

ÚNETE A NUESTRO PRÓXIMO MEETUP

¿CÓMO LO HAGO UNA Y MIL VECES?

Estructuras de loop en R



Azalea Reyes Aguilar

Profesora en la Facultad de Psicología,
UNAM. RLadies Querétaro



8 Jun 2023



6 p. m. GMT-6
7 p. m. GMT-5

REGISTRATE

meetup.com/es/rladies-cuernavaca

Síguenos en



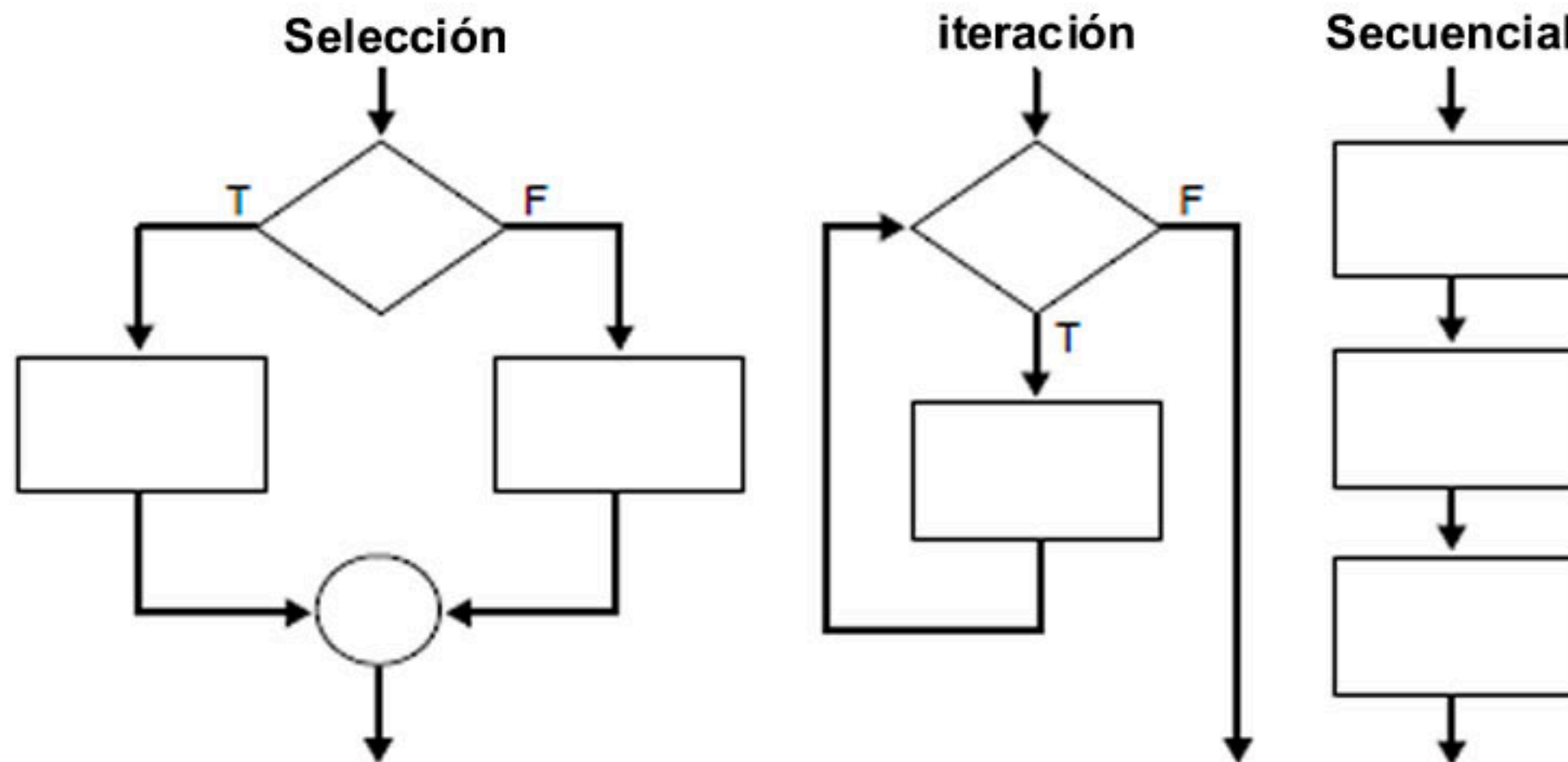
@RLadiesCuernavaca



[github.com/RLadiesCuerna](https://github.com/RLadiesCuernavaca)

ESTRUCTURAS DE CONTROL

CONTROLAR LA MANERA EN QUE SE EJECUTA NUESTRO CÓDIGO



determinar la lógica y el orden en que ocurren las operaciones

ESTRUCTURAS DE CONTROL

¿PARA QUÉ SIRVEN?

- Automatización de tareas repetitivas
- Procesamiento de grandes conjuntos de datos
- Flexibilidad y generalización
- *Optimización de recursos
- Implementación de algoritmos complejos

ESTRUCTURAS DE CONTROL

> ?control

Usage

```
if(cond) expr  
if(cond) cons.expr else alt.expr  
  
for(var in seq) expr  
while(cond) expr  
repeat expr  
break  
next
```

ESTRUCTURAS DE CONTROL

Estructuras condicionales

{

Usage

```
if(cond) expr
```

```
if(cond) cons.expr else alt.expr
```

Estructuras iterativas

{

```
for(var in seq) expr
```

```
while(cond) expr
```

```
repeat expr
```

cláusulas

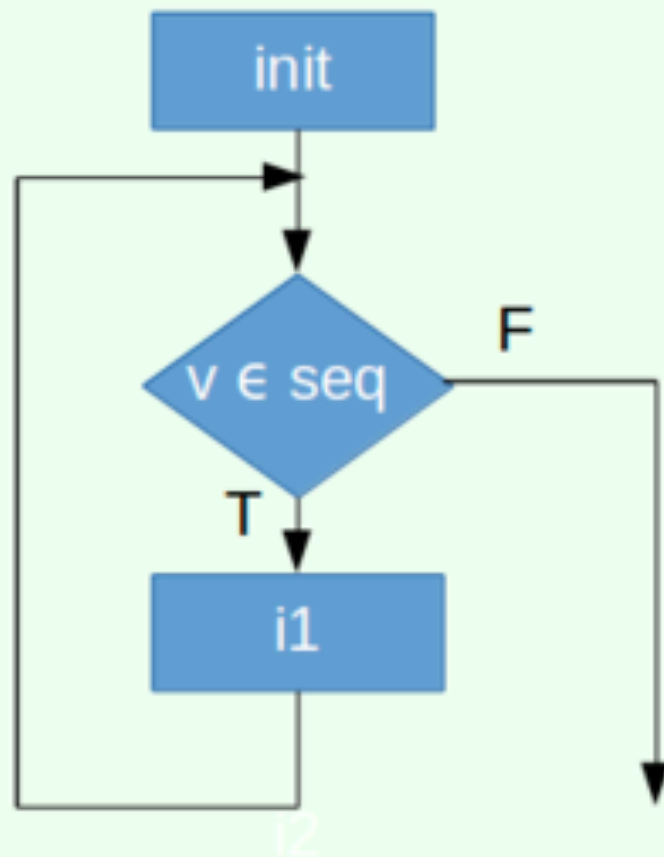
{

```
break
```

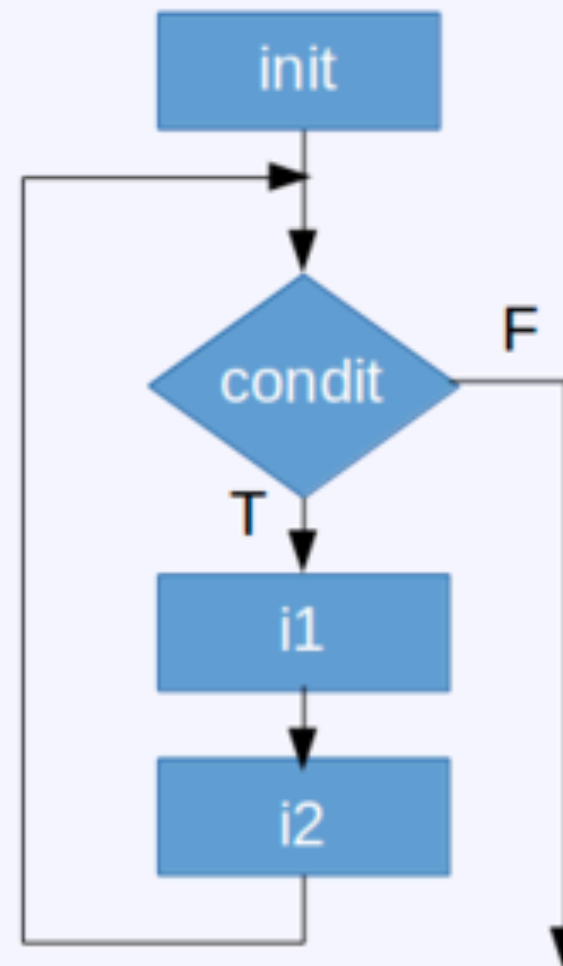
```
next
```

ESTRUCTURAS DE CONTROL ITERATIVAS

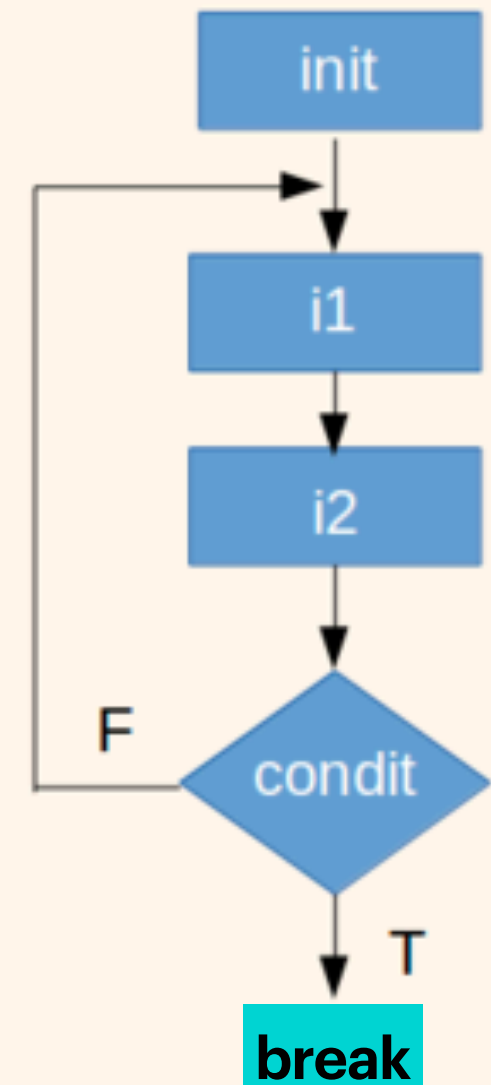
For loop



while loop



repeat loop





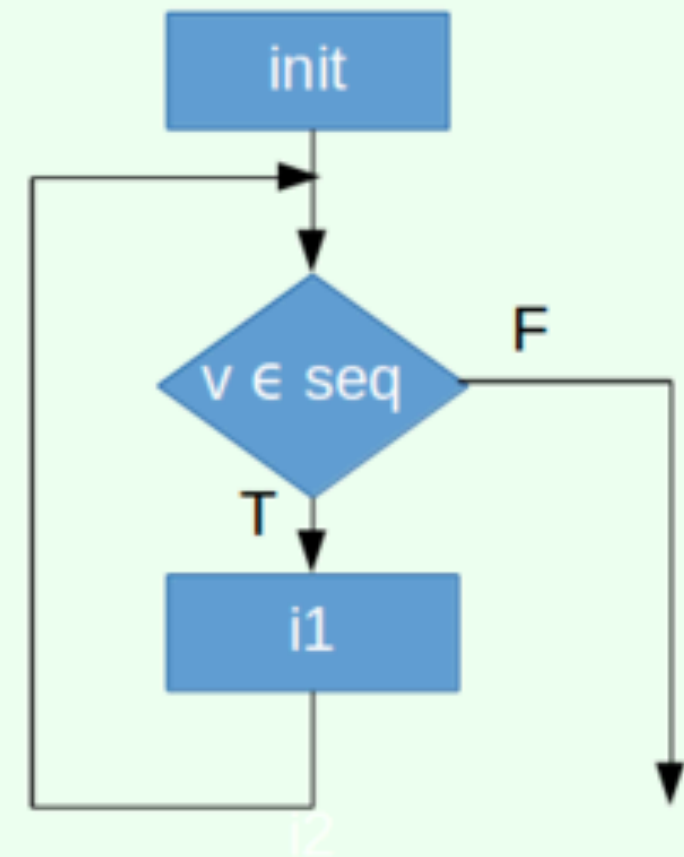
for

for

Loop finito, el número de iteraciones se conoce y se especifica al inicio

```
for(elemento in objeto) {  
    operacion_con_elemento  
}
```

For loop



for

Loop finito, el número de iteraciones se conoce y se especifica al inicio

```
for(elemento in objeto) {  
    operacion_con_elemento  
}
```

PARA cada elemento **EN** un objeto,
haz la siguiente operación



break & next

break

nos permite **interrumpir** un bucle

```
└  
for(i in 1:10) {  
  · print(i)  
}  
└
```

```
[1] 1  
[1] 2  
[1] 3  
[1] 4  
[1] 5  
[1] 6  
[1] 7  
[1] 8  
[1] 9  
[1] 10
```

```
└  
for(i in 1:10) {  
  · if(i == 3) {  
    · · break  
  }  
  · print(i)  
}  
└
```

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

next

```
└  
for(i in 1:10) {  
  · print(i)  
}  
└
```

```
[1] 1  
[1] 2  
[1] 3  
[1] 4  
[1] 5  
[1] 6  
[1] 7  
[1] 8  
[1] 9  
[1] 10
```

nos deja avanzar a
la **siguiente** iteración del bucle,
“saltándose” la actual

```
for(i in 1:10) {  
  · if(i == 3) {  
    · · next  
  }  
  · print(i)  
}
```

```
[1] 1  
[1] 2  
[1] 4  
[1] 5  
[1] 6  
[1] 7  
[1] 8  
[1] 9  
[1] 10
```



while

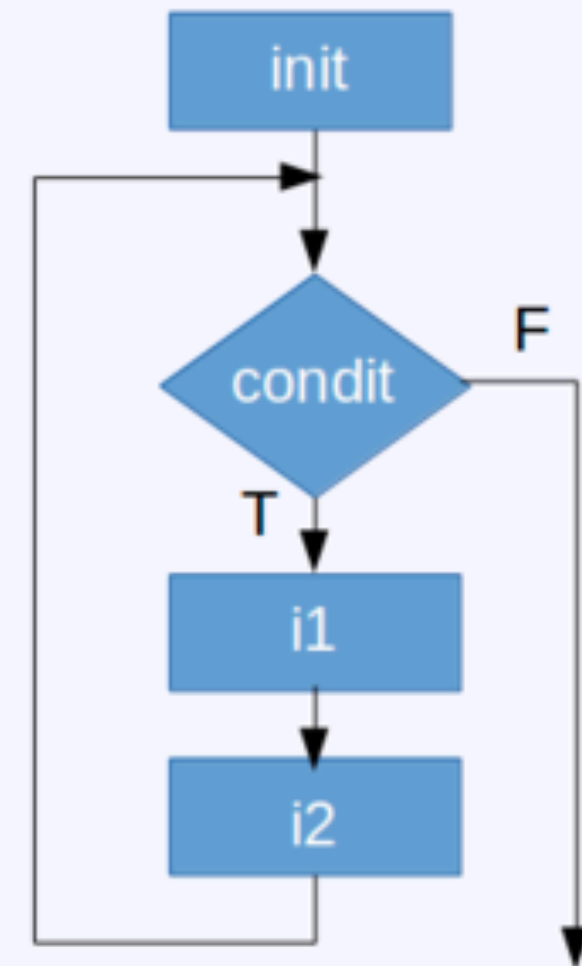
while

Loop infinito, el número de iteraciones NO SE CONOCE [mientras sea verdadera una condición]

Requiere una condición lógica [inicial] para determinar el número de iteraciones

```
while(condicion) {  
    operaciones  
}
```

while loop



while

Loop infinito, el número de iteraciones NO SE CONOCE [mientras sea verdadera una condición]

Requiere una condición lógica [inicial] para determinar el número de iteraciones

```
while(condicion) {  
    operaciones  
}
```

MIENTRAS esta condición sea VERDADERA,
haz estas operaciones



repeat

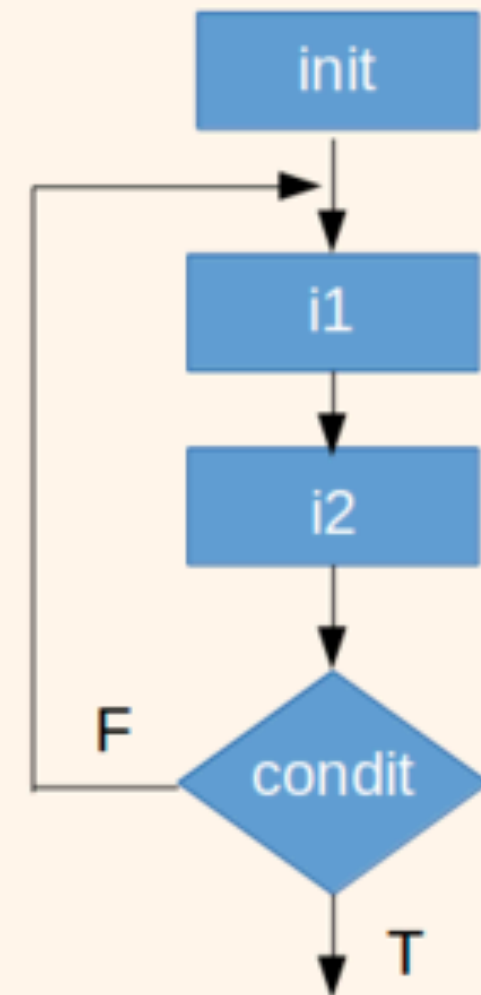
repeat

Loop infinito, el número de iteraciones NO SE CONOCE

Requiere una condición lógica [final (if-break)] para que el loop se detenga

```
repeat {  
    operaciones  
  
    un_break_para_detener  
}
```

repeat loop



repeat

Loop infinito, el número de iteraciones NO SE CONOCE

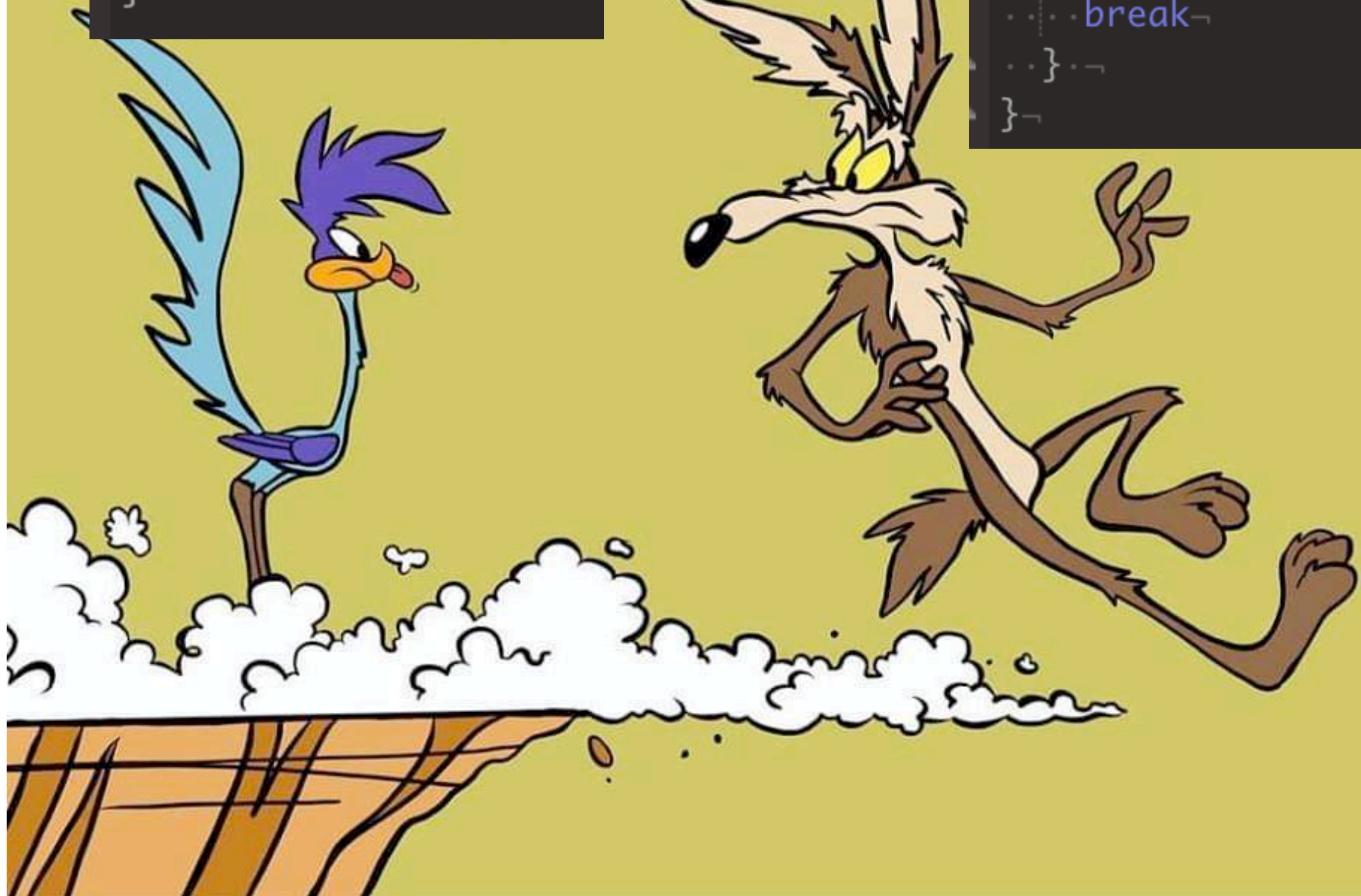
Requiere una condición lógica [final (if-break)] para que el loop se detenga

```
repeat {  
    operaciones  
  
    un_break_para_detener  
}
```

Repite esta operación,
if (condicion)
break

```
while (superficie) {  
  .. corre  
}
```

```
repeat {  
  .. correr  
  .. if (!superficie) {  
    .. break  
  }  
}
```





FAMILIA: apply()

FAMILIA NUMEROSA

reciben como argumentos a un objeto y al menos una función.

```
apply(X, MARGIN, FUN)
```

```
apply(X = df, MARGIN = 2, FUN = mean)
```

```
apply(df, 2, mean)
```

- `apply()`
- `eapply()`
- `lapply()`
- `mapply()`
- `rapply()`
- `sapply()`
- `tapply()`
- `vapply()`

How to use apply() functions

```
apply(X = data, MARGIN = 2, FUN = mean)
```

indiv	height_0	height_10	height_20
A	15	20	23
B	10	18	24
C	12	14	18
D	9	15	17
E	17	19	26

mean	12.6	17.2	21.6
------	------	------	------








library(purrr)



GRACIAS



R-LADIES CUERNAVACA

parade = c(, , , , )

```
for (monster in parade) {  
  if (shape(monster) == triangle) {  
    monster_style = monster + sunglasses  
  }  
  else {  
    monster_style = monster + hat  
  }  
  print(monster_style)  
}
```

