- exp(1)`, asigna un valor a la variable `e` (base de los logaritmos naturales).

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for the `signif` function and variable assignments. The line `e <- exp(1)` is highlighted in blue.
- Environment Pane:** Shows the global environment with variables `a`, `b`, and `e`. The value of `e` is `2.71828182845905`, which is highlighted with a red rectangle.
- Console:** Shows the execution output of the code. The last command executed is `e <- exp(1)`, which is also highlighted with a red rectangle.

```

56 # signif(x,n) redondea a x, con n dígitos significativos (default n=6)
57
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)
65
66 ### DEFINIR VARIABLES Y ASIGNAR VALORES ###
67
68 e <- exp(1) # Asigna un valor a la variable (base de los logaritmos
69 naturales e = 2.718281)
70 e # Imprime el valor que tenga la variable
71
72 x = 0.005 # También se puede usar el símbolo <- en lugar de igual
73
74 x0 = e ** (2*x) # Se la asigna una función a x0
75
76 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
77
78 cat(tex, x0) # Se obtiene los resultados con la instrucción
79 cat, que concatena y convierte a string
80
81 # Otro ejemplo:
82 x0 = 1
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Environment Pane:

Variable	Value
a	6384671
b	0.5342198
e	2.71828182845905

Console:

```

R 4.3.1 ~ /
[1] 6384671
> b <- 0.5342198
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
>

```

Al ingresar `e` y dar enter nos devuelve el valor asignado en `e`.

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for rounding numbers and defining variables. The variable `e` is assigned the value of `exp(1)`.
- Environment:** A table showing the values of variables in the Global Environment. The variable `e` is highlighted with a red box, showing its value as 2.71828182845905.
- Console:** Shows the execution of the code. The output for `e` is highlighted with a red box, showing the value 2.718282.

```
# signif(x,n) redondea a x, con n dígitos significativos (default n=6)
signif(27.384956102)
signif(39.6429304521, 5)
signif(61.378045912, 2)
signif(316.6971243547, 3)

### DEFINIR VARIABLES Y ASIGNAR VALORES ###
e <- exp(1) # Asigna un valor a la variable (Base de los logaritmos
# naturales e = 2.718281)
# Imprime el valor que tenga la variable
x = 0.005 # También se puede usar el símbolo <- en lugar de igual
x0 = e ** (2*x) # Se le asigna una función a x0
tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
cat(tex, x0) # Se obtiene los resultados con la instrucción
cat, que concatena y convierte a string
# Otro ejemplo:
x0 = 1
```

Environment:

values	
a	6384671
b	0.5342198
e	2.71828182845905

Console:

```
> b
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e
[1] 2.718282
>
```


Asignamos a la variable `x` un valor (se puede usar `<-` en lugar de `=`).

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for variable assignment and printing. The code includes comments in Spanish and uses the `<-` operator for assignment.
- Console:** Shows the execution output of the code, including the assignment of `x = 0.005`.
- Environment:** A table listing the current values of variables in the Global Environment.

Source Editor Code:

```
56 # signif(x,n) redondea a x, con n dígitos significativos (default n=6)
57
58 signif(27.384956102)
59
60 signif(39.6429304521, 5)
61
62 signif(61.378045912, 2)
63
64 signif(316.6971243547, 3)
65
66 ### DEFINIR VARIABLES Y ASIGNAR VALORES ###
67
68 e <- exp(1) # Asigna un valor a la variable (Base de los logaritmos
69 naturales e = 2.718281)
70 e # Imprime el valor que tenga la variable
71
72 x = 0.005 # También se puede usar el símbolo <- en lugar de igual
73
74 x0 = e ** (2*x) # Se le asigna una función a x0
75
76 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
77
78 cat(tex, x0) # Se obtiene los resultados con la instrucción
79 cat, que concatena y convierte a string
80
81 # Otro ejemplo:
82 x0 = 1
```

Console Output:

```
R 4.3.1 ~ />
[1] 0.5342198
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> |
```

Environment Variables:

values	
a	6384671
b	0.5342198
e	2.71828182845905
x	0.005

Le asignamos a **x0** la función matemática que utilizará los valores guardados en **e** y **x**. Esta ecuación se traduce en $x0$ es igual a e potencia (2 por x).

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for rounding and defining variables. The line `x0 = e ** (2*x)` is highlighted in blue.
- Environment:** A table showing the values of variables in the Global Environment. The variable `x0` is highlighted with a red box.
- Console:** Shows the output of the R code executed in the source editor. The line `x0 = e ** (2*x)` is highlighted with a red box.

Environment Table:

Variable	Value
a	6384671
b	0.5342198
e	2.71828182845905
x	0.005
x0	1.01005016708417

Console Output:

```

R 4.3.1 ~/>
> 1/3.0
[1] 0.3333333
> round(54.2)
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
>
  
```

Aca se genera una variable llamada tex que contiene una cadena de texto.

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for variable assignment and function definition. Line 75 is highlighted: `tex = "El valor de x0 es: "`.
- Environment Pane:** Shows the current environment with variables `a`, `b`, `e`, `tex`, `x`, and `x0`. The variable `tex` is highlighted with a red box, showing its value as `"El valor de x0 es: "`.
- Console:** Shows the execution history. The last command entered is `tex = "El valor de x0 es: "`, which is highlighted with a red box.

```

70 x = 0.005 # Tambi?n se puede usar el simbolo <- en lugar de igual
71
72
73 x0 = e ** (2*x) # Se la asigna una funci?n a x0
74
75 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
76
77 cat(tex, x0) # Se obtiene los resultados con la instrucci?n
78 cat, que concatena y convierte a string
79
80 # Otro ejemplo:
81 x0 = 1
82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de rengl?n
88
89 ### DEFINIR FUNCIONES Y PASAR PAR?METROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la funci?n a la variable d,
92 con los par?metros a,b,c; y la funci?n b cuadrada menos 4 por a por c
93
94 d(2,2,1) # Se llama a la funci?n y se le dan los par
95 ?metros para el c?lculo y regresa el resultado
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

La intruccion **cat** contatena la cadena de texto guarda en tex y el valor guardado en la variable

x0.

The screenshot displays the RStudio interface. The main editor window shows R code with comments explaining the steps: assigning a value to x, calculating x0 as 2*x, creating a text string 'tex', and using the 'cat' function to concatenate 'tex' and 'x0'. The Environment pane on the right shows the current values of variables: 'a' (6384671), 'b' (0.5342198), 'e' (2.71828182845905), 'tex' ('El valor de x0 es: '), 'x' (0.005), and 'x0' (1.01005016708417). The Console pane at the bottom shows the execution of the code, with the final output of 'cat' being 'El valor de x0 es: 1.01005'.

```

70
71 x = 0.005 # Tambi?n se puede usar el simbolo <- en lugar de igual
72
73 x0 = e ** (2*x) # Se la asigna una funci?n a x0
74
75 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
76
77 cat(tex, x0) # Se obtine los resultados con la intrucci?n
78 cat, que concatena y convierte a string
79
80 # Otro ejemplo:
81 x0 = 1
82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # si usamos "\n" se cambia de rengl?n
88
89 ### DEFINIR FUNCIONES Y PASAR PAR?METROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la funci?n a la variable d,
92 con los par?metros a,b,c; y la funci?n b cuadrada menos 4 por a por c
93
94 d(2,2,1) # Se llama a la funci?n y se le dan los par
95 ?metros para el c?lculo y regresa el resultado
96
97
98
99
100

```

Environment pane (Values):

Variable	Value
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1.01005016708417

Console:

```

R 4.3.1 ~ /
[1] 54
> round(97.5684197, 2)
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
>

```

En esta parte estamos asignando a la variable x0 un valor, en este caso el numero 1.

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for variable assignment and function definition. The line `x0 = 1` is highlighted in blue.
- Environment:** A table showing the current environment's variables and their values. The variable `x1` is highlighted with a red box.
- Console:** Shows the output of the R commands. The command `x1 = x0 - pi * x0 + 1` is highlighted with a red box.

Source Editor Code:

```

70
71 x = 0.005 # Tambi?n se puede usar el s?mbolo <- en lugar de igual
72
73 x0 = e ** (2*x) # Se la asigna una funci?n a x0
74
75 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
76
77 cat(tex, x0) # Se obtiene los resultados con la instrucci?n
cat, que concatena y convierte a string
78
79 # Otro ejemplo:
80
81 x0 = 1
82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de rengl?n
88
89 ### DEFINIR FUNCIONES Y PASAR PAR?METROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la funci?n a la variable d,
con los par?metros a,b,c; y la funci?n b cuadrada menos 4 por a por c
92
93 d(2,2,1) # Se llama a la funci?n y se le dan los par
?metros para el c?lculo y regresa el resultado
94

```

Environment Table:

Variable	Value
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979

Console Output:

```

R 4.3.1 ~ /
[1] 97.57
> signif(27.384956102)
[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
>

```

En la variable x1 estamos guardando una ecuacion, la cual al ejecutarla nos dara el resultado.

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for variable assignment, string concatenation, and a function definition. The code is as follows:


```

70
71 x = 0.005 # Tambien se puede usar el simbolo <- en lugar de igual
72
73 x0 = e ** (2*x) # Se la asigna una funcion a x0
74
75 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
76
77 cat(tex, x0) # Se obtiene los resultados con la instruccion
78 cat, que concatena y convierte a string
79
80 # Otro ejemplo:
81 x0 = 1
82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # si usamos "\n" se cambia de renglon
88
89 ### DEFINIR FUNCIONES Y PASAR PARAMETROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la funcion a la variable d,
92 con los parametros a,b,c; y la funcion b cuadrada menos 4 por a por c
93
94 d(2,2,1) # Se llama a la funcion y se le dan los parametros para el calculo y regresa el resultado
95
96
97
98
99
      
```
- Environment:** A table showing the values of objects in the Global Environment. The 'x1' variable is highlighted with a red box, showing its value as -1.14159265358979.

Global Environment	
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979
- Console:** Shows the execution of the code. The final output for 'x1' is highlighted with a red box:


```

[1] 27.385
> signif(39.6429304521, 5)
[1] 39.643
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
> x1
[1] -1.141593
>
      
```

En esta ocasión nos dará los resultados de x_0 y x_1 , como usamos n se cambia de renglón y x_1 nos lo pondrá en otro renglón.

The screenshot shows the RStudio interface with a script editor, an environment pane, and a console.

Script Editor (Valores_numericos.R.txt):

```

70
71 x = 0.005 # También se puede usar el símbolo <- en lugar de igual
72
73 x0 = e ** (2*x) # Se le asigna una función a x0
74
75 tex = "El valor de x0 es: " # A la variable tex, se le asigna una cadena
76
77 cat(tex, x0) # Se obtiene los resultados con la instrucción
78 cat, que concatena y convierte a string
79
80 # Otro ejemplo:
81 x0 = 1
82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de renglón
88
89 ### DEFINIR FUNCIONES Y PASAR PARÁMETROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la función a la variable d,
92 con los parámetros a,b,c; y la función b cuadrada menos 4 por a por c
93
94 d(2,2,1) # Se llama a la función y se le dan los parámetros para el cálculo y regresa el resultado
95
96

```

Environment Pane:

values	
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979

Console:

```

R43.1 ~ /
> signif(61.378045912, 2)
[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
> x1
[1] -1.141593
> cat("x0 =", x0, "\n", "x1 =", x1)
x0 = 1
x1 = -1.141593
>

```


En esta parte a la variable d le asignamos los parametros a, b, c y la funcion.

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code. Lines 91-93 define a function `d` and call it with arguments `2, 2, 1`.


```

82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de renglón
88
89 ### DEFINIR FUNCIONES Y PASAR PARÁMETROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la función a la variable d,
92 con los parámetros a,b,c; y la función b cuadrada menos 4 por a por c
93 d(2,2,1) # Se llama a la función y se le dan los parámetros para el cálculo y regresa el resultado
94
95
96 ### GRAFICACIÓN DE FUNCIONES ###
97
98 g = function(x) sin(cos(x)*exp(-x/2))
99 plot(g, -8, -5, # Rango
100      lwd = 1, # Grosor
101      main = "Gráfico de g", # Título del gráfico
102      col = "red", # Color de la línea
103      xlab = "x", # Etiqueta de x
104      ylab = "g(x)", # Etiqueta de y
105      axes = TRUE, # Ejes x,y visibles
106      n = 1000) # Número de puntos
107
91:1
      
```
- Environment:** Shows the current state of the workspace.

values	
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979

Functions	
d	function (a, b, c)
- Console:** Shows the execution output. The last command is `d = function(a,b,c) b^2-4*a*c`, which is highlighted with a red box.


```

[1] 61
> signif(316.6971243547, 3)
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
> x1
[1] -1.141593
> cat("x0 =", x0, "\n", "x1 =", x1)
x0 = 1
x1 = -1.141593
> d = function(a,b,c) b^2-4*a*c
> |
      
```

En esta parte se llama a la función pero se le dan los valores a los parametros a, b y c. Nos regresa el resultado.

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for defining functions and plotting. The function `d` is defined as `d = function(a,b,c) b^2-4*a*c`. The function `d(2,2,1)` is called, and the result `-4` is displayed in the console.
- Environment:** Shows the global environment with variables `a`, `b`, `e`, `tex`, `x`, `x0`, and `x1`. The function `d` is also listed under the 'Functions' section.
- Console:** Shows the execution of the code. The output of `d(2,2,1)` is `[1] -4`.

```

82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de renglón
88
89 ### DEFINIR FUNCIONES Y PASAR PARÁMETROS
90 d = function(a,b,c) b^2-4*a*c # se asigna la función a la variable d,
91 con los parámetros a,b,c; y la función b cuadrada menos 4 por a por c
92
93 d(2,2,1) # Se llama a la función y se le dan los parámetros para el cálculo y regresa el resultado
94
95
96 ### GRAFICACIÓN DE FUNCIONES ###
97
98 g = function(x) sin(cos(x)*exp(-x/2))
99 plot(g, -8, -5, # Rango
100      lwd = 1, # Grosor
101      main = "Gráfico de g", # Título del gráfico
102      col = "red", # Color de la línea
103      xlab = "x", # Etiqueta de x
104      ylab = "g(x)", # Etiqueta de y
105      axes = TRUE, # Ejes x,y visibles
106      n = 1000) # Número de puntos
107
93:1

```

Environment:

Variable	Value
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979

Functions:

Function Name	Definition
d	function (a, b, c)

Console:

```

R 4.3.1 ~ /
[1] 317
> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
> x1
[1] -1.141593
> cat("x0 =", x0, "\n", "x1 =", x1)
x0 = 1
x1 = -1.141593
> d(2,2,1)
[1] -4
>

```


A la variable g se le asigna una funcion (function (x)) con sus valores.

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for defining variables and functions. The function `g` is defined as `g = function(x) sin(cos(x))*exp(-x/2)` on line 98.
- Console:** Shows the execution of the code. The final command entered is `> g = function(x) sin(cos(x))*exp(-x/2)`, which is highlighted with a red box.
- Environment:** Lists the objects in the Global Environment. The variable `g` is listed with the value `function (x)`, highlighted with a red box.

```

82
83 x1 = x0 - pi * x0 + 1
84
85 x1
86
87 cat("x0 =", x0, "\n", "x1 =", x1) # Si usamos "\n" se cambia de renglón
88
89 ### DEFINIR FUNCIONES Y PASAR PARÁMETROS
90
91 d = function(a,b,c) b^2-4*a*c # se asigna la función a la variable d,
92 con los parámetros a,b,c; y la función b cuadrada menos 4 por a por c
93
94 d(2,2,1) # Se llama a la función y se le dan los parámetros para el cálculo y regresa el resultado
95
96 ### GRAFICACIÓN DE FUNCIONES ###
97
98 g = function(x) sin(cos(x))*exp(-x/2)
99 plot(g, -8, -5, # Rango
100      lwd = 1, # Grosor
101      main = "Gráfico de g", # Título del gráfico
102      col = "red", # Color de la línea
103      xlab = "x", # Etiqueta de x
104      ylab = "g(x)", # Etiqueta de y
105      axes = TRUE, # Ejes x,y visibles
106      n = 1000) # Número de puntos
107
98:1
  
```

```

> e <- exp(1)
> e
[1] 2.718282
> x = 0.005
> x0 = e ** (2*x)
> tex = "El valor de x0 es: "
> cat(tex, x0)
El valor de x0 es: 1.01005
> x0 = 1
> x1 = x0 - pi * x0 + 1
> x1
[1] -1.141593
> cat("x0 =", x0, "\n", "x1 =", x1)
x0 = 1
x1 = -1.141593
> d = function(a,b,c) b^2-4*a*c
> d(2,2,1)
[1] 0
> g = function(x) sin(cos(x))*exp(-x/2)
>
  
```

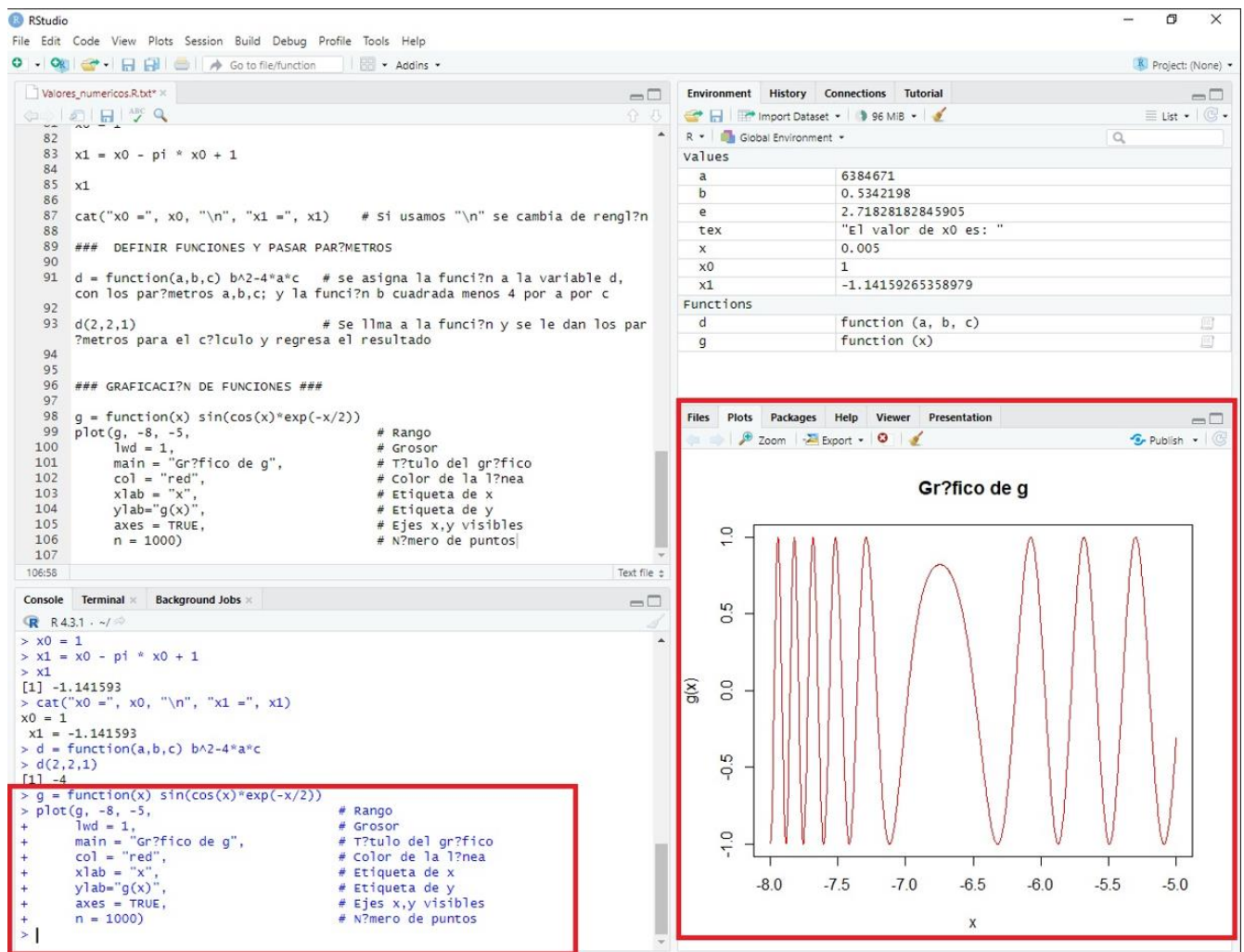
Environment:

values	
a	6384671
b	0.5342198
e	2.71828182845905
tex	"El valor de x0 es: "
x	0.005
x0	1
x1	-1.14159265358979

Functions:

d	function (a, b, c)
g	function (x)

Se le asignan valores y se grafica la funcion guardada en la variable g.



Conclusión

El entendimiento y uso de métodos numéricos en ingeniería como en el mundo real es de una utilidad excepcional, estos métodos son una sucesión de operaciones matemáticas utilizadas para encontrar una solución numérica aproximada, es decir se trata de una serie de cálculos para acercarnos lo más posible a una solución numérica razonablemente buena. Son especialmente útiles para facilitar la resolución de problemas que conllevan una enorme cantidad de cálculos, lo que nos permite ahorrar tiempo.

Referencias

Vasquez, I. R. S. (n.d.). *METODOS NUMERICOS PARA INGENIERIA*. Edu.Co. Retrieved October 23, 2023, from <https://disi.unal.edu.co/~lctorress/MetNum/LiMetNu2.pdf>

Noguera, I. B. (2020, October 29). ¿Qué son los métodos numéricos? *Ingeniería Química Reviews*. <https://www.ingenieriaquimicareviews.com/2020/10/metodos-numericos.html>