Fundamentos R –3

Las herramientas del científico de datos

Juan Manuel Moreno — jmmoreno@profesores.imf.com

IMF
Smart Education

ÍNDICE

1. Objetivos unidad 4
2. Estructuras de control
3. Funciones

IMF
Smart Education

01 Objetivos unidad 4

IMF
Smart Education

1.– Objetivos Unidad 4

IMF Smart Education

• Conocer qué es el lenguaje de programación R y habituarse al IDE de programación RStudio.

• Aprender sobre la sintaxis básica de R, desde la creación de scripts y notebooks en donde el alumno podrá realizar anotaciones en Markdown, realizar comentarios, conocer los principales operadores, hasta, implementar variables y conocer los diferentes tipos de variables que existen en R.

• Distinguir y saber utilizar las siguientes estructuras de datos en R: Vectores, arrays, factores, listas, matrices y dataframes.

• Saber controlar el flujo de un programa a través de sentencias condicionales y bucles.

• Desarrollar funciones propias, diferenciando entre parámetros de entrada y salida, así como utilizar funciones propias que integra R internamente.

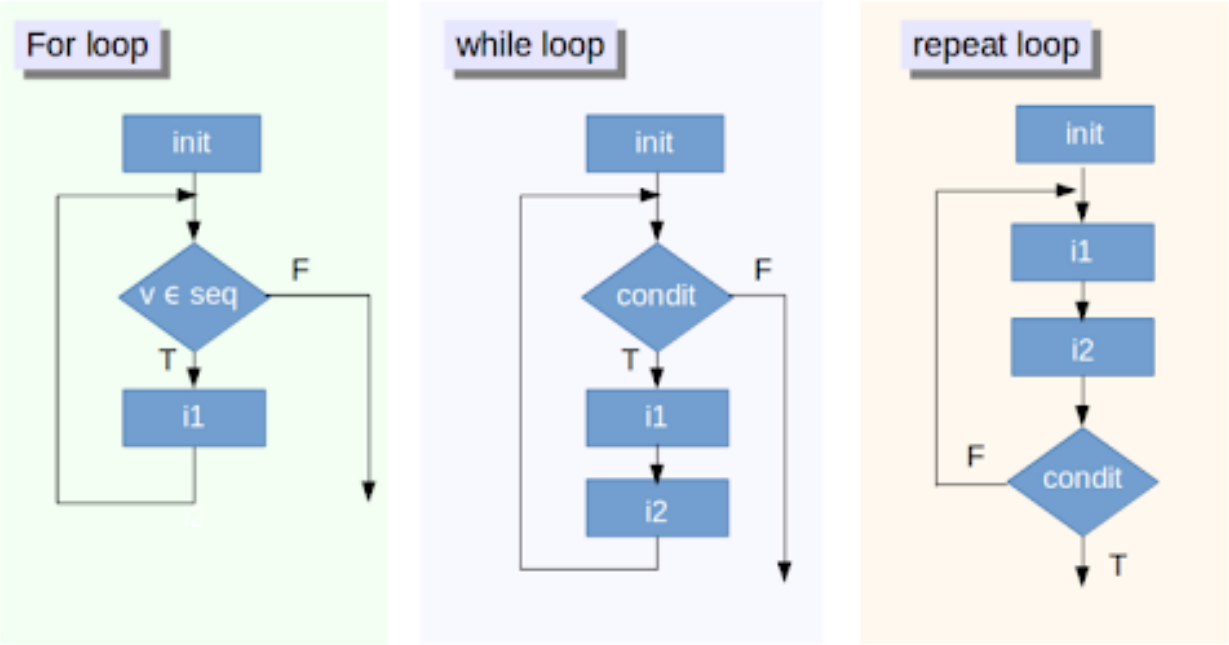
02

Estructuras de control

IMF
Smart Education

2.1.– Estructuras de control

- Las estructuras de control en R, pueden dividirse en los siguientes bloques.
  - **Condicionales** (`if – else`)
  - **Bucles** (`for`, `while` y `repeat`)



2.2.– Condicionales If

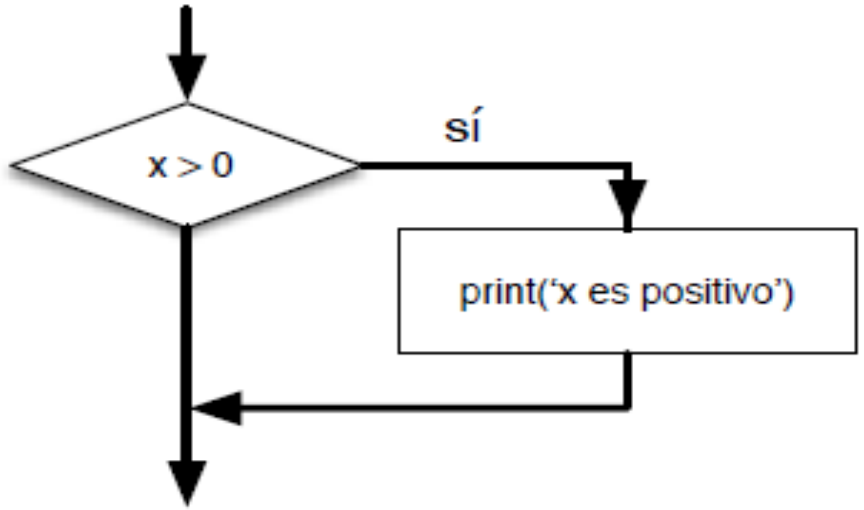
- Se utilizan para controlar la ejecución de un programa a través de expresiones booleanas (True, False).
- En R distinguiremos tres tipos de condicionales
    - Condicional **simple**
    - Condicional **anidado**.
    - Condicional **encadenado**.

2.3.– Condicionales If – simple

- Es el tipo más básico se estructura como
- **`if`** (expresión booleana se cumple)
    - Realizar una acción
- **`else`** (si no).
    - Realizar otra acción
- No tiene porqué aparecer obligatoriamente el bloque else.
- La sintaxis de un condicional en R es:

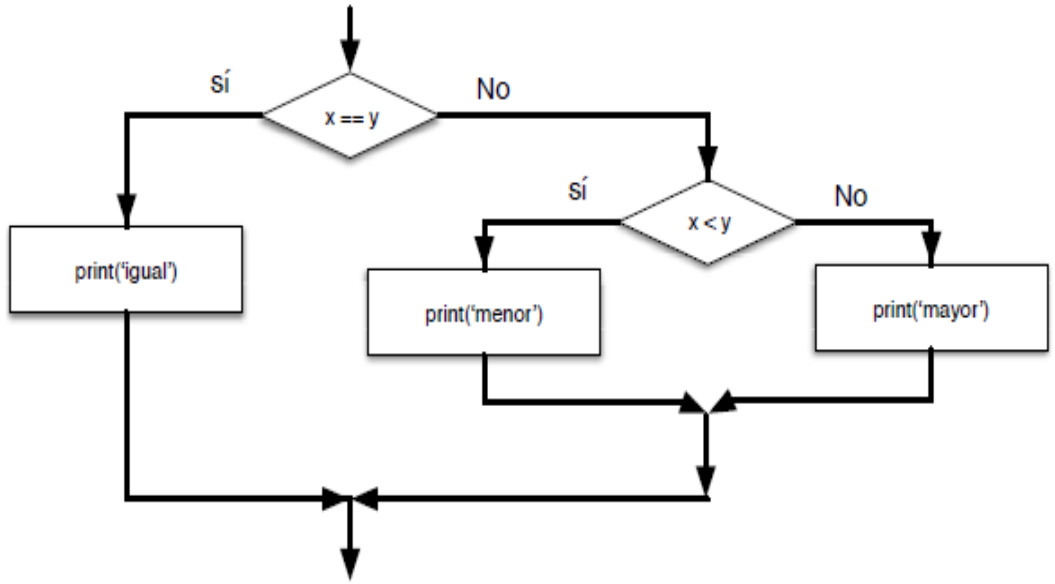
```
if (condición booleana) {
…
} else {

}
```



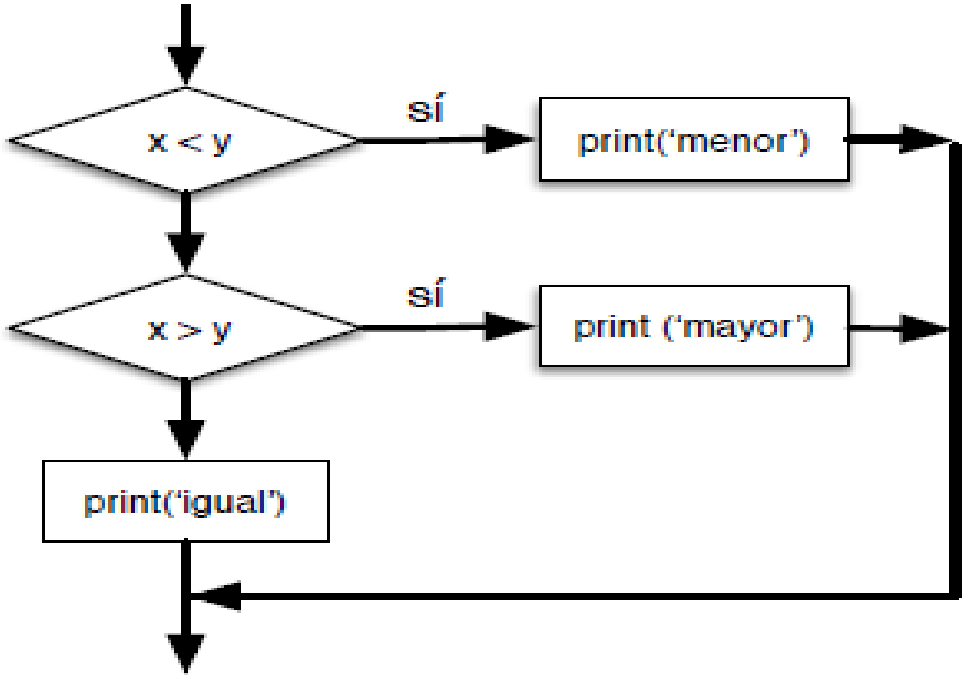
2.4.– Condicionales If – anidado

- Puede entenderse como tener un condicional dentro de un condicional. Por ejemplo.
- **if** (condición booleana){
    - **if** (condición booleana) {
        - Realizar una acción
    - } **else** {
        - Realizar otra acción }
- **else** (si no).
    - Realizar otra acción



2.5.– Condicionales If – Encadenado

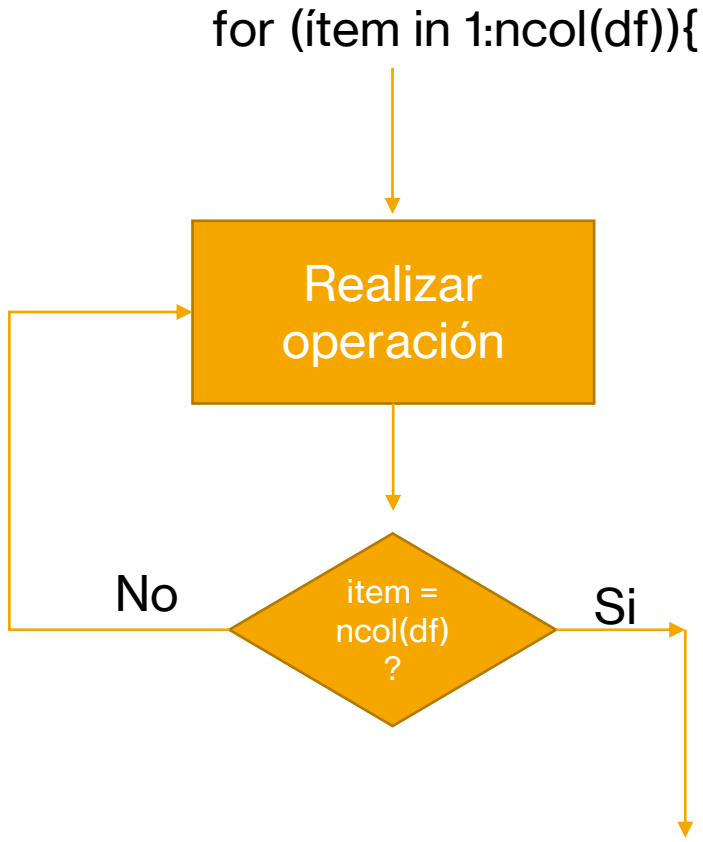
- Al término de un condicional (simple o anidado), aparece otro condicional (simple o anidad). Por ejemplo:
- `if (condición booleana){`
    - `Realizar una acción`
- `} else if (condición boolena) {`
    - `Realizar una acción`
- `} else {`
    - `Realizar otra acción }`



2.6.– Bucles – For

- Realiza de forma secuencial una misma instrucción, hasta llegar al fin de la secuencia.
- La sintaxis de un bucle **`for`** es.

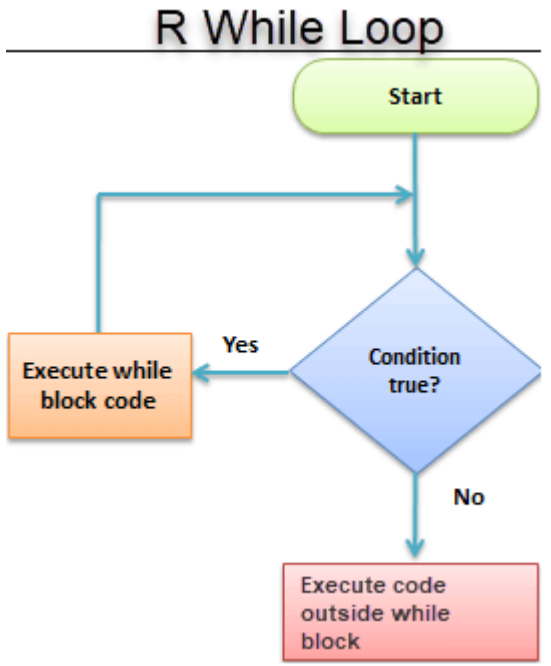
```
for (variable in secuencia) {
….
}
```



2.7.- Bucles – While

- Se realiza una misma acción hasta que se cumpla una instrucción de parada, dada por una condición booleana.
- La sintaxis de un bucle **while** en R es:

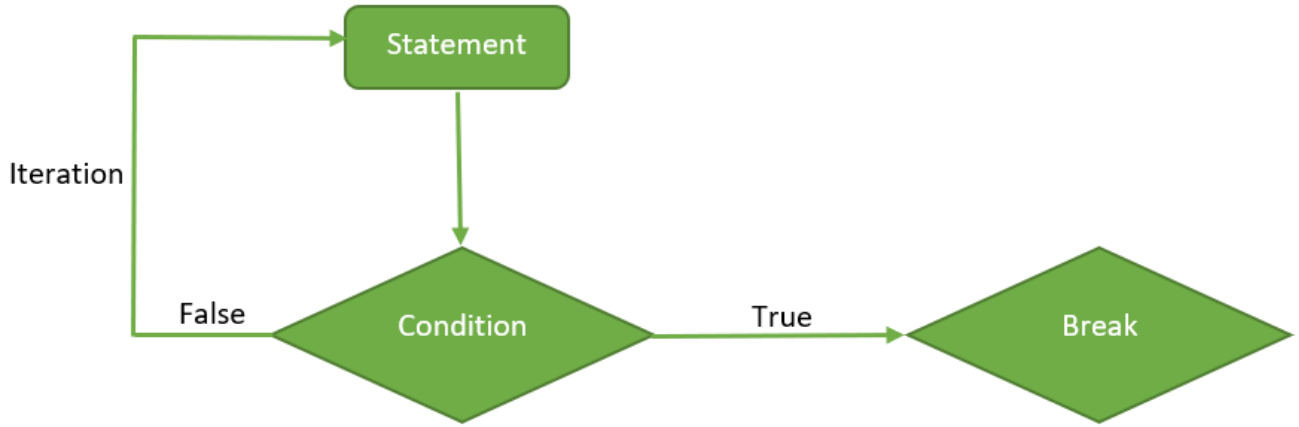
```
while (condición booleana) {
 …
 ¿se cumple ahora la condición? – Instrucción de parada
 }
```



2.8.– Bucles – Repeat

- Se parte desde el valor de una variable, se realiza de forma repetitiva la misma operación hasta que el resultado cambie, por lo tanto, dentro del bucle **`repeat`** hay una instrucción condicional.
- Para salir de un bucle **`repeat`**, hay que emplear una condición para romper el ciclo, **`break`**.
- La sintaxis de un bucle **`repeat`** es:

```
repeat
{
 ...
if( condicion booleana )
   {
     break
  }
}
```



03

Funciones

IMF
Smart Education

3.– Funciones

3.1.– Funciones

• Para aislar cualquier parte del código, hacerlo modular y reproducible se emplean funciones.

• Una función en R se denota con la palabra reservada **`function`**, a la cuál se le pasan los parámetros de entrada (o no).

• A través de la sentencia **`return`**, se devuelve el parámetro de salida.

• Para devolver más de un parámetro de salida, lo más aconsejable es emplear una lista para devolver los n parámetros.

• La estructura de una función en R es:

```
def nombre.funcion(parametro1, parametro2, …. , parametro n) {
….
return (variable de salida)
}
```

3.– Funciones

3.1.– Funciones – Apply

• Existen una serie de funciones, que forman parte de la familia de funciones **`apply`**, que sirven para aplicar a una estructura de datos, ya sea por sus filas o columnas una misma función.

- **`apply`**: Realiza la misma operación sobre un eje (1 filas, 2 columnas)
- **`lapply`**: Mismo funcionamiento que apply pero, optimizado para listas, el resultado lo devuelve en forma de lista.
- **`sapply`**: Recibe una lista y, devuelve un vector.
- **`tapply`**: Realiza una operación sobre un vector en función de un vector de categorías, muy recomendado en dataframes.
- **`mapply`**: Opera entre matrices o vectores, devuelve el resultado en forma de vector o, de lista si devuelve más de un resultado.

Seguimiento práctico del contenido

A partir de aquí, veremos las diferentes estructuras de control y arrays mediante los notebooks.

4_5_Herramientas_de_control.RMD
4_6_Funciones.RMD

IMF

Smart Education