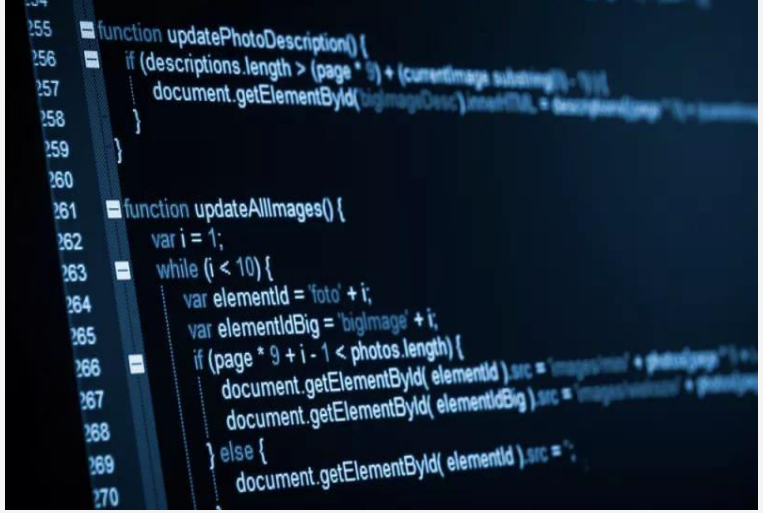
Introducción a la línea de comandos y a la programación para análisis bioinformáticos.



Teórico 6

Introducción a la programación.

La programación

- Evoluciona: diferentes lenguajes, nacen, mueren, cambian y derivan de otros lenguajes.
- Todos mantienen conceptos clave y los bloques básicos de la construcción de programas informáticos.
- Los problemas de análisis deben descomponerse en una serie de acciones o procesos programables (con las herramientas que uno conoce!)
- Una de las claves es comenzar con programas pequeños con objetivos simples, y luego reutilizar o reciclar partes de programas que ya hemos escrito.

Recordar que...

- Programas compiladas (C, C++)
  - compilador
    - una vez se traduce de “humano” a “computadora”
- Programas interpretados (Python, Bash, R, Perl, Matlab, rubi…)
  - interpretador
    - mientras se corre se va traduciendo de “humano” a “computadora”.

Conceptos:

- **Argumentos**: archivos sobre los que el programa opera. **Opciones**: (tipo de argumento?) valores que se envían al programa para modificar su funcionamiento al momento de ejecutarse.

- **Código fuente** o simplemente **Código**: Un programa, en general se refiere al texto donde se define su funcionamiento. El código se escribe!

- **Ejecutar** o **correr**: es la acción de poner a operar un programa.

Conceptos:

- **Función**: es un "sun"programa una rutina que puede ser ejecutarse dentro de un programa repetidamente dentro un un programa.
- **Parsear**: extraer datos particulares de un texto, usualmente más grande.
- **Retorna o devuelve**: lo que te devuelve como resultado una función o programa.

Variables:

- Similar a matemática. “Una variable es un nombre, en este caso X, que **puede tomar cualquier valor**”
  - X + 5 = 73
  - X + 68 = 4
  - X + “ndo” = “lindo”
- Para realizar acciones repetidas sobre un conjunto de elementos distintos.
  - Acción: “Hola **$Nombre**, como estas?”
  - Variable: Nombre

Variables:

- **Atributos**:
  - ***Nombre*:** usualmente texto plano, única palabra.
  - ***Tipo*:** designa el tipo de datos que contiene la variable.
  - ***Valor*:** la información que la variable contiene en un determinado momento.
  - ***Alcance*:** indica donde, en que ambiente, la variable puede ser utilizada.

Variables:

- Atributos:
  - ***Nombre*:**
    - evitar caracteres raros, “.” al final, #, “ “, no utilizar número al principio de las variables.
    - en algunos lenguajes se debe indicar la variable con “$”
    - deben ser distintivos y descriptivo, un “ayuda memoria” (debemos recordar que contiene)

Variables:

- Atributos:
  - ***Tipo*:**
    - Entero: número sin fracción decimal, negativos y positivos.
      - Ejs. -2, 34556, -8849, 0
      - Rango +/- 2.14 x 10^9
      - Otras alternativas similares: long o unsigned
    - Flotante: número con fracciones decimales. Negativos o positivos.
      - Ejs. 23.7, -0.098, $3x10^{79}$ [3E79]

Variables:

- Atributos:
  - ***Tipo*:**
    - Booleano: Verdadero (1) o Falso (0).
      - Se utilizan para operaciones lógicas.
    - *String* (cadena de texto): secuencia de caracteres de texto. Puede incluir caracteres como letras, dígitos , puntuación, finales de líneas y otros (no recomendados).
      - Ejs. arbol, ty56, J345_34h, 456.67, FFF_Fe.2
      - Dentro de los programas los *strings* suelen delimitarse por comillas simples o dobles.
    - Es posible y a veces necesario:
      - cambiar el tipo de una variable.
      - establecer explícitamente que tipo de variable es cada una

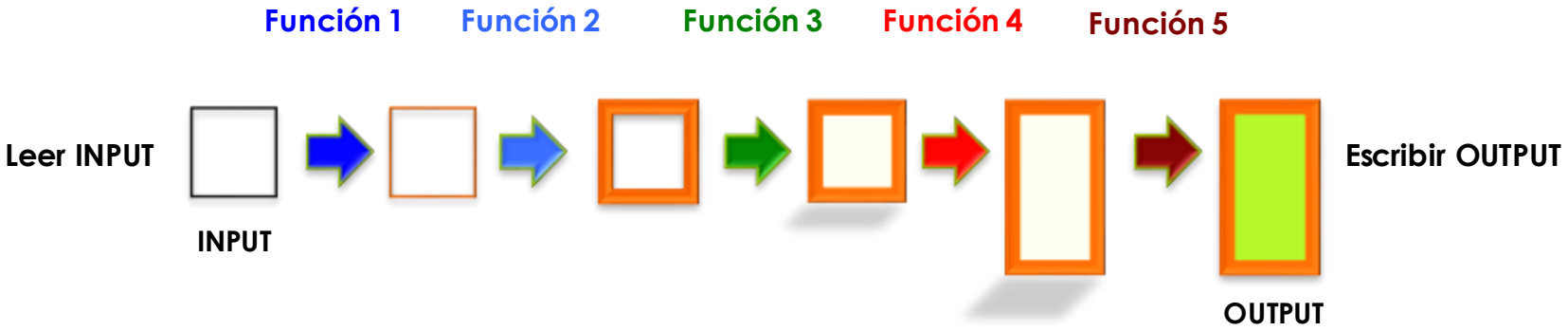
Variables:

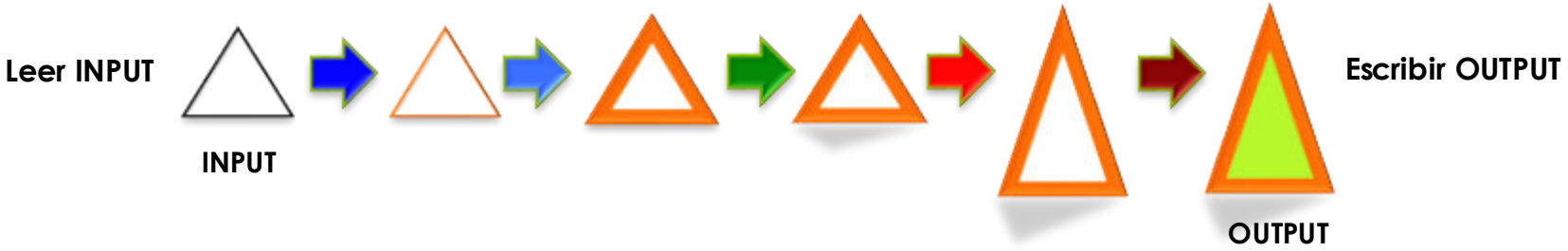
- Pueden contener otras variables dentro.
  - ***Arrays*** o matriz: una colección de datos dentro de una variable.
    - Una (vector o lista) o múltiples dimensiones.
      - Ejs.:
        - lista1=[aa, ab, ac, ad, af…]
        - matriz1=[[32, a], [2, b], [12, c]]
        - mat=[[2.4, 3.2, 18.9], [6.3, 0.9, 7], [1.2, 23.2, 4.3], [1.0, 8.3, 4]]
        - juego=[“pato1”, “pato2”, “pato3”, “ganso”]
    - En la mayoría de los lenguajes es posible acceder a la posición del valor dentro del *array* dando las coordenadas.
      - Ejs.: matriz1[2] devuelve “a”

Qué podemos hacer con una variable?

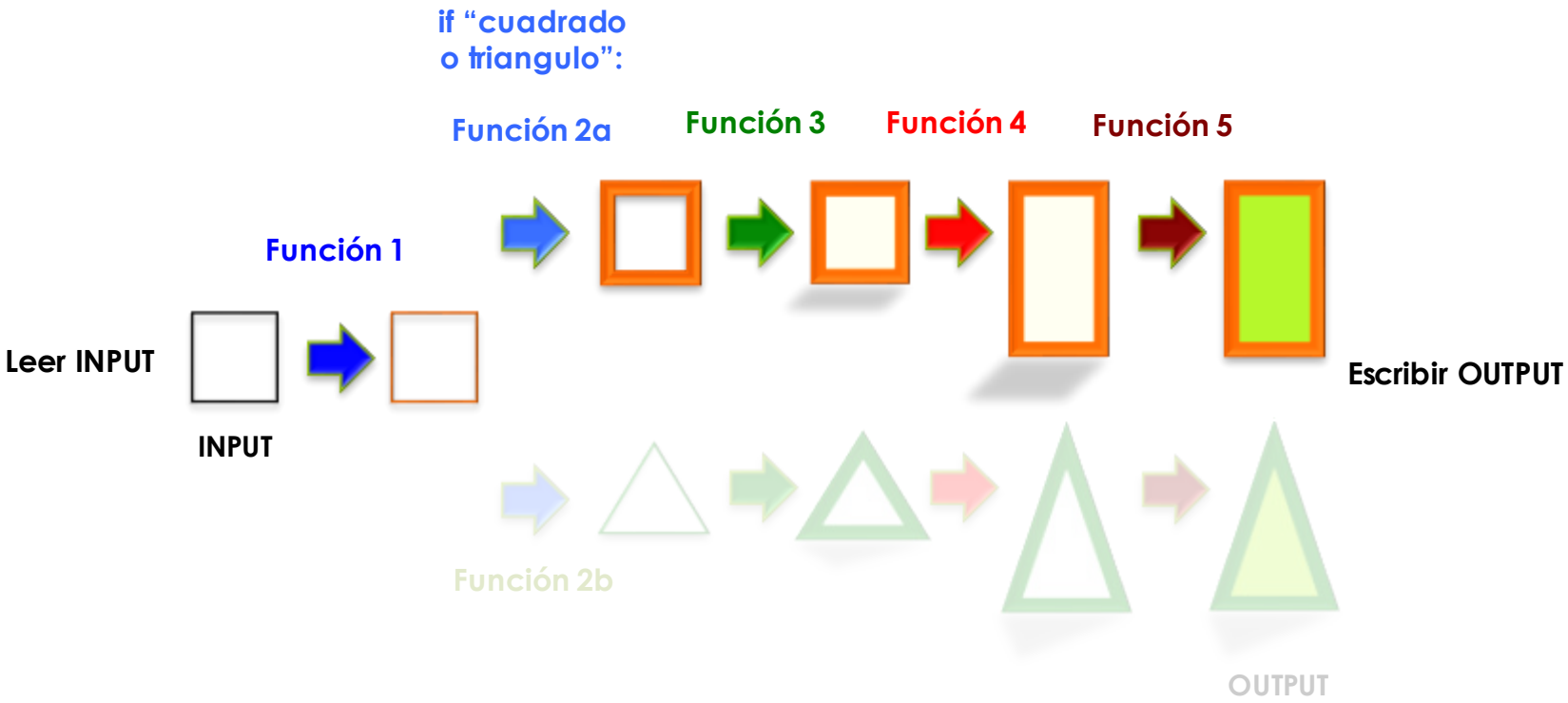
- Operaciones **matemáticas**
  - valor_x=$(echo 30)
    1. expr $valor_x / 2
    2. expr $valor_x * 2 * $valor_x / 5
    3. echo "" | awk '{print '$valor_x' /234}'
- Operaciones **lógicas**
  - valor_y=$(echo furnarius)
    1. expr $valor_y == "furnarius"
    2. expr $valor_y != $valor_x
    3. expr $valor_x \< 3
- Pasar **parámetros** a las funciones dentro de un programa.

Procesos: **lineales**, decisiones y *loops* (bucle)

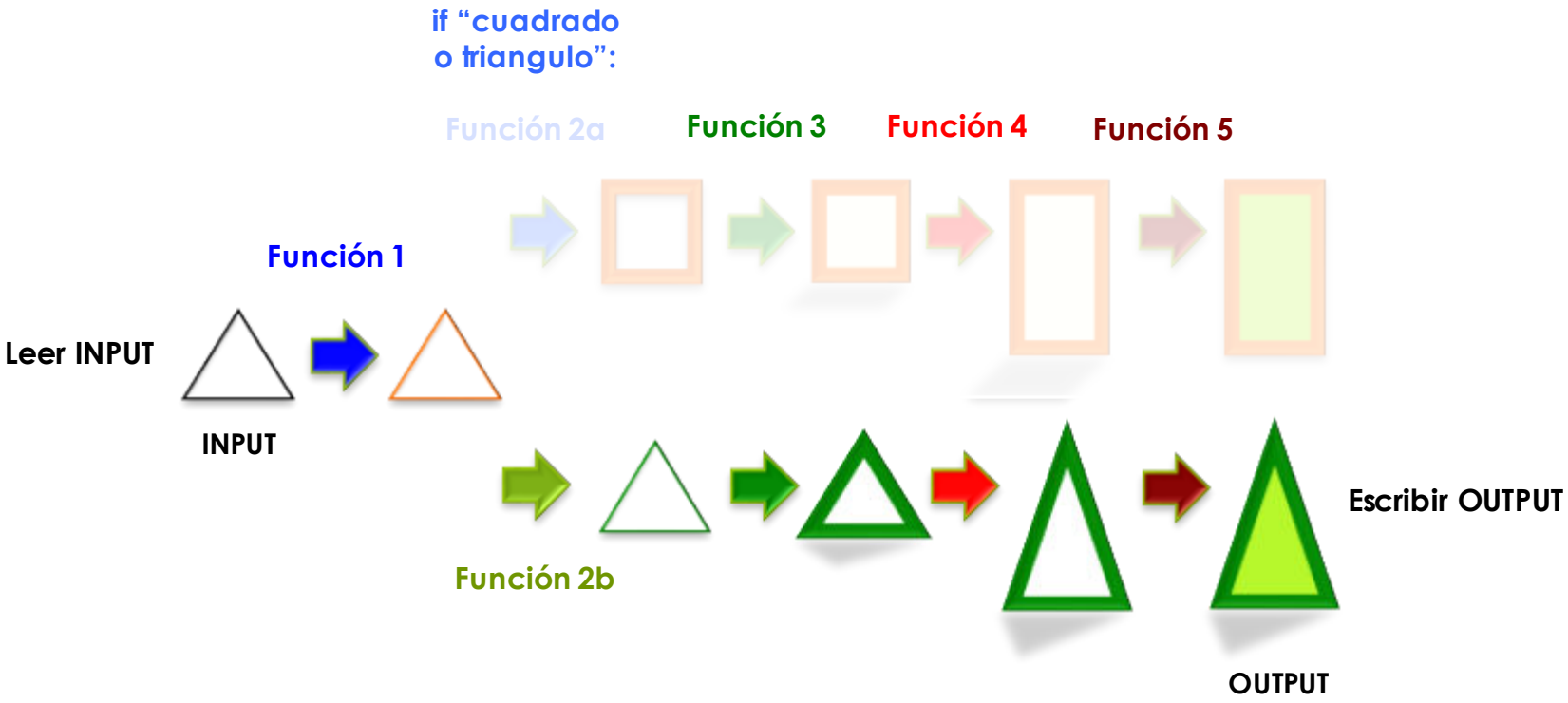




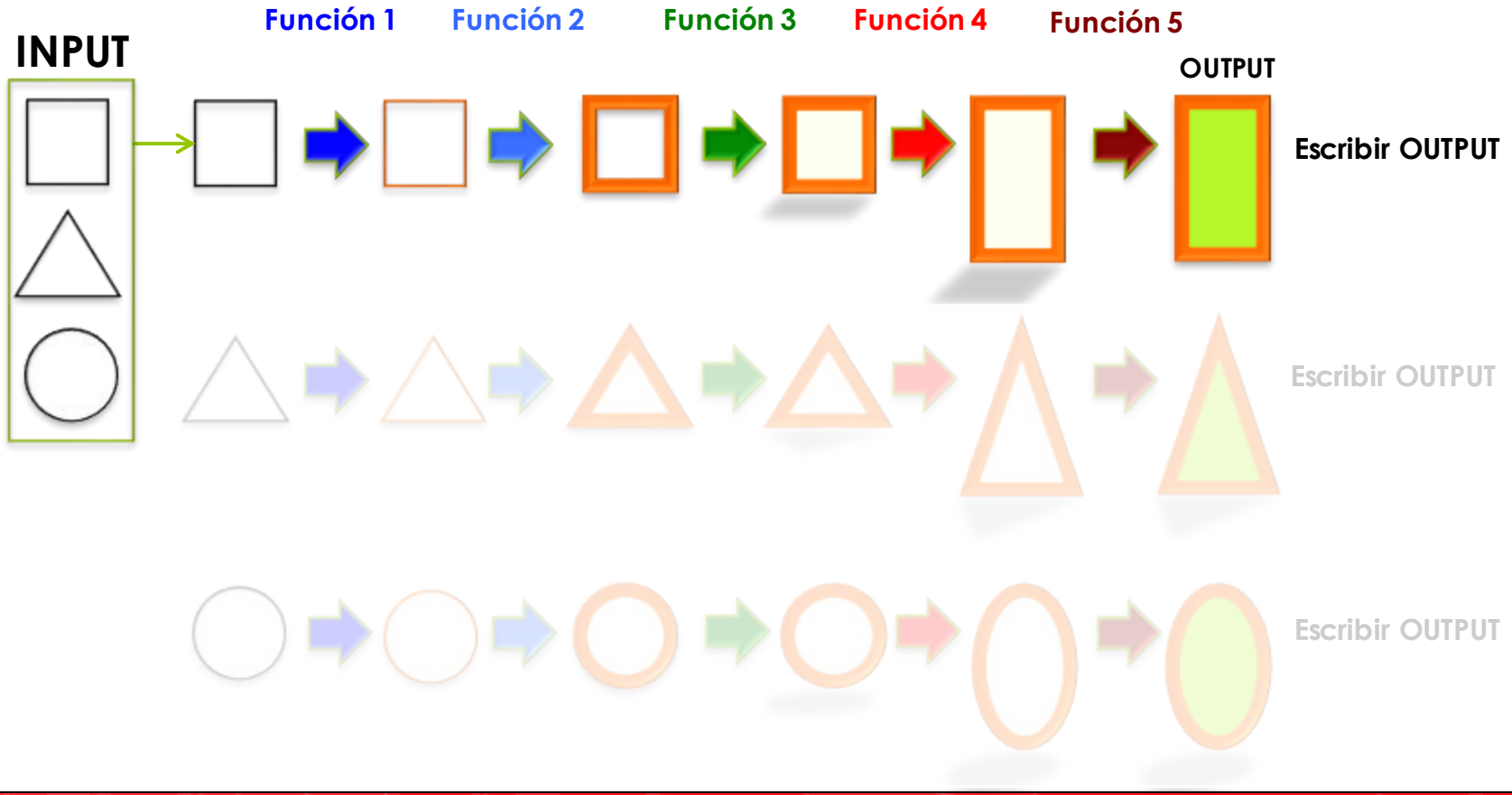
Procesos: lineales, **decisiones** y *loops* (bucle)



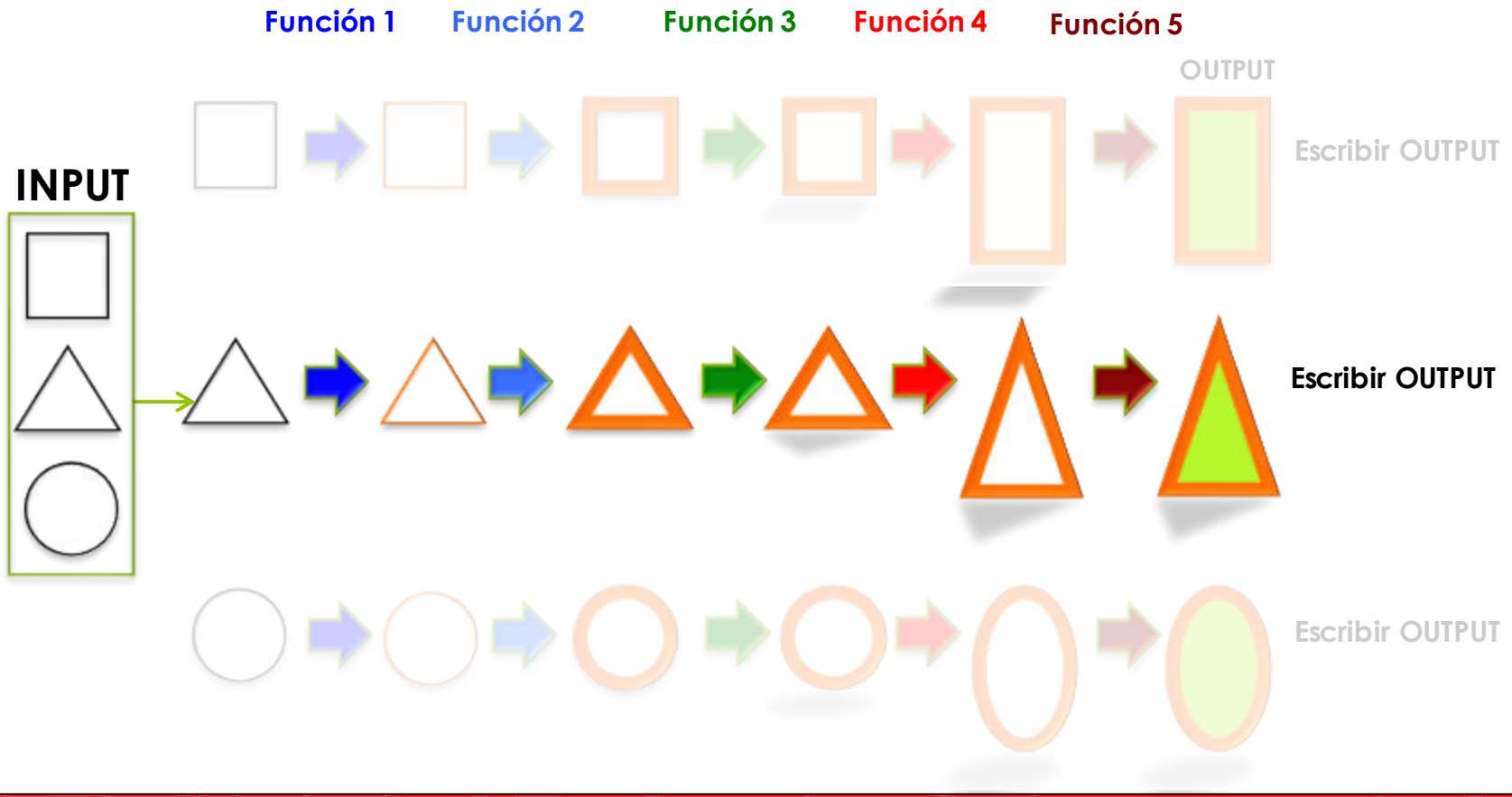
Procesos: lineales, **decisiones** y *loops* (bucle)



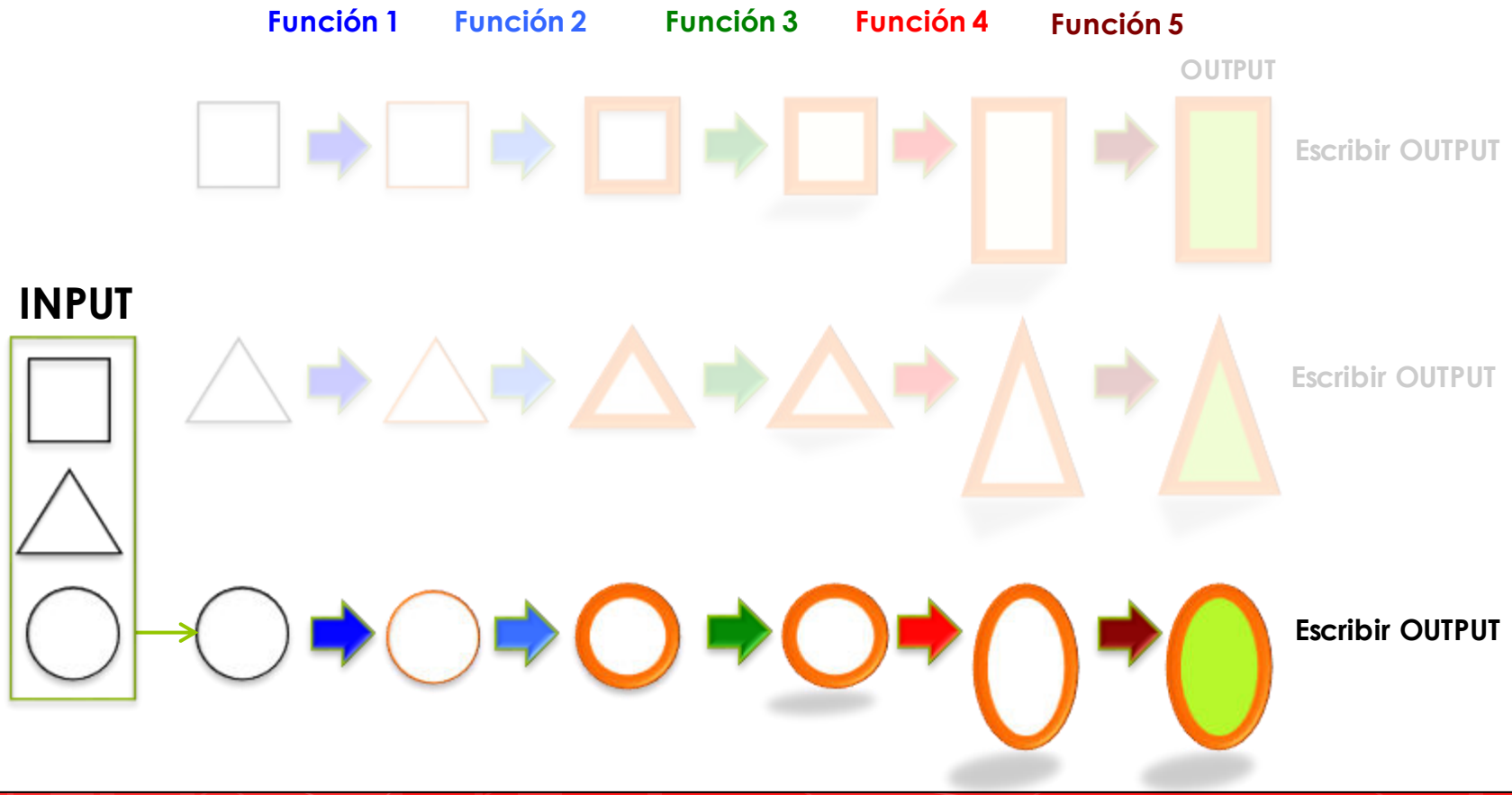
Procesos: lineales, decisiones y ***loops* (bucle)**



Procesos: lineales, decisiones y ***loops* (bucle)**



Procesos: lineales, decisiones y ***loops* (bucle)**

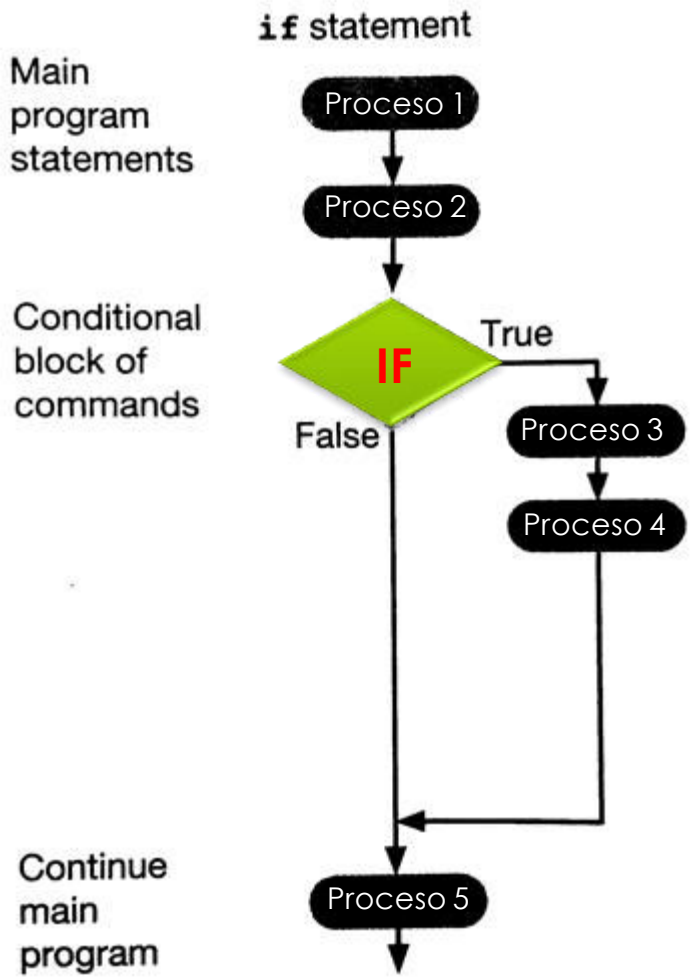


Control de flujo

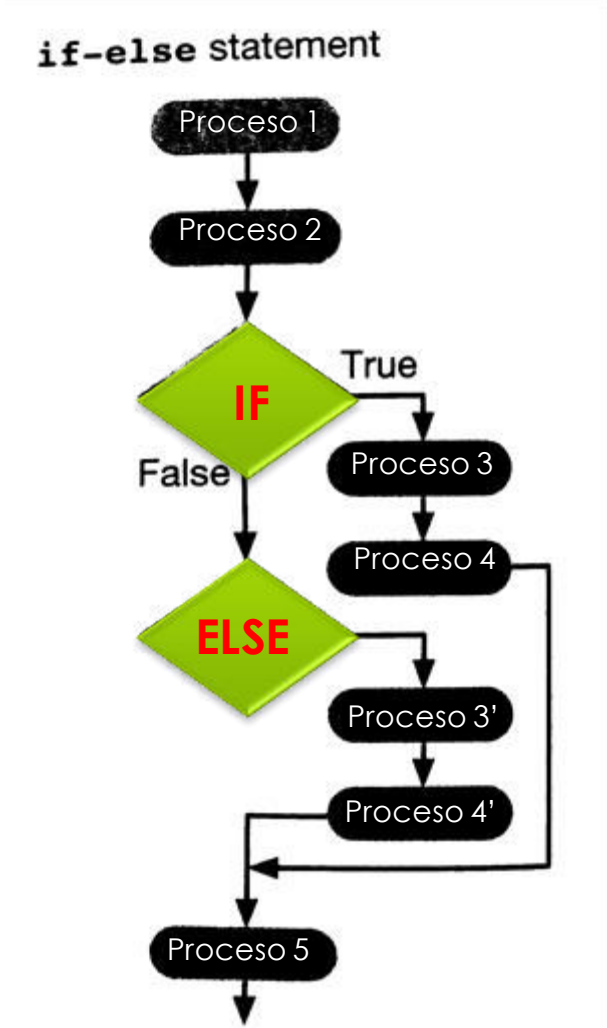
- Decisiones: "**If**"
- Repetir proceso (loops):
  - en una lista (**for**)
  - hasta cumplir un determinado (**while**)

> Definen uno de los bloques de construcción de programas

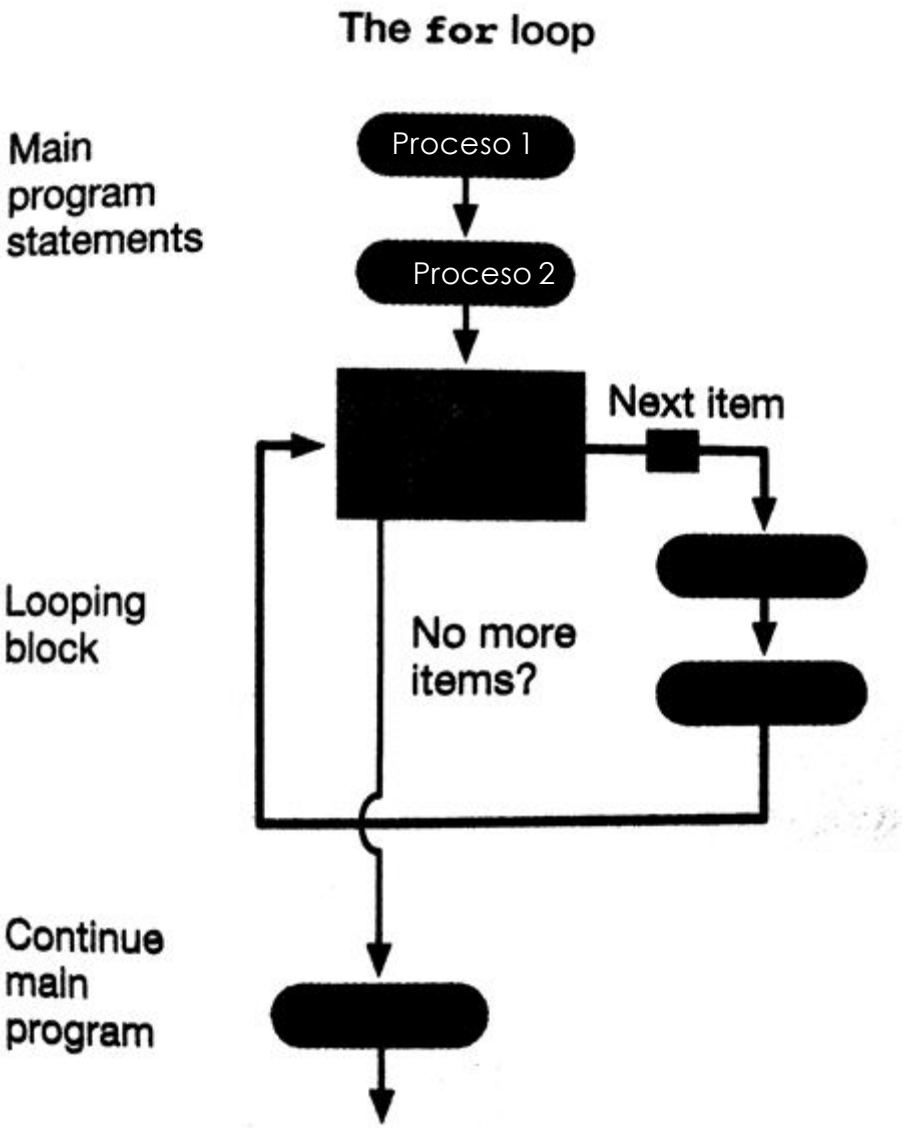
> Es la clave para automatizar procesos!


if statement
Main program statements
Proceso 1
Proceso 2
Conditional block of commands
IF
True
False
Proceso 3
Proceso 4
Continue main program
Proceso 5

if


if-else statement
Proceso 1
Proceso 2
IF
True
False
Proceso 3
Proceso 4
ELSE
Proceso 3'
Proceso 4'
Proceso 5

if - else


The for loop
Main program statements
Proceso 1
Proceso 2
Next item
Looping block
No more items?
Continue main program

for