Semana 8 y 9

Contents

1 Operadores y Estructuras de control (Sem 8 y 9)

Nos permite controlar la manera en cómo se ejecuta nuestro código (establecen **condicionales**).

Las estructuras de control más usadas en R son las siguiente:

Estructura de control	Descripción
`if` , `else`	Si, de otro modo
`for`	Para cada uno en
`while`	Mientras
`break`	Interrupción
`next`	Siguiente

1.1 if, else

if se usa cuando deseamos que una operación se ejecute cuando cumple una condición deseada (Ej. **Si** esta condición es cierta, **entonces** haz una operación específica)

else se usa para indicarle a R qué hace en caso de que la condición de un *if* no se cumpla.

La estructura de un la función *if* es la siguiente

```
#if (condicion) {
#  proceso_si_se_cumple_condicion
#}
```

Si la condición es verdadera, entonces se realiza la operación. En caso contrario, no se realiza la operación. Veamos algunos ejemplos:

```r
#Se cumple la condición y se muestra "verdadero".
if (19 > 15) {
  "verdadero"
}
```

```
## [1] "verdadero"
```

```r
#Si no se cumple la condición, el código no se ejecuta.
if (10 > 15) {
  "verdadero"
}
```

la función *else* complementa a *if*, es decir, le asigna un proceso siempre en cuando la condición inicial no se cumpla. En otras palabras, un *if* con *else* es la manera de decirle a R:

- **SI** esta condición es cierta, **ENTONCES** haz estas operaciones.**DE OTRO MODO** haz estas otras operaciones.

La estructura de in **if** con **else** es el siguiente:

```r
#if (condition) {
#  proceso1
#} else {
#  proceso2
#}
```

Del ejemplo anterior podemos hacer las siguientes modificaciones:

```r
#Se cumple condición
if (18 > 15) {
  "Verdadero"
} else {
  "Falso"
}
```

```
## [1] "Verdadero"
```

```r
#No se cumple condición
if (10 > 15) {
  "Verdadero"
} else {
  "Falso"
}
```

```
## [1] "Falso"
```

Para ilustrar el uso de *if* y *else* definiremos una función que calcule el promedio de calificaciones de un estudiante y, dependiendo de la calificación calculada, nos devuelva un mensaje específico.

```r
promedio <-
  function(nota){
    mean(nota)
  }

promedio(c(5,8,9,6,5))
```

```
## [1] 6.6
```

Ahora realizaremos un proceso de tal forma que la función nos muestre si un estudiante ha aprobado o no. Si asumimos que un estudiante necesita obtener 5 o más en promedio par aprobar podemos decir:

- **SI** el promedio de un estudiante es igual o mayor a 5, **ENTONCES** mostrar "Aprobado", **DE OTRO MODO**, mostrar "Reprobado.

```r
promedio <-
  function(notas) {
    media <- mean(notas)

    if (media >= 5) {
      print("Aprobado")
    } else{
      print("Desaprobado")
    }
  }
```

```r
promedio(c(6,4,5,7,5,6))
```

```
## [1] "Aprobado"
```

```r
promedio(c(3,4,4,4,5,3))
```

```
## [1] "Desaprobado"
```

1.2 If else

La función *ifelse()* permite vectorizar if, else. En lugar de escribir una línea de código para cada comparación, podemos usar una sola llamada a esta función, que se aplicará a todos los elementos de un vector.

Si intentamos usar if else con un vector, se nos mostrará una advertencia:

```r
if (1:10 < 3) {
  "Verdadero"
}
```

```
## [1] "Verdadero"
```

El mensaje muestra que la condición sólo es evaluada para el primer elemento del vector 1:10, es decir para 1. Los demás elementos son ignorados.

Por el contrario, con *ifelse* le indicamos que la condición debe de cumplirse para cada uno de los elementos del vector. Esta función tiene tres argumentos:

- un vector
- un valor que indique qué mostrar si la condición se cumple
- un valor que indique qué mostrar si la condición no se cumple

```
vector <- 1:10
ifelse(vector > 5, "TRUE", "FALSE")
```

```
##  [1] "FALSE" "FALSE" "FALSE" "FALSE" "FALSE" "TRUE"  "TRUE"  "TRUE"  "TRUE"
## [10] "TRUE"
```

```
edad <- c(10,15,17,19, 20, 13, 11, 16, 15)

ifelse(edad >= 15, "TRUE", "FALSE")
```

```
## [1] "FALSE" "TRUE"  "TRUE"  "TRUE"  "TRUE"  "FALSE" "FALSE" "TRUE"  "TRUE"
```

```
#Determinar cual de los números del 1 al 60 son pares y cuales impares ¿que se debe cumplir?

numeros <- 1:60
ifelse(numeros %% 2 == 0, "par", "impar")
```

```
##  [1] "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar"
## [10] "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"
## [19] "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar"
## [28] "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"
## [37] "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar"
## [46] "par"   "impar" "par"   "impar" "par"   "impar" "par"   "impar" "par"
## [55] "impar" "par"   "impar" "par"   "impar" "par"
```

Otro ejemplo más complejo. Solicitamos sólo los números que son exactamente divisibles entre 2 y 3.

```
numeros <- 1:20

ifelse(numeros %% 2 == 0 & numeros %% 3 == 0,
       "Divisible",
       "No divisible")
```

```
##  [1] "No divisible" "No divisible" "No divisible" "No divisible" "No divisible"
##  [6] "Divisible"    "No divisible" "No divisible" "No divisible" "No divisible"
## [11] "No divisible" "Divisible"    "No divisible" "No divisible" "No divisible"
## [16] "No divisible" "No divisible" "Divisible"    "No divisible" "No divisible"
```

Conocer esta función puede ser particularmente útil para recodificar datos. Por ejemplo:

```
sexo <- c(0, 1, 0, 0, 0, 0, 0, 1, 1, 0)
sexo <- ifelse(sexo == 0, "hembra", "macho")
sexo
```

```
##  [1] "hembra" "macho"  "hembra" "hembra" "hembra" "hembra" "hembra" "macho"
##  [9] "macho"  "hembra"
```

1.3 Uso de condicionales y estructuras de control en un dataframe

Recordemos que para hacer uso de las condicionales podemos hacer uso de las funciones *if*, *else*, *if else* y/o uso de indexaciones combinado con operadores lógicos y/o relacionales. (https://github.com/AngeloCris/computacionCientifica_2021/blob/master/semana7_8/Semana7-y-8.md).

```
datosCrangrejo <-
  read.table("datos/datos_cangrejos.txt",
             sep = ",",
             header = TRUE)
datosCrangrejo
```

Para determinar el total de filas de mi data frame usamos la función *length*.

```
length(datosCrangrejo$Year)
```

```
## [1] 71013
```

La base de datos muestra los registros mensuales de la longitud y sexo de un cangrejo para el periodo 2008 y 2014. En función a lo mencionado anteriormente, resolvamos lo siguiente.

- Crear una columna denominada "sexo" que reemplace las categorías de la columna "sex". Considere lo siguiente
  - "F" = "female"
  - "0" = "ovígera"

```
#Uso del ifelse
#Para condicionales vectorizados
datosCrangrejo$sexo <-
  ifelse(datosCrangrejo$Sex == "F", "Females", "Ovigera")
```

- Crear una columna denominada "estacion" que reemplace las categorías de la columna "Month". Considere lo siguiente:
  - verano = 12, 1, 2
  - otoño = 3, 4, 5
  - invierno = 6, 7, 8
  - primavera = 9, 10, 11

```
datosCrangrejo$estacion <- "NA"
datosCrangrejo$estacion[datosCrangrejo$Month %in% c(12,1,2)] <- "verano"
datosCrangrejo$estacion[datosCrangrejo$Month %in% c(3,4,5)] <- "otoño"
datosCrangrejo$estacion[datosCrangrejo$Month %in% c(6,7,8)] <- "invierno"
datosCrangrejo$estacion[datosCrangrejo$Month %in% c(9,10,11)] <- "primavera"
```

- Determinar las posiciones de la categoría "primavera" y luego determinar los valores que se asocian a esas posiciones.

```r
#Deseamos extraer las posiciones de las categorías que pertenecen a la estación primavera.
which(datosCrangrejo$estacion == "primavera")

#Extrayendo los valores de las variables de esta categoría
datosCrangrejo[which(datosCrangrejo$estacion == "primavera"),]
```

- Extraer todos los registros de cangrejos que superan la talla media.

```r
media <- mean(datosCrangrejo$C_Length, na.rm = T)
head(which(datosCrangrejo$C_Length > media))
```

```
## [1] 1 2 3 4 5 6
```

```r
head(datosCrangrejo[which(datosCrangrejo$C_Length > media),])
```

```
##   Year   Sector Month Sex C_Length    sexo  estacion
## 1 2013 Sector_1    11   F   123.64 Females primavera
## 2 2013 Sector_1    11   F   118.47 Females primavera
## 3 2013 Sector_1    11   F   115.53 Females primavera
## 4 2013 Sector_1    11   F   113.98 Females primavera
## 5 2013 Sector_1    11   F   108.75 Females primavera
## 6 2013 Sector_1    11   F   106.12 Females primavera
```

```r
head(datosCrangrejo[datosCrangrejo$C_Length > media,])
```

```
##   Year   Sector Month Sex C_Length    sexo  estacion
## 1 2013 Sector_1    11   F   123.64 Females primavera
## 2 2013 Sector_1    11   F   118.47 Females primavera
## 3 2013 Sector_1    11   F   115.53 Females primavera
## 4 2013 Sector_1    11   F   113.98 Females primavera
## 5 2013 Sector_1    11   F   108.75 Females primavera
## 6 2013 Sector_1    11   F   106.12 Females primavera
```

- Extraer todos los registros de cangrejos hembras que superan la talla media.

```r
datosCrangrejo$C_Length > media
which(datosCrangrejo$C_Length > media & datosCrangrejo$sexo == "Females")

head(datosCrangrejo[which(datosCrangrejo$C_Length > media & datosCrangrejo$sexo == "Females"),])
```

- Asumamos que la mínima talla de extracción de la especie de cangrejo sea 110 mm. Crear una columna que identifique si la longitud registrada supera o no la talla mínima.

```r
tmin <- 110
datosCrangrejo$lengthMin <- "NA"
datosCrangrejo$lengthMin <- ifelse(datosCrangrejo$C_Length > 110, ">tmin", "<tmin")
head(datosCrangrejo)
```

```
##   Year   Sector Month Sex C_Length    sexo  estacion lengthMin
## 1 2013 Sector_1    11   F   123.64 Females primavera     >tmin
## 2 2013 Sector_1    11   F   118.47 Females primavera     >tmin
## 3 2013 Sector_1    11   F   115.53 Females primavera     >tmin
## 4 2013 Sector_1    11   F   113.98 Females primavera     >tmin
## 5 2013 Sector_1    11   F   108.75 Females primavera     <tmin
## 6 2013 Sector_1    11   F   106.12 Females primavera     <tmin
```

1.4 for

1.5 while