**Semana 2**

**07. Estructuras de Control**

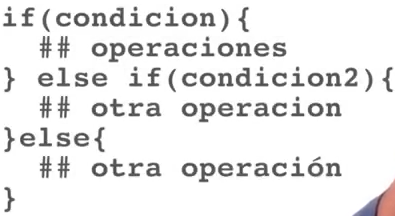
En este módulo aprenderemos a programar en R por primera vez. Lo primero será aprender las diferentes estructuras de control que se pueden usar en R.

Las estructuras de control nos permiten controlar el flujo de ejecución de un programa. Dependiendo de si se cumplen ciertas condiciones en el momento, también permiten llevar a cabo tareas o una operación de manera repetida. Hay varios tipos:

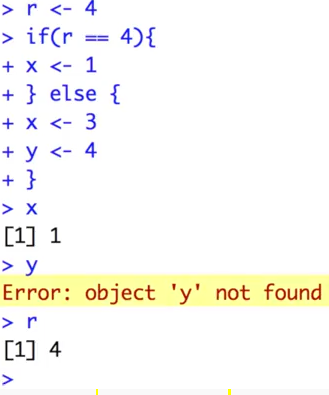
* Condicionales: **if, else**
* Bucle de Repetición: **for**
* Bucle de condición: **while**
* Bucle Infinito **repeat**
* Romper un bucle: **break**
* Saltarse un bucle: **next**

1. **La estructura if**

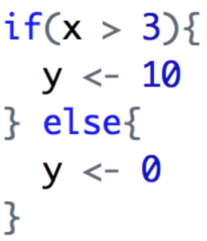
La primera estructura de control que vamos a estudiar es **if**, que para representarse en el programa sigue la siguiente estructura:



En las llaves vamos a poner las operaciones que se van a llevar a cabo. Podemos poner más condiciones poniendo otro else con otro if. A continuación podemos encontrar un ejemplo.



La variable y da error porque no fue creada al ser r igual a 4. Si r valiera 3, entonces y tendría un valor. Podemos observar que la forma idiomática o más usada es:

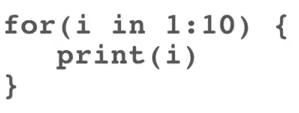


Dentro de las llaves se ponen las operaciones que van a llevarse a cabo. Otra forma muy utilizada en R es una asignación **if** a un objeto (por ejemplo y), asignándolo con el operador “<-”, dejando claro que el valor de la variable de ese valor.

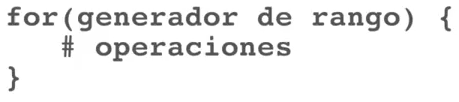
Se puede prescindir de **else** cuando usamos una estructura de control **if.** Esto es muy común utilizarlo.

1. **La estructura de control for**

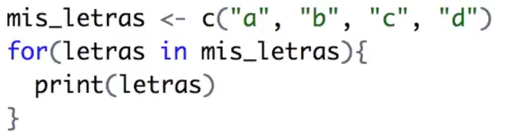
En este caso es para un bucle de repetición. La forma más común de usarlo es como



Es muy fácil ver este ejemplo, porque lo que hace es tomar cada uno de los valores de i y luego imprime ese valor. Lo que sucede es la forma general del **for**:

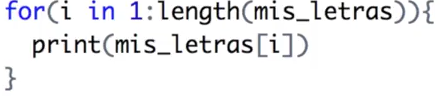


Un generador de rangos es una estructura que podemos ir recorriendo 1 a 1. El objeto va tomando esos valores y los va utilizando o puede servir dentro de la estructura **for** para llevar a cabo una tarea. Normalmente es suficiente poner e número de veces que se quiere hacer una operación. Por ejemplo, en este caso:



Podemos generar un vector llamado mis\_letras de caracteres, que se puede utilizar un generador de rangos de letras. Cada valor va tomando los valores del vector mis\_letras, yendo imprimiendo en cada uno de ellos las letras.

La función *seq\_along()* genera un vector donde los elementos son los enteros que aparecen dentro del vector que se le aporte. En lugar de la función tal cual, va imprimiendo el valor de manera indexada. También se puede hacer en lugar de *seq\_along()* podemos indicar las posiciones con los dos puntos.



**For** también funciona con vectores de números. Por ejemplo, en la siguiente función va tomando los valores desde 1 hasta 100 de 0,5 en 0,5, y luego lo imprime.

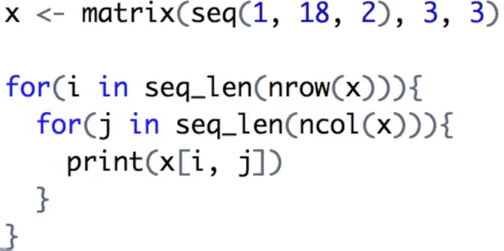
*x<-seq(from=1,to=100,by=0,5)*

*for(i in x){*

*print(i)*

*}*

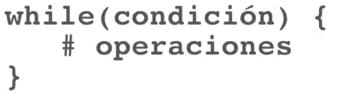
También podemos formar bucles anidados, es decir, un **for** dentro de esto, lo que sirve para leer todos los elementos de una matriz o un data frame.



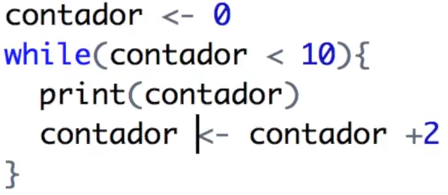
Lo que se observa es que se recorre la matriz una por una, recordando que como se llenan de izquierda a derecha y de arriba abajo. No debe abusarse de los bucles anidados, porque se van almacenando y se pierde en el programa. Se recomienda que se vayan haciendo de dos en dos.

1. **La estructura de control while**

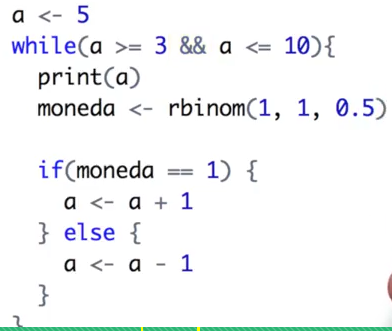
**While** que es un bucle de condición (significa “mientras”), no sabe el número de pasos que va a durar una condición, y la forma de representar la operación es de la siguiente forma:



Por ejemplo, generamos una variable *contador*, y mientras que sea 10, si entramos en *while* lo primero que hace es imprimir primero el *contador* y luego le va sumando de 2 en 2, imprimiendo los valores 0, 2, 4, 6…

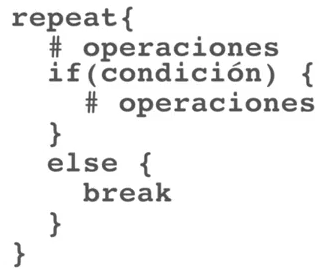


**While** puede tener más de una condición por medio de operadores lógicos.



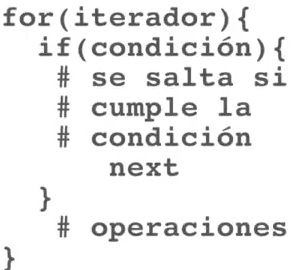
1. **Combinación de repeat y break**

**Repeat** entra directo al bucle y una vez dentro podemos preguntar por medio de **if** si la condición se cumple o no.

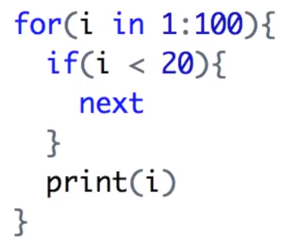


También podemos usar un **break** para romper el operador y seguir con la fluidez del programa. Sin embargo, puede ser que la condición de parada nunca se cumpla y se genere un **bucle infinito**. Debemos tener cuidado con esto, al igual que usamos un bucle **while**, ya que si siempre se cumple se puede producir.

La palabra **next** se usa para saltarse los pasos de un bucle.



Por ejemplo, en una estructura de control for, lo que podemos hacer es que imperamos sobre un factor de 1 a 100, pero si nos queremos saltar los 20 primeros valores, se puede usar la siguiente estructura:



Si esta i es menor que 20 nos la saltamos con la estructura **if** dentro del **for** y dentro del cuerpo de **if** incluimos un **next**.

1. **Resumen de la lección**

* **If, while** y **for** se usan como estructuras de control para controlar el flujo de un programa en R.
* Hay que evitar generar bucles infinitos, que generan problemas.
* Usar estas estructuras de modo interactivo puede volverse complicado, por lo que se recomienda utilizar las funciones **\*apply.**