Programación en R y Python

Curso: Bases para Data Science - Estadística, R y Python

Katherine Morales / Néstor Montaño

Sociedad Ecuatoriana de Estadística

Octubre-2020


SOCIEDAD
ECUATORIANA
ESTADÍSTICA

Nota:
Con *Alt* + *F* o *Option* + *F* puede hacer que estas dapositivas ocupen todo el navegador (es decir que se ignore el aspecto de diapositiva que tiene por default la presentación)

Loops (for, while)

Nótese que mientras en R se trata de evitar el uso de loops (por ejemplo, usando los comando del paquete purrr), en Python el uso de loops es básico.

- Recordar que Python usa sangría en lugar de llaves
- range() es la función que crea secuencias en Python
- range() crea un objeto tipo range (no un vector como en R)
- range(start, stop) en Python es el equivalente de start:(stop-1) en R, notese que no se incluye el valor final
- range(start, stop, step) equivale a seq(start, stop-1, by= step)

Loops (for, while)

```python
# Python
for i in range(0,10):
    print(i)
```

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

```r
# R
for( i in 0:9) {
    print(i)
}
```

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

Loops (for, while)

```python
# Python
for i in range(1, 12, 2):
    print(i)
```

```
## 1
## 3
## 5
## 7
## 9
## 11
```

```r
# R
for( i in seq(1, 11, 2)) {
    print(i)
}
```

```
## [1] 1
## [1] 3
## [1] 5
## [1] 7
## [1] 9
## [1] 11
```

Loops (for, while)

```python
# Python
i = 0
while i < 10:
  print(i)
  i+=1
```

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

```r
# R
i = 0
while(i < 10){
  print(i)
  i <- i + 1
  }
```

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

Loops (for, while)

```python
# Python
# Si x es mayor a y, hacer x - y
x = 4
y = 2
if x > y: print(x - y)
```

```
## 2
```

```r
# R
# Si x es mayor a y, hacer x - y
x <- 4
y <- 2
if(x > y)  x - y
```

```
## [1] 2
```

Loops (for, while)

```python
# Python
# Si x es mayor a y, hacer x - y
# caso contrario hacer x + y
x = 4
y = 7
if x > y:
   print(x - y)
else:
   print(x + y)
```

```
## 11
```

```r
# R
# Si x es mayor a y, hacer x - y
# caso contrario hacer x + y
x <- 4
y <- 7
if(x > y){
   x - y
} else {
   x + y
}
```

```
## [1] 11
```

Loops (for, while)

```python
# Python
# Si x es mayor a y, hacer x - y
# Si son iguales hacer x * y
# caso contrario hacer x + y
x = 4
y = 4
if x > y:
   print(x - y)
elif x == y:
   print(x * y)
else:
   print(x + y)
```

```
## 16
```

```r
# R
# Si x es mayor a y, hacer x - y
# Si son iguales hacer x * y
# caso contrario hacer x + y
x <- 4
y <- 4
if(x > y){
   x - y
} else if(x==y) {
   x * y
} else {
   x + y
}
```

```
## [1] 16
```

Ejercicio

Realizar en python y en R una secuencia que empiece en *x* y el siguiente valor dependa de si *x* es par o no, si es par entonces se le aumenta 5, caso contrario se le aumenta 7, la secuencia debe terminar cuando se alcance el siguiente valor de la secuencia sea mayor a 30. Inicie con x igual a 2, en pantalla debería mostrarse 2, 7, 14, 19, 26, 31.

Ejercicio

Ejercicio

```python
#Python
x= 2
while x <= 30:
   print(x)
   if x % 2 == 0:
      x+= 5
   else:
      x+= 7
```

```
## 2
## 7
## 14
## 19
## 26
```

```r
# R
x <- 2
while(x <= 30){
   print(x)
   if(x %% 2 == 0){
      x <- x + 5
   } else{
      x <- x + 7
   }
}
```

```
## [1] 2
## [1] 7
## [1] 14
## [1] 19
## [1] 26
```

Ejercicio: Otra opcion en R

```python
#Python
x= 2
while x <= 30:
   print(x)
   if x % 2 == 0:
      x+= 5
   else:
      x+= 7
```

```
## 2
## 7
## 14
## 19
## 26
```

```r
# R
x <- 2
while(x <= 30){
   print(x)
   x <- ifelse(x %% 2 == 0, x+5, x+7 )
}
```

```
## [1] 2
## [1] 7
## [1] 14
## [1] 19
## [1] 26
```

Funciones

Supongamos que queremos tener la capacidad de con un sólo comando generar la secuencia realizada anteriormente para cualquier valor de x.

Pues, esto se resuelve con una **función** que reciba un valor y a cambio retorne la secuencia realizada antes, vamos a aprender como definir funciones con python y R.

Ejercicio: Otra opcion en R

```python
# Python
def muestra_secuencia(x):
    while x <= 30:
        print(x)
        if x % 2 == 0:
            x+= 5
        else:
            x+= 7
muestra_secuencia(2)
```

```
## 2
## 7
## 14
## 19
## 26
```

```r
# R
muestra_secuencia <- function(x){
   while(x <= 30){
      print(x)
      x <- ifelse(x %% 2 == 0, x+5, x+7 )
   }
}
muestra_secuencia(2)
```

```
## [1] 2
## [1] 7
## [1] 14
## [1] 19
## [1] 26
```

Ejercicio: Otra opcion en R

```python
# Python
def muestra_secuencia(x):
    while x <= 30:
        print(x)
        if x % 2 == 0:
            x+= 5
        else:
            x+= 7
a= 2
muestra_secuencia(a)
```

```
## 2
## 7
## 14
## 19
## 26
```

```python
a
```

```
## 2
```

```r
# R
muestra_secuencia <- function(x){
   while(x <= 30){
      print(x)
      x <- ifelse(x %% 2 == 0, x+5, x+7 )
   }
}
a <- 2
muestra_secuencia(a)
```

```
## [1] 2
## [1] 7
## [1] 14
## [1] 19
## [1] 26
```

```r
a
```

```
## [1] 2
```

Funciones

La función anterior sólo imprime en pantalla resultados, pero una función puede retornar valores.

Ahora vamos a realiar una función que (como hicimos anteriormente) reciba dos valores, **x**, **y** y retorne

- Si x es mayor a y, hacer x - y
- Si son iguales hacer x * y
- caso contrario hacer x + y

Ejercicio: Otra opcion en R

```python
# Python
# Si x es mayor a y, hacer x - y
# Si son iguales hacer x * y
# caso contrario hacer x + y
def funcion_2_xy(x, y):
    if x > y:
       return x - y
    elif x == y:
       return x * y
    else:
       return x + y

funcion_2_xy(5, 3)
```

```
## 2
```

```python
funcion_2_xy(2, 2)
```

```
## 4
```

```python
funcion_2_xy(2, 7)
```

```
## 9
```

```r
# R
# Si x es mayor a y, hacer x - y
# Si son iguales hacer x * y
# caso contrario hacer x + y
funcion_2_xy <- function(x, y){
   if(x > y){
      x - y
   } else if(x==y) {
      x * y
   } else {
      x + y
   }
}

funcion_2_xy(5, 3)
```

```
## [1] 2
```

```r
funcion_2_xy(2, 2)
```

```
## [1] 4
```

```r
funcion_2_xy(2, 7)
```

```
## [1] 9
```

¿Cómo se sienten luego de esta avalancha de información?

Fin

Curso: Bases para Data Science - Estadística, R y Python

Katherine Morales / Néstor Montaño