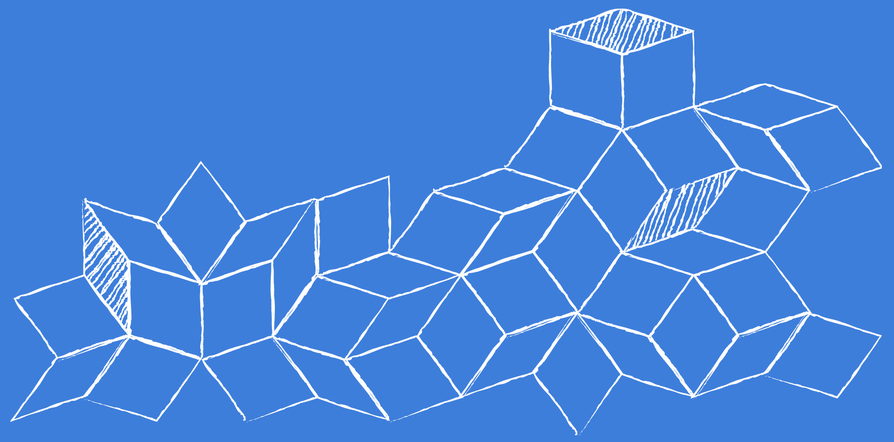
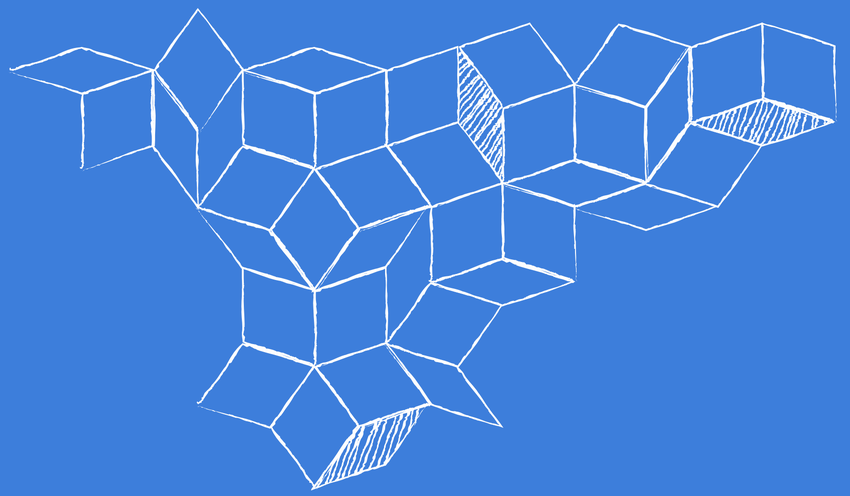


STEMUC

Academia para **escolares**



Suma MAT UC

precingeniería

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

PRIMEROS PASOS
EN PROGRAMACIÓN

Clase 3

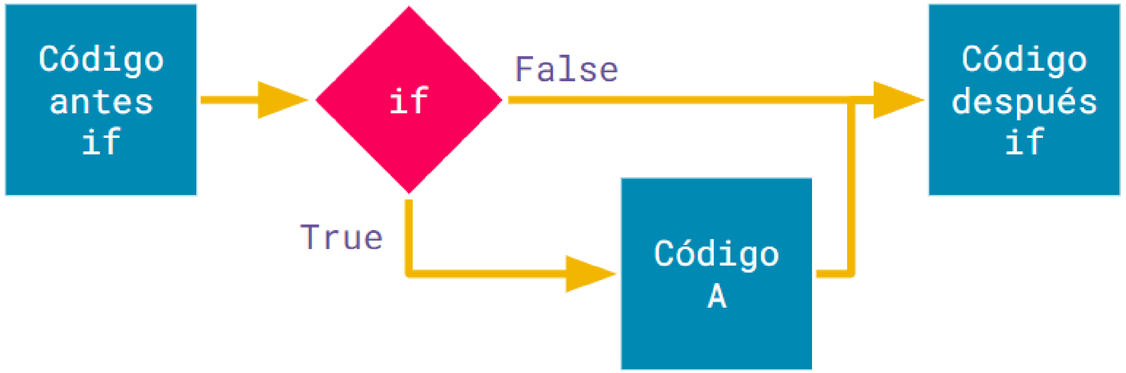
Instrucciones repetitivas y aleatoriedad

Otra vez aprendimos
muchas cosas ayer…

Repasémoslo un poco

if

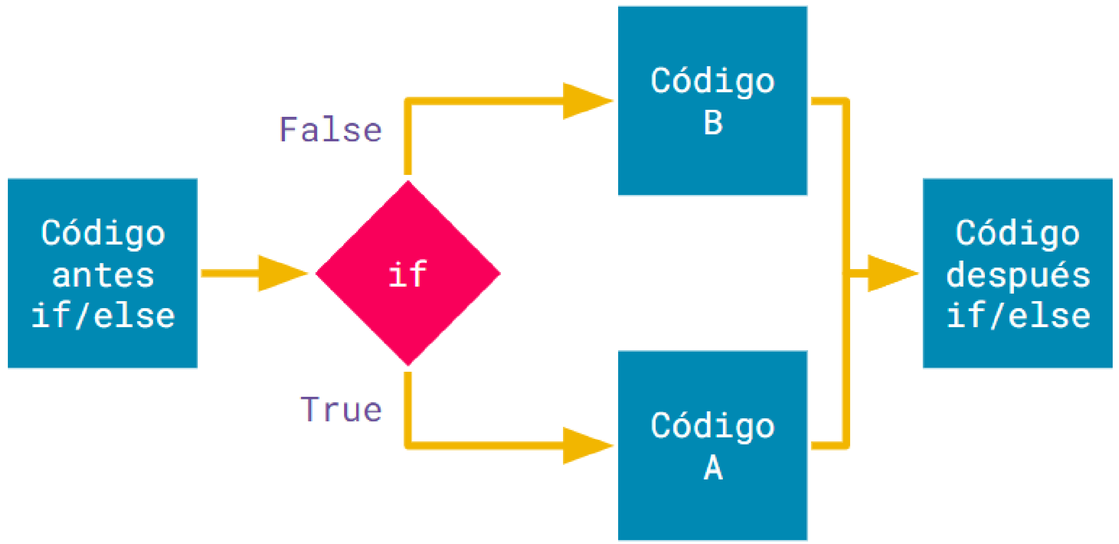
La instrucción `if` permite ejecutar una sección de código **si** se cumple una condición.




```
# Código antes
if condición:
    # Código A
    # …

# Código después
```

else

La instrucción `else` permite ejecutar una sección de código **si no** se cumple una o más condiciones anteriores.



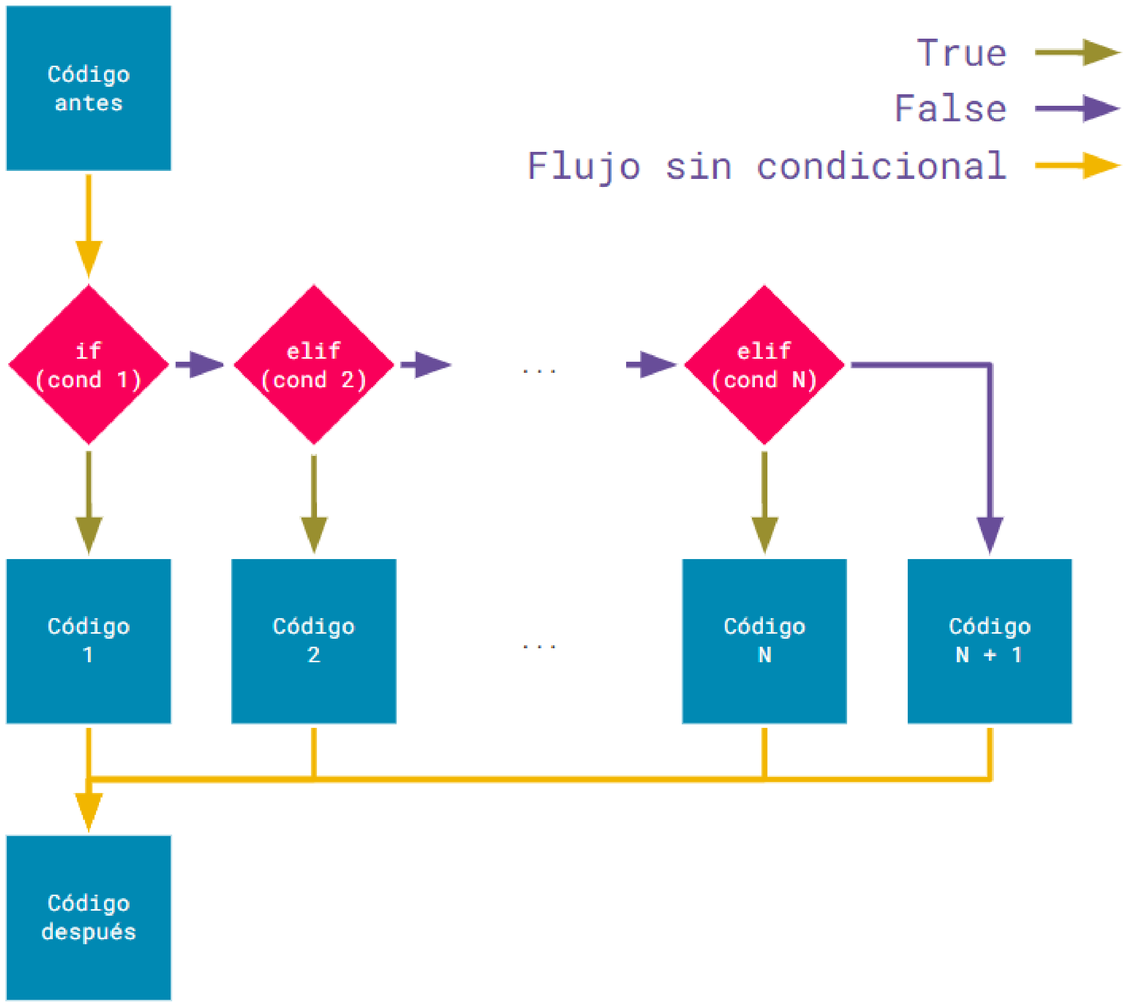

```
# Código antes
if condición:
    # Código A
    # …
else:
    # Código B
    # …

# Código después
```

elif

La instrucción `elif` permite ejecutar una sección de código si se cumple una condición y no se ha cumplido ningún `if` o `else` anterior.

```
# Código antes
if condición_1:
    # Código 1
    # …
elif condición_2:
    # Código 2
    # …
# …
elif condición_N:
    # Código N
    # …
else:
    # Código N + 1
    # …

# Código después
```

elif




```
# Código antes
if condición_1:
    # Código 1
    # …
elif condición_2:
    # Código 2
    # …
# …
elif condición_N:
    # Código N
    # …
else:
    # Código N + 1
    # …

# Código después
```

Ayer quedaron algo pendientes con los Quiz

Aprovechen de terminarlos para calentar la materia

Ahora sí,

¡vemos materia nueva!

En la clase de hoy…

- Aprenderemos la instrucción `while`.
  Con ellos crearemos **flujos repetitivos** o iterativos.

- Conocerán el módulo `random` y las instrucciones `random` y `randint`.
  Con ellas le daremos **aleatoriedad** a nuestros programas.

Control de flujo: ciclos o *loops*

En muchos programas es necesario repetir operaciones o líneas de código. Casos típicos incluyen:

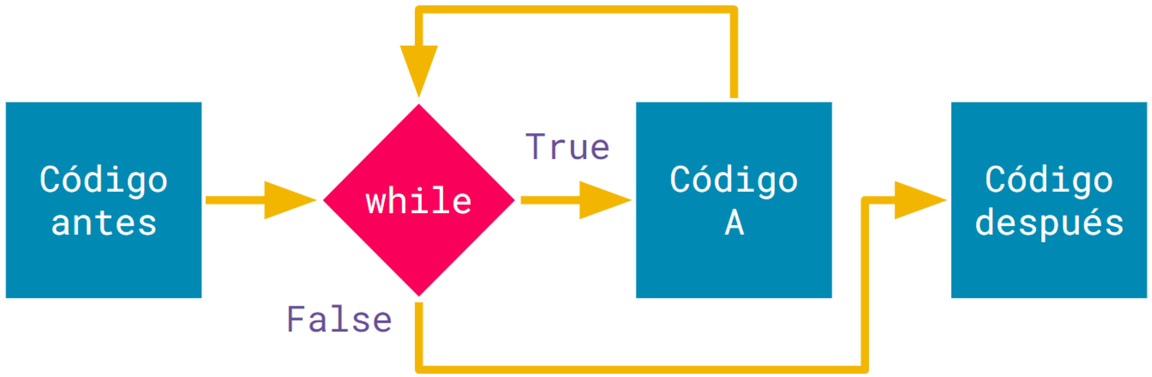
- Recibir una cantidad indefinida de *inputs*.
- Calcular algo hasta llegar a un resultado.
- Ejecutar operaciones hasta que se indique que el programa debe cerrarse.

Para ello, Python cuenta con dos comandos primitivos de ciclo:

1. `while` Permite ejecutar una serie de líneas mientras se cumpla una condición.
2. `for` Permite iterar sobre una secuencia.

Ciclos: `while`

La instrucción `while` permite ejecutar varias veces la misma sección de código.

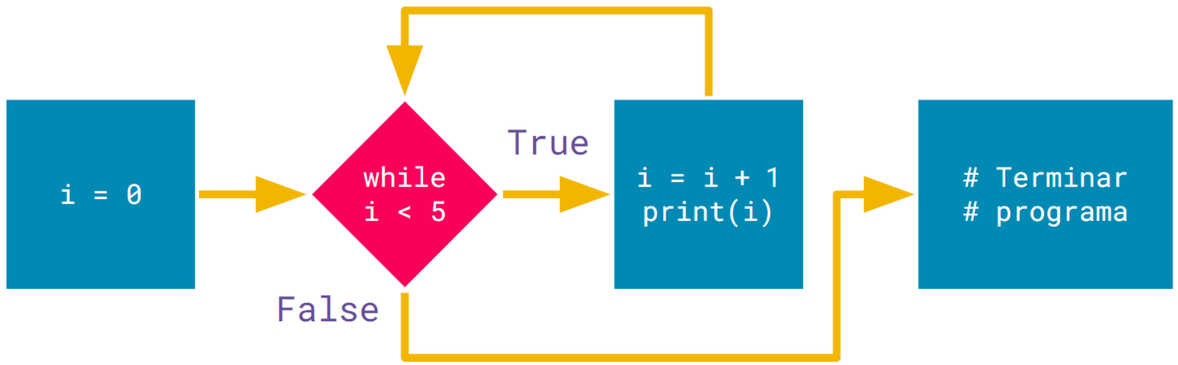



```
# Código antes
while condición:
    # Código A
    # …

# Código después
```

Importante: Necesitamos que el código modifique la condición para poder salir del ciclo, sino el código nunca terminará.

Ciclos: `while`

La instrucción `while` permite ejecutar varias veces la misma sección de código.




```
i = 0
while i < 5:      # Mientras i < 5
    i = i + 1     # Sumo 1 a i
    print(i)      # E imprimo
# Terminar programa
```

Ciclos: `while`

Ciclos: `while`

También podemos usar *strings* en la condición, por ejemplo, para trabajar sólo con texto.

```
nombre = input("Ingrese su nombre: ")

while nombre != "Pepa":
    print("Qué onda?! No eres Pepa!!!")
    nombre = input("Ingrese su nombre: ")

print("Que bueno que llegaste Pepa 💖")
```

Veamos un ejemplo

Funciones existentes: `import`

La instrucción `import`, nos permite usar código definido en otros archivos llamados módulos o librerías. Los módulos se cargan en memoria mediante importación.

```
import nombre_módulo

nombre_módulo.nombre_función(…)
```

```
from nombre_módulo import nombre_función

nombre_función(…)
```

Funciones existentes: `random`

Python cuenta con muchas librerías (en inglés *libraries*) que se pueden importar. Estas se denominan *built-in functions*.

Entre ellas vamos a aprender `random`, que nos entrega funciones que producen números aleatorios.

```
import random

print(random.random())       # float entre 0 y 1 (inclusive)
print(random.randint(1,2))   # int entre 1 y 2 (inclusive)
```

Funciones existentes: `random`

Python cuenta con muchas librerías (en inglés *libraries*) que se pueden importar. Estas se denominan *built-in functions*.

Entre ellas vamos a aprender `random`, que nos entrega funciones que producen números aleatorios.

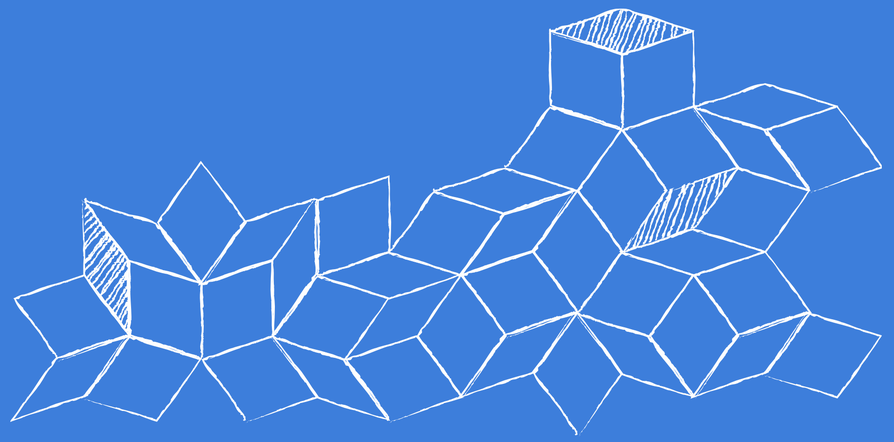
```
from random import random, randint

print(random())      # float entre 0 y 1 (inclusive)
print(randint(1,2)) # int entre 1 y 2 (inclusive)
```

Veamos un ejemplo

Vayamos a practicar la materia

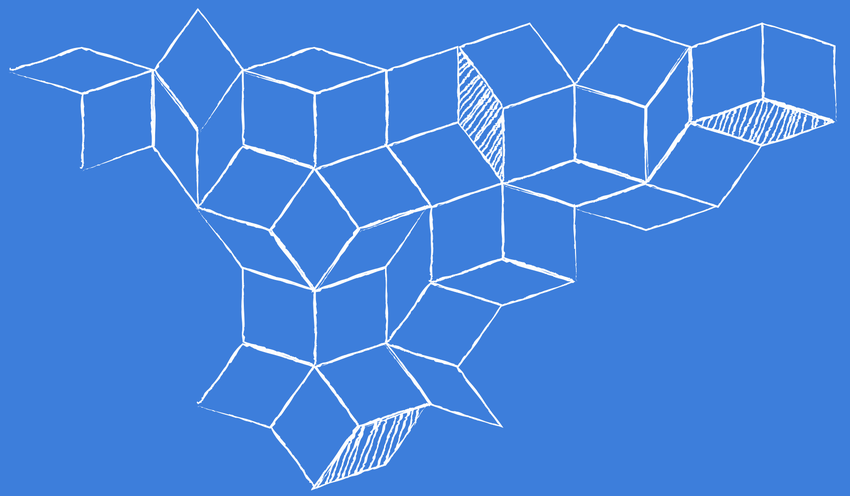
Problemas

- Has un cachipún que utilice ciclos.
  - Nivel 1: Se juega hasta que haya un ganador.
  - Nivel 2: A la tercera
- Lanzar dados.
  - Nivel 1: Hasta que el total sume 20 o más.
  - Nivel 2: Salga 2 veces seguida el mismo número.
- Adivina un número.
  - El computador tiene un número secreto y se pide *input* hasta que el usuario lo adivine.
- Escribe tu propia aventura (o la de Pepa 🐢)
  - Pídele acciones a un usuario hasta que “acabe el día” o el protagonista se vaya a dormir.



STEMUC

Academia para **escolares**



Suma MAT UC

precingeniería