{ Funciones + Loops }

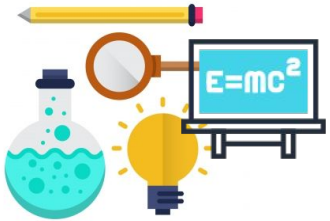
PROCESO DE ANÁLISIS

{PREGUNTAS}

{Datos}

{ EDA }

{ Modelo }



{ Visualización}

{ Toma de decisiones }

{ Conocimiento }

METODOLOGIA: KABAN

"Durante las pasadas tres décadas, Kanban, que se define como **«un sistema de producción altamente efectivo y eficiente«**, ha contribuido a generar un panorama manufacturero óptimo y competitivo. El origen de la metodología Kanban debemos buscarlo en los procesos de producción **"just-in-time" (JIT) ideados por Toyota,** en los que se utilizaban tarjetas para identificar necesidades de material en la cadena de producción."

https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/


B3K

METODOLOGIA: KABAN

TOYOTA GLOBAL KNOWLEDGE CENTER

La Tarjeta Kanban

- Just-in-Time: Tarjeta Kanban

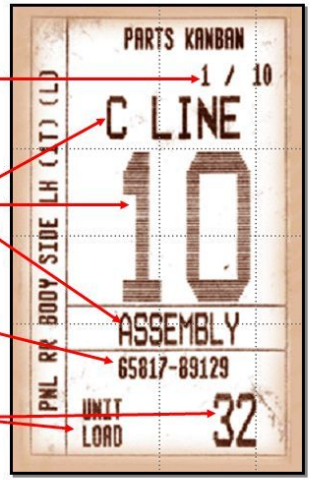
  Mucha información en poco papel

Número de envíos
Este es el envio 1 de 10

Dónde se necesita la parte
Estación 10 de la línea de ensamble C

Qué parte se necesita
Panel lateral trasero izquierdo

Cuántas partes se entregarán por envío



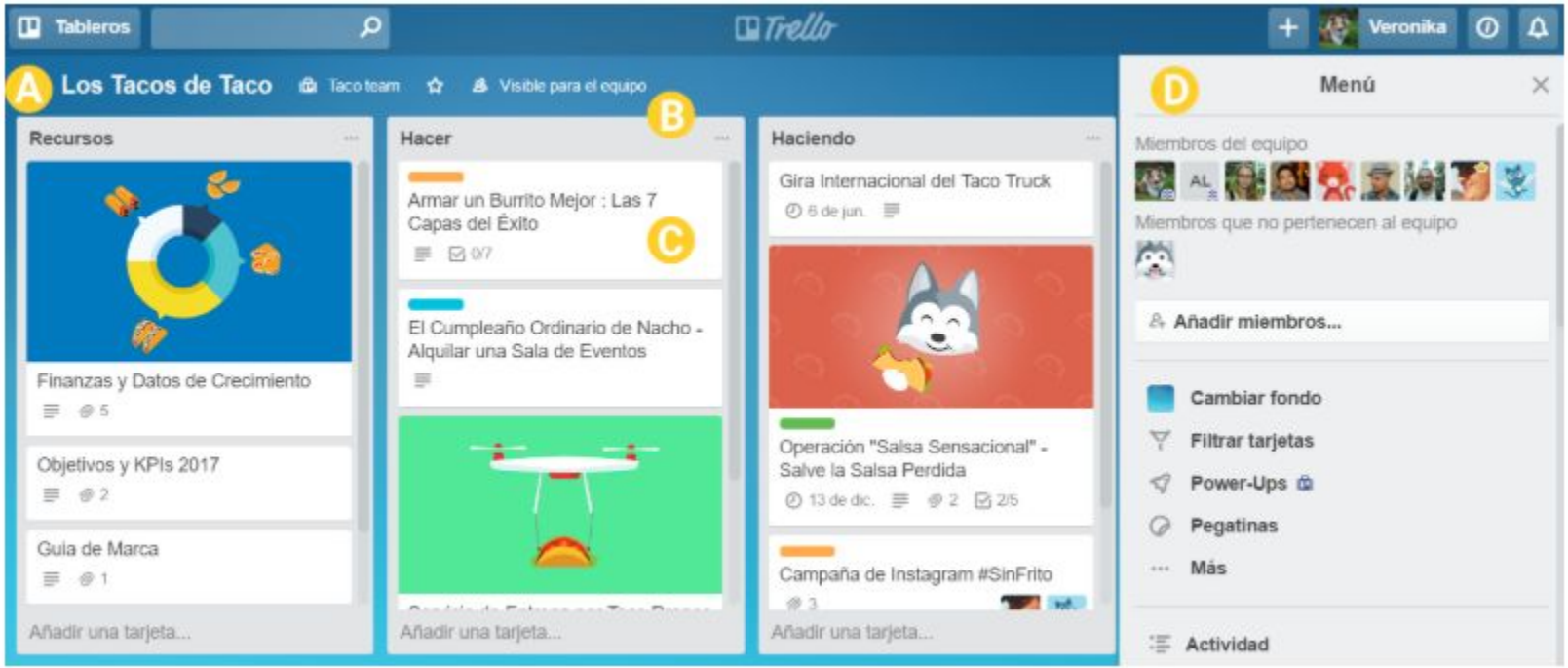
Un total de 320 (10 envíos de 32) paneles laterales traseros izquierdos serán entregados a la estación 10 de la línea de ensamble C

METODOLOGIA: KABAN

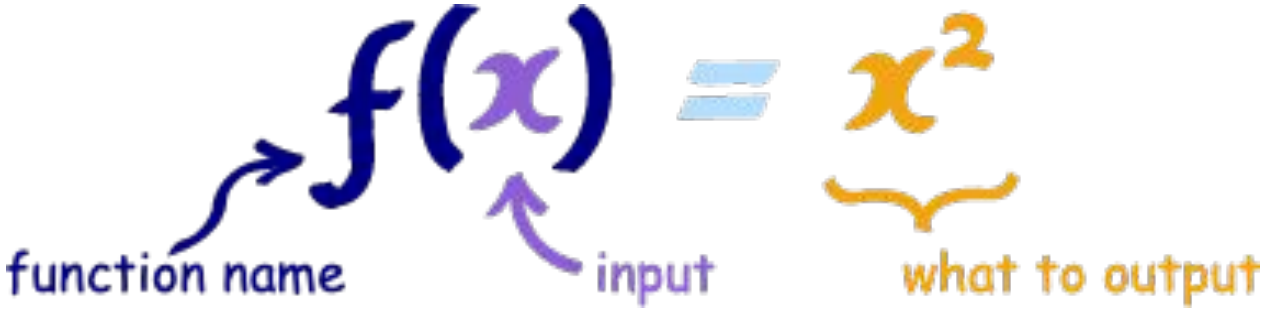
METODOLOGIA: KABAN

1. Visualice el flujo de trabajo
2. Limite la cantidad de Trabajo en Proceso
3. Realice un seguimiento de su tiempo
4. Lectura fácil de indicadores visuales
5. Identifique los cuellos de botella

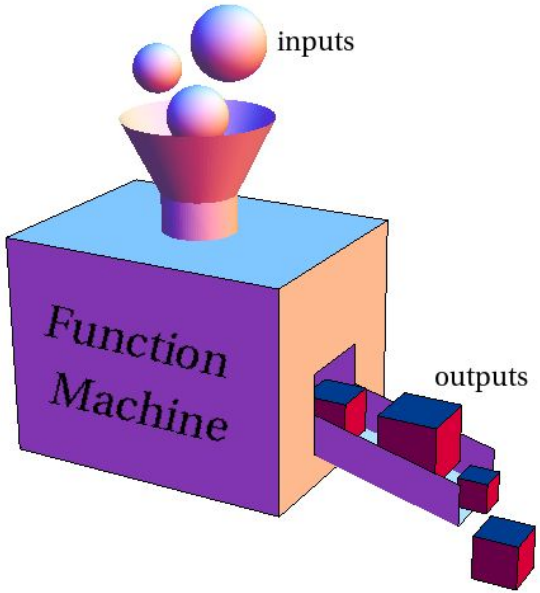
METODOLOGIA: KABAN



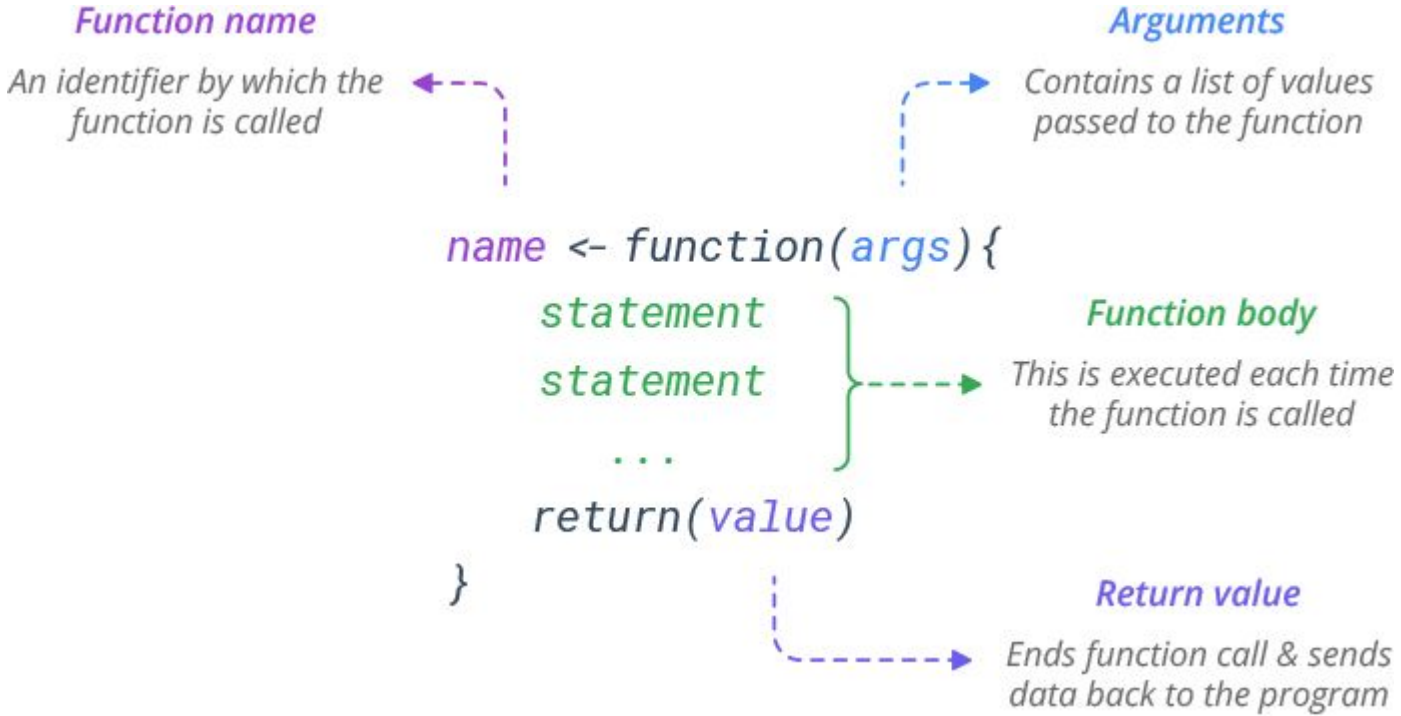
Funciones en R



Funciones en R



Funciones en R

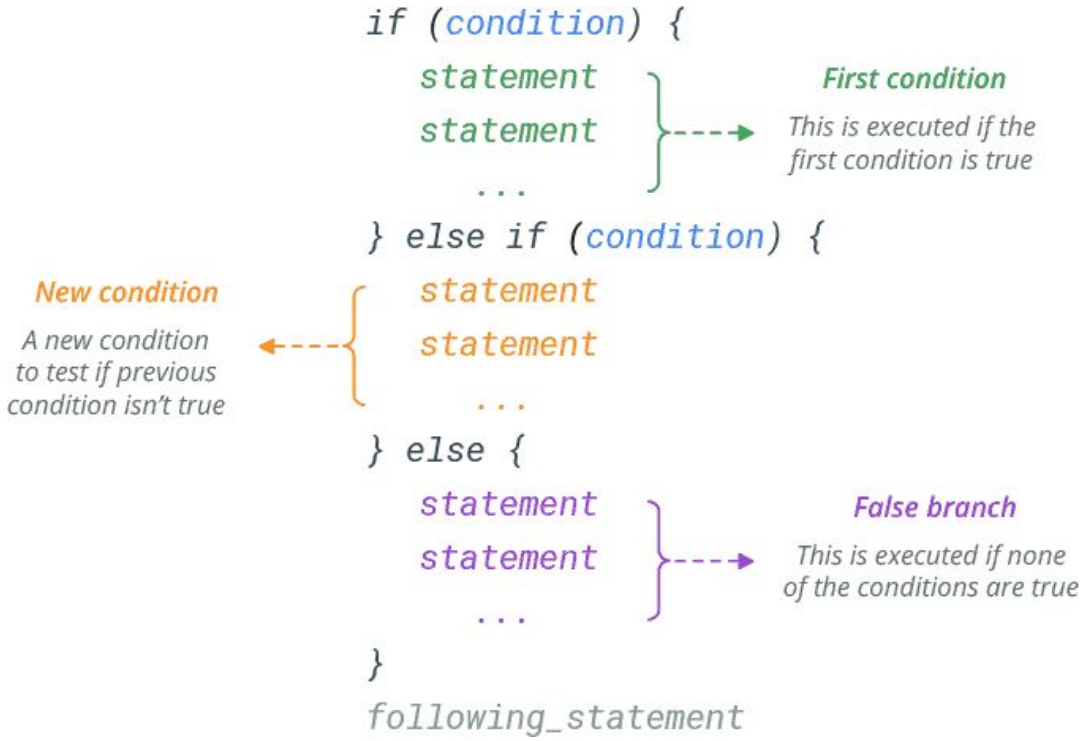


Funciones en R

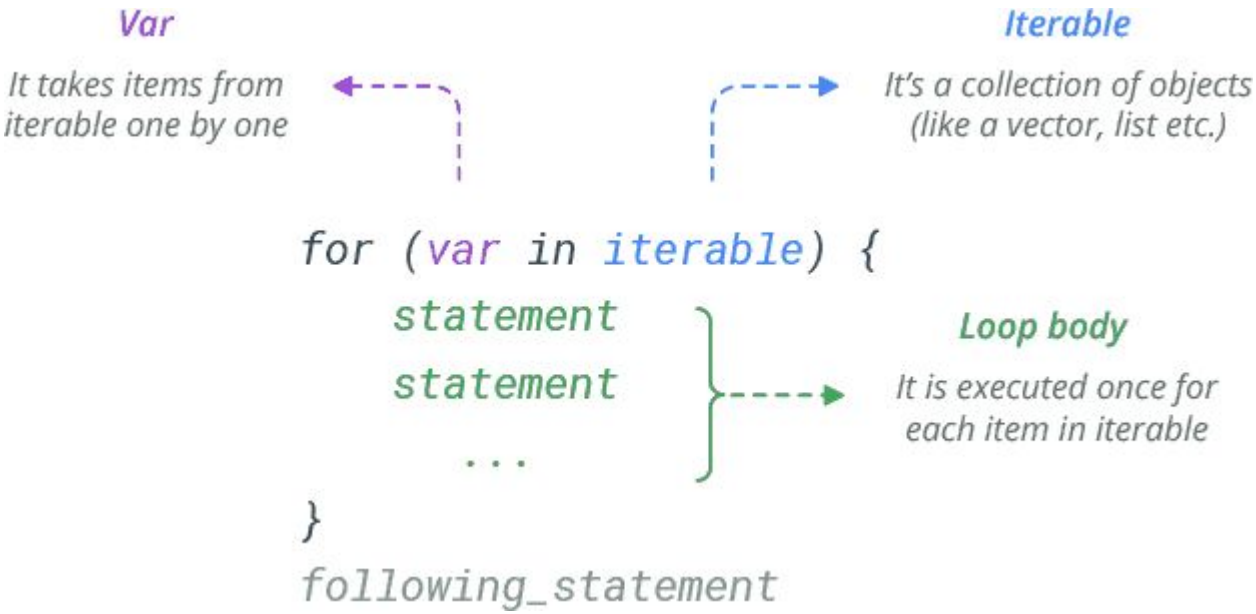
```
calculos <- function(x, y) {
 suma <- x + y
 resta <- x - y
 multiplicacion <- x * y
 division <- x / y
 return (c(cal_suma = suma, cal_resta = resta,
   cal_multiplicacion = multiplicacion, cal_division = division))
}

calculos(6, 3)
```

If else if


if (condition) {
statement
statement
...
First condition
This is executed if the first condition is true
} else if (condition) {
New condition
A new condition to test if previous condition isn't true
statement
statement
...
} else {
statement
statement
...
False branch
This is executed if none of the conditions are true
}
following_statement

{ Ciclos: For }


Var
It takes items from iterable one by one
Iterable
It's a collection of objects (like a vector, list etc.)
for (var in iterable) {
statement
statement
...
}
following_statement
Loop body
It is executed once for each item in iterable

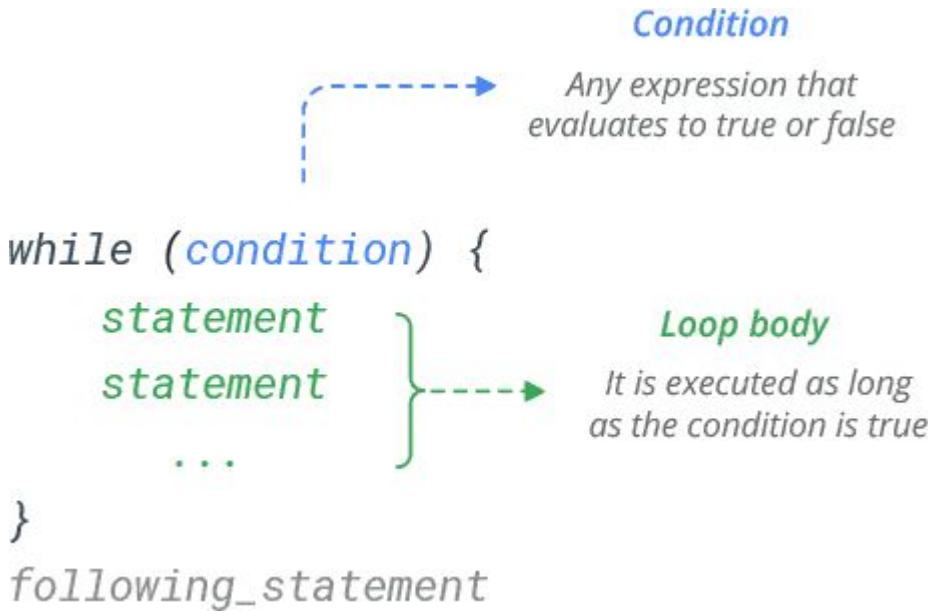
{ Ciclos: For }

```r
num1 <- 3:10

#forma I
for (x in num1) {
  print(x*x)
}

#forma II
for (x in 1:length(num1)){
  print(num1[x]*num1[x])
}
```

{ Ciclos: While }



{ Ciclos: While }

```
while(condicion booleana ){

    # Bloque de código
}
```

Ejemplo:

```
i<-0
while (i<10){
    print(i)
    i<-i+1 #incremento
}
```