Diplomado en Ciencia de Datos Introducción a RStudio

Estructuras de control

En la mayoría de lenguajes de programación, que soportan el paradigma imperativo, existen tres estructuras de control básicas:

- Secuencia
- Selección (o decisión)
- Iteración (o repetición)

Secuencia

La ejecución del programa (algoritmo codificado en el lenguaje de programación) se realiza en orden de aparición de las sentencias. Una `sentencia` puede ser *simple* (solamente una instrucción) o *compuesta* (varias instrucciones que siguen un orden y que fueron agrupadas).

Ejemplo

```
x = 1
y = 2
cat(x, y)
```

```
## 1 2
```

En el ejemplo hay tres sentencias. Dos sentencias de asignación de valores y una sentencia de impresión. Se ejecuta una después de la otra.

Estructura if

Si la instrucción lógica es verdadera *if* imprime la siguiente instrucción. Los comandos entre llaves se comportan como una instrucción. La indentación es voluntaria, se suele utilizar para facilitar la lectura del código.

```
x = 5
y = 3
if (x > y){
    cat("x es mayor que y\n")
    cat("Este texto se imprimirá únicamente si x es mayor que y \n")
}
```

```
## x es mayor que y
## Este texto se imprimirá únicamente si x es mayor que y
```

```
cat("Este texto se imprimirá sin que ello dependa de si x es mayor que y\n")
```

```
## Este texto se imprimirá sin que ello dependa de si x es mayor que y
```

Ahora observe el cambio

```
x = 3
y = 5
if (x > y){
    cat("x es mayor que y\n")
    cat("Este texto se imprimirá únicamente si x es mayor que y \n")
}
cat("Este texto se imprimirá sin que ello dependa de si x es mayor que y\n")
```

```
## Este texto se imprimirá sin que ello dependa de si x es mayor que y
```

Los comandos agrupados entre {} después del *if* no se ejecutan porque la sentencia lógica es falsa.

Estructura if- else

Cuando hay dos alternativas de ejecución, dependiendo de la evaluación de una condición, se puede emplear la estructura if-else. Ahora observe el siguiente ejemplo.

```
x = 3; y = 5
if (x > y) {
    print("La condición es verdadera")
    print("Este texto se imprimirá únicamente si x es mayor que y")
} else {
    print('La condición es falsa')
    print("Este texto se imprimirá únicamente si x NO es mayor que y")
}
```

```
## [1] "La condición es falsa"
## [1] "Este texto se imprimirá únicamente si x NO es mayor que y"
```

```
print("Este texto se imprimirá sin que ello dependa de si x es mayor que y")
```

```
## [1] "Este texto se imprimirá sin que ello dependa de si x es mayor que y"
```

Lo que se agregó aquí fue un else. Este debe ir después de un if y se ejecuta cuando la condicion del if no se cumple, es decir cuando su valor lógico es: FALSE.

Ejercicio

Reemplaze Campo Elías por su nombre y ejecute el trozo de R, analice el resultado

```
N = "Campo Elías"
if (length(N) > 10) A="es un nombre largo" else A="es un nombre corto"
cat(N, A, sep=" ... ")
```

```
## Campo Elías ... es un nombre corto
```

Estructuras de repetición (iteración, ciclos o bucles)

Ciclo while

```
contador = 0
cat('¡Empezando!\n')
```

```
## ¡Empezando!
```

```
while (contador < 10) {
    cat(contador, ' ', end = '')
    contador = contador + 1
}
```

```
## 0   1   2   3   4   5   6   7   8   9
```

```
cat('\n¡Hecho!\n')
```

```
##
## ¡Hecho!
```

Ciclo for

```
cat('\nInicio:\n')
```

```
##
## Inicio:
```

```
for (i in 1:20) cat(i, ' ', end='')
```

```
## 1   2   3   4   5   6   7   8   9   10   11   12   13   14   15   16   17   18   19   20
```

```
cat('\nHecho.')
```

```
##
## Hecho.
```

Ejercicio Reescriba los dos ejemplos anteriores cambiado de `while` a `for` y viceversa, según el caso.

Sentencia break

```
for (i in 1:10) {
  cat(i)
  if (i == 5) break
}
```

```
## 12345
```

Estructuras de control anidadas

```
x = 5
y = 3
cat("x =", x, "y =", y)
```

```
## x = 5 y = 3
```

```
if (x > y){
    cat("\nCinco es mayor que tres\n")
    for (i in 1:10) cat(i, end=", ")
    x = 9.9
}
```

```
##
## Cinco es mayor que tres
## 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 ,
```

```
y = 2
cat("\nx =", x, ", y =", y)
```

```
##
## x = 9.9 , y = 2
```

Ejemplo de la tablas multiplicar

```
cat("\nAlgunas Tablas de Multiplicar:\n")
```

```
##
## Algunas Tablas de Multiplicar:
```

```
for (i in 6:8) { # i va aumentado desde 1, luego va a 2,3,...
    cat('\nTabla del', i,'\n')
    for (j in 1:10) cat(i, "x", j, " = ", i*j,"\n", sep='')
}
```

```
##
## Tabla del 6
## 6x1 = 6
## 6x2 = 12
## 6x3 = 18
## 6x4 = 24
## 6x5 = 30
## 6x6 = 36
## 6x7 = 42
## 6x8 = 48
## 6x9 = 54
## 6x10 = 60
##
## Tabla del 7
## 7x1 = 7
## 7x2 = 14
## 7x3 = 21
## 7x4 = 28
## 7x5 = 35
## 7x6 = 42
## 7x7 = 49
## 7x8 = 56
## 7x9 = 63
## 7x10 = 70
##
## Tabla del 8
## 8x1 = 8
## 8x2 = 16
## 8x3 = 24
## 8x4 = 32
## 8x5 = 40
## 8x6 = 48
## 8x7 = 56
## 8x8 = 64
## 8x9 = 72
## 8x10 = 80
```

Ejemplo de una sucesión creciente de asteríscos

```
# Sucesión creciente de asteríscos
num_lineas = 11
for (i in 1:num_lineas) {
    for (j in 1:i) cat('*', end='')
```

```
    cat('\n')
}
```

```
## *
## * *
## * * *
## * * * *
## * * * * *
## * * * * * *
## * * * * * * *
## * * * * * * * *
## * * * * * * * * *
## * * * * * * * * * *
## * * * * * * * * * * *
```

Funciones en R

Funciones predefinidas

Están en las librerías básicas de R es decir en las que se instalan cuando con R y que que están disponibles en cualquier sesión de trabajo, por ejemplo *sin* y *round*, *cat*:

```
round(sin(pi),4)
```

```
## [1] 0
```

```
round(sin(pi/4),4)
```

```
## [1] 0.7071
```

```
# en un vector de más dimensiones
angulo = seq(0,pi,pi/8)
cat("\nangulo:", round(angulo,4))
```

```
##
## angulo: 0 0.3927 0.7854 1.1781 1.5708 1.9635 2.3562 2.7489 3.1416
```

```
cat("\nseno:",round(sin(angulo),4))
```

```
##
## seno: 0 0.3827 0.7071 0.9239 1 0.9239 0.7071 0.3827 0
```

Ejemplo función *rbind*

```
rbind(angulo=round(angulo,4),seno=round(sin(angulo),4))
```

```
##        [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]
## angulo    0 0.3927 0.7854 1.1781 1.5708 1.9635 2.3562 2.7489 3.1416
## seno      0 0.3827 0.7071 0.9239 1.0000 0.9239 0.7071 0.3827 0.0000
```

Creación de funciones en R

Ejemplo función suma:

```
suma = function(x,y){
    s = x + y
    return(s)
}
```

Utilizándolo para sumar 3 y 5:

```
suma(3,5)
```

```
## [1] 8
```

Una función que regresa dos valores

```
mysum_prod = function(x,y){
    s = x + y
    p = x * y
    return(rbind(suma=s, producto=p))
}
```

$3 + 5$ y 3×5

```
mysum_prod(3,5)
```

```
##          [,1]
## suma        8
## producto   15
```

Seguir practicando R

El menú de RStudio permite entrar directamente a la documentación para seguir avanzado y utilizando R. Además el menú grafico tiene los comandos más utilizados en R y RStudio.

Se está utilizando mucho, por ejemplo, para producir informes seriados, por ejemplo diarios, cambiando el archivo de datos.