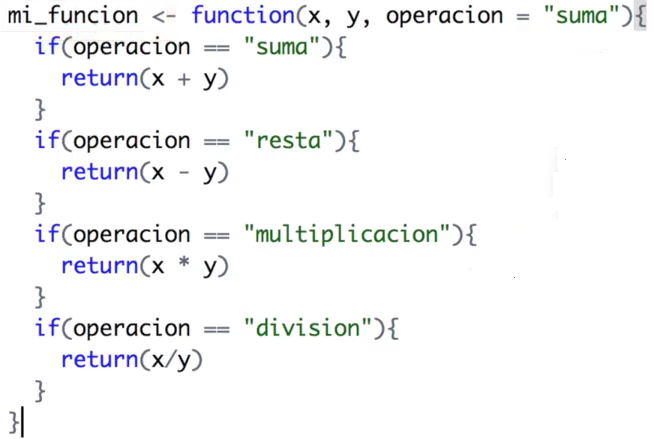
**Semana 2**

**09. Creando tu primera función en R**

En esta lección vamos a aprender a crear funciones en R, aprendiendo los pasos hasta poder crearlas. La creación de funciones en R es uno de los pilares de la programación y el análisis de datos, nos facilita la organización de nuestro código y de nuestros análisis. Podríamos definir una función desde la consola, pero lo ideal es usar el editor, que nos permite pasar las funciones a un archivo de texto y luego recuperarlas usando la función *source()*. Una vez que creas una función se puede guardar en un archivo y luego crear un paquete, organizando el ambiente en un lugar.

El primer paso es abrir RStudio y crear un nuevo script de R dentro de la sección *New file*. El siguiente paso es la especificación de la función, es decir, definir qué hará la función. En nuestro caso vamos a crear una función que reciba dos números y lleve a cabo la operación que queramos. El siguiente apso es escribir la función y una vez que esté terminada la seleccionamos con el ratón y la ejecutamos en el botón de *Run.* Estamos viendo en pantalla la definición escrita.



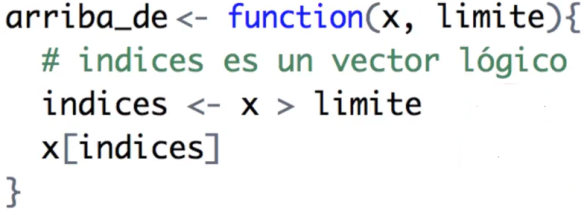
La operación es una cadena en la que se indica qué es lo que se va a llevar a cabo. Por tanto, para llamar la función, podemos usar una gran variedad de situaciones (si no se indica la operación el programa supone que se está pidiendo la operación por omisión, que es la suma):

*>mi\_funcion(2,2)*

>*mi­­\_function(2,2,”suma”)*

>*mi\_funcion(2,2,”multiplicacion”)*

Ahora vamos a escribir una función en la que al pasarle un vector regrese un subconjunto de dicho vector inferior al valor que indiquemos.



Donde limite es el límite inferior que vamos a usar. En resumen, la especificación de una función es la descripción formal de las capacidades de la función y la interacción con el usuario.

Una vez que nuestra función está dentro del ambiente de nuestra consola, podemos observar que al escribir la función, al usar la tecla Tab dentro del paréntesis sale una lista con todos los parámetros que tenemos.

Podemos observar que al ejecutar la función, cuando solo hay un parámetro sale error, y esto se debe a que todos los parámetros deben estar presentes.

* **Tarea en RStudio**

A continuación, voy a realizar la única tarea que se pide en la semana 2. Hay que destacar que antes de trabajar con un archivo que tenga símbolos especiales (como la ñ o las tildes), hay que usar el parámetro *fileEncoding=TRUE o fileEnconding=”latin1”*. Hay que introducir el comando *View()*.

La función *class()* te permite guardar los tipos de datos en una variable aparte. Luego podemos indicar que son las etiquetas con el parámetro colClasses. La función *save()* escribe una representación externa de los objetos especificados, es decir, la guarda. En R, las rutas de archivo siempre son especificadas con diagonales “/”.

Si deseas escribir una función que acepte un número variable de argumentos, en R puedes usar “…”. Para hacer esto se especifica “…” en los argumentos de la función, normalmente en el último lugar. Esto sin embargo no ocurre para la función *paste(),* que al juntarse con el argumento *sep=””* separa los elementos concatenados. La función *toupper()* transforma en mayúsculas.

La función que te epermite cambiar las letras por números se hace de la siguiente forma:

*numeros\_por\_vocales <- function(...){*

*chartr("aeio", "4310", tolower(paste(...)))*

*}*

La función *operador\_binario* recibe una función con los elementos especificados:

*operador\_binario <- function(fun, a, b){*

*fun(a, b)*

*}*