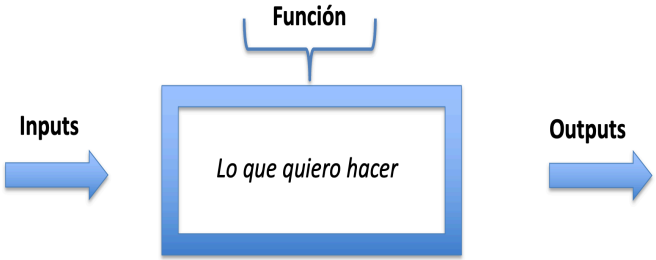
Clase 4

1. Funciones
2. if else
3. Iteraciones
4. Live coding
5. Introducción al paquete survey
6. Live coding

Funciones

Función

Una **función** es algún procedimiento que queremos aplicar a algún objecto para obtener algún resultado



Hasta ahora nosotros ya hemos estado usando algunas funciones en R, por ejemplo:

- `mean(x)` : La función mean, toma el vector x y nos devuelve la media
- `length(x)` : La función length, toma el vector x y nos devuelve su tamaño
- `select(dataframe, columnas)` : La función select, toma un dataframe y nombres de columnas y nos devuelve un nuevo dataframe con las columnas seleccionadas

Estas funciones fueron diseñadas para automatizar tareas comunes de una manera más potente y general.

Cuándo escribir una función

Una de las principales razones para considerar escribir una función es cuando hemos copiado y pegado un bloque de código más de dos veces.
Escribir una función tiene grandes ventajas:

- Podemos darle a un nombre a nuestra función para que el código sea más fácil de entender.
- Si los inputs de nuestra función cambian, solo necesitamos actualizar el código una sola vez, en lugar de muchas veces.
- Elimina la posibilidad de cometer errores al copiar y pegar (es decir, actualizar el nombre de una variable en un lugar, pero no en otro).

Crear una nueva función

1. Elegir un nombre para la función. Tip: elegir un nombre que aluda al proceso que queremos realizar
2. Enumerar las entradas, o argumentos, a la función dentro de la función. Ejemplo: function(x,y,z)
3. Colocar el código que has desarrollado en el cuerpo de la función, dentro del bloque que sigue inmediatamente a la función.

```
miFuncion <- function(inputs) {

  output = # Alguna operacion/proceso

    return(output)

}
```

Tips

- Es más fácil comenzar con un código ya trabajado y convertirlo en una función; es más difícil crear una función y luego intentar que funcione.
- Siempre debemos verificar nuestra función con algunas entradas diferentes

if else

if else

Una **if else** *statement* nos permite ejecutar código de acuerdo a ciertas condiciones. En general usamos:

```
if (condicion) {
  # codigo ejecutado cuando la condicion regresa TRUE
} else {
  # codigo ejecutado cuando la condicion regresa FALSE
}
```

Condición

IMPORTANTE La condición debe devolver o TRUE o FALSE, no un vector de TRUE/FALSE ni un elemento vacío

Iteraciones

Otra herramienta que nos ayuda para evitar escribir el mismo código muchas veces son las **iteraciones**.
Iterar significa repetir la misma operación en múltiples inputs, por ejemplo aplicar una función en cada entrada de un vector o en diferentes columnas o en diferentes conjuntos de datos.

for loops

Un for loop es una herramienta para hacer iteraciones. En general usamos:

```
for (elemento in secuencia) {

   # alguna operacion

}
```

En general usamos for loops cuando queremos:

- Modificar un objeto existente, en lugar de crear un nuevo objeto.
- Iterar a lo largo de algun objeto y aplicar una operación

Live coding

Introducción al paquete survey

Cuando nuestros datos tienen un **diseño muestral complejo**, es decir, que fueron obtenidos mediante un procedimiento aleatorio para seleccionar una muestra a partir de una población estructurada con estratos y/o conglomerados, no podemos usar estadísticas clásicas.

El paquete **survey** nos permite estimar estadisticas tomando en cuenta diseños muestrales complejos

Mini tutorial

1. Instalar el paquete:
   install.packages("survey")
2. Cargar el paquete:
   library(survey)
3. Identificar las variables necesarias para definir nuestro diseño muestral con la función *svydesign()*, que tiene los siguientes inputs:
   - id = Fórmula para indicar las variables que definen los conglomerados
   - strata: Fórmula para definir los posibles estratos
   - weights: Fórmula para definir los pesos
   - data: Dataframe con los datos muestrales

Nota: Dependiendo nuestros tipo de datos, podríamos necesitar más variables muestrales en nuestro diseño.

Para indicar una **fórmula**, usamos el símbolo˜.

Ejemplo: Supongamos que tenemos una base que se llama MisDatos que tiene las variables muestrales: identificador, estrata y pesos. Definiríamos nuestro diseño de encuesta así:

```
svydesign(id = ~identificador, strata = ~estrata,
          weights = ~pesos, data = MisDatos)
```

Paso 4. Definir que estadística queremos calcular. El paquete survey tiene distintas funciones como por ejemplo:

- svymean
- svyvar
- svyquantile
- svytotal

Ejemplo: Estimar la media de la variable kg:

```
# Guardo mi diseño
Design <- svydesign(id = ~identificador, strata = ~estrata,
          weights = ~pesos, data = MisDatos)

# Llamo a la funcion svymean
svymean(~kg, design = Design)
```

Live coding

Referencias y recursos

- P. Kuhnert & B. Venables, An Introduction to R: Software for Statistical Modeling & Computing
- Grolemund, G., & Wickham, H. (2017). R for Data Science. O'Reilly Media. *https://r4ds.had.co.nz/*
- Lumley T., survey package in R *https://cran.r-project.org/web/packages/survey/survey.pdf*