UNIVERSIDAD
TECNOLÓGICA NACIONAL
FACULTAD REGIONAL
RESISTENCIA

LABORATORIO DEL LENGUAJE PYTHON

MÓDULO 2

GUÍA
TEÓRICA

CONDICIONALES Y BUCLES

python™

ALGORITMOS Y ESTRUCTURAS DE DATOS

GUÍA TEÓRICA

ESTRUCTURAS CONDICIONALES E ITERATIVAS PYTHON

Laboratorio de lenguajes de programación

2.Estructuras de Control

2.1 Sentencias Condicionales (if / elif / else)

Las sentencias condicionales permiten ejecutar bloques de código cuando se cumple una condición lógica. Python utiliza `if` (si) , `else` (sino) y `elif` (else if - sino si abreviado) para manejar decisiones.

Sintaxis general:

En Pseudocódigo se utiliza la estructura:

```
SI condición ENTONCES
    acciones
SINO
    otras acciones
FIN SI
```

O se anidan condicionales para expresar SINO SI con nuevas condiciones

En Python se utiliza:

```
if condición:
    # bloque verdadero
elif otra_condición:
    # otro bloque
else:
    # bloque por defecto
```

En vez de la palabra clave **entonces** se usan dos puntos (:). No hay sentencias de finalización de la estructura (fin si), se indica con la i**ndentación.** Si hay un bloque de código a la misma altura que donde comenzó ***if,*** esa parte ya no está en el bloque condicional.

Pseudocódigo

```
Acción VerificarNumero Es
     AMBIENTE
          numero : Entero;

     PROCESO
          numero := 10;
          SI numero > 0 Entonces
               Escribir("El número es positivo");
          FinSi;
FinAccion.
```

Python

```python
numero = 10

if numero > 0:

    print("El número es positivo")
```

if + else: dos caminos posibles

Pseudocódigo

```
Accion VerificarEdad Es
     AMBIENTE
          edad : Entero;
     PROCESO
          edad := 16;
          SI edad ⩾ 18 Entonces
               Escribir("Es mayor de edad");
          Sino
               Escribir("Es menor de edad");
          FinSi;
FinAccion.
```

Python

```python
edad = 16

if edad >= 18:

    print("Es mayor de edad")

else:

    print("Es menor de edad")
```

if + elif + else: múltiples condiciones posibles

Pseudocódigo

```
Accion EvaluarNota Es
     AMBIENTE
          nota : Entero;
     PROCESO
          nota := 7;
          SI nota ⩾ 9 Entonces
               Escribir("Excelente");
          Sino
               SI nota ⩾ 6 Entonces
                    Escribir("Aprobado");
               Sino
                    Escribir("Desaprobado");
               FinSi;
          FinSi;
FinAccion.
```

Python

```python
nota = 7

if nota >= 9:

    print("Excelente")

elif nota >= 6:

    print("Aprobado")
```

```
else:

    print("Desaprobado")
```

2.2.Estructuras de control Iterativas - Bucles

Bucle while

El bucle while repite un bloque de código mientras se cumpla una condición. Es útil cuando no sabemos cuántas veces se repetirá.

Es análogo a la estructura **mientras** en Pseudocódigo:

```
MIENTRAS condición HACER
   acciones
FIN MIENTRAS
```

En Python se utiliza:

```
while condición:

    # bloque de instrucciones
```

Se repite mientras una condición sea verdadera.

```
contador = 0
while contador < 3:
    print("Hola")
    contador += 1
```

Pseudocódigo

```
Accion numero_primo es
Ambiente
     N, i: entero
     Primo: Logico
Proceso
     esc('Ingrese un número');
     Leer(N);
```


```
      Primo:= verdadero;
      i:= 2;
      Mientras i ⩽ (N/2) y Primo Entonces
            Si (N mod i)= 0 entonces
                  Primo = falso;
            FinSi;
      FinMientras;
      Si Primo entonces
            esc('El número es primo') ;
      Sino
            esc('El número no es primo') ;
      FinSi;
FinAccion.
```

Python

```python
# Acción: número_primo

# Entrada de datos
N = int(input("Ingrese un número: "))

# Inicialización de variables
Primo = True
i = 2

# Proceso
while i <= N / 2 and Primo:
    if N % i == 0:
        Primo = False
    i += 1

# Salida
if Primo:
    print("El número es primo")
else:
    print("El número no es primo")
```

Tampoco hay sentencias de cierre. En este caso la sentencia print no esta dentro del bucle a estar a la misma altura que la declaración de while. Por lo que el bloque del bucle while es hasta `contador += 1`

```
contador = 0
while contador < 3:
    print("Hola")
    contador += 1
print("fin")
```

Bucle for

El bucle `for` es análogo al PARA. Se usa cuando sabemos cuántas veces debe repetirse.

En Pseudocódigo se utiliza la estructura:

```
PARA variable DESDE inicio HASTA fin HACER
   acciones
FIN PARA
```

Se usa cuando sabemos cuántas veces queremos repetir algo.

```
for i in range(5):
    print(i)
```

Una funcionalidad del bucle **for** en Python, que no está disponible en pseudocodigo, es la de recorrer **estructuras de datos que sean iterables.** Por ejemplo, los **strings** en python se pueden recorrer de carácter a carácter también.

```
for variable in iterable:

    # bloque de instrucciones
```

```
name = Nicolas
for letter in name:

      print(letter, \n)

    # muestra caracter a caracter las letras del nombre Nicola
```

ℹ️ La sentencia **"/n"** en el print indica **salto de línea,** se muestra cada print uno debajo del otro.

En una estructura de **lista** (aún no lo vemos, pero permite definir una colección de elementos que tienen un orden según su posición - llamado **índice).** En cada iteración la variable del for (animal) toma el valor de uno de los elementos de la lista, siguiendo el orden en que fueron declarados

```
animales = ["gato", "perro", "conejo"]

for animal in animales:

    print(animal)
```

En cada iteración el índice no representa un número sino cada elemento de la lista. En este ejemplo imprimirá los nombres de cada animal en la lista.

Más ejemplos:

Pseudocódigo

```
Accion Ejercicio_1_1_17 Es
    AMBIENTE
        A, B, SUMA, i : ENTERO
    PROCESO
        SUMA := 0
       Escribir("Este programa calcula el producto de dos enteros
usando solo sumas.")
       Escribir("Ingrese el primer número: ")
       Leer(A)
       Escribir("Ingrese el segundo número: ")
       Leer(B)

        // Usar valor absoluto de B para repetir la suma
        Para i := 1 Hasta ABS(B) Hacer
            SUMA := SUMA + A
        FinPara

        // Ajustar el signo si B es negativo
        Si B < 0 Entonces
            SUMA := -SUMA
        FinSi

        Escribir("El producto es igual a: ", SUMA)
FinAccion
```


Python

```python
# Este programa calcula el producto de dos enteros usando solo sumas

# Entrada de datos
A = int(input("Ingrese el primer número: "))
B = int(input("Ingrese el segundo número: "))

# Inicialización
SUMA = 0

# Tomamos el valor absoluto de B para repetir la suma
for i in range(abs(B)):
    SUMA += A

# Ajustamos el signo si B era negativo
if B < 0:
    SUMA = -SUMA

# Salida
print("El producto es igual a:", SUMA)
```

2.3 Break y continue

⚠️estas sentencias no existen en pseudocodigo

- break: termina el bucle anticipadamente, ni bien se encuentra con la sentencia, sin importar lo que hay debajo
- continue: salta al siguiente ciclo del bucle, no ejecuta las líneas de código después de la sentencia, pero no corta el bucle, continúa otra iteración.

En pseudocódigo

Se simula usando **condiciones dentro del bucle** para evitar seguir ejecutando.

```
bandera := VERDADERO;

MIENTRAS bandera HACER

    SI condicion ENTONCES

        bandera := FALSO;
```

```
    FIN SI;

FIN MIENTRAS;
```

En Python

El break y continue sirven para controlar el flujo de bucles de forma anticipada

break:

```python
for i in range(5):
    if i == 3:
        break
    print(i)
```

Este código dará como resultado:

```
PS C:\Users\natas\OneDrive\Documentos\Python_Lab> & 

0
1
2
PS C:\Users\natas\OneDrive\Documentos\Python_Lab>
```

continue:

```python
for i in range(5):
    if i == 2:
        continue
    print(i)
```

continue:

```
PS C:\Users\natas\OneDrive\Documentos\Python_Lab> & 

0
1
3
4
PS C:\Users\natas\OneDrive\Documentos\Python_Lab>
```

💻Proba ejecutar los ejemplos en el google colab
3_Condicionales e Iterativas Python

3. EXTRAS

Para incrementar un contador o acumulador se puede abreviar la operación

`c = c + 1` es lo mismo que `c += 1`

---

si queres investigar sobre una sentencia especial más → el **pass**

The pass Statement: How to Do Nothing in Python
Pass vs. Continue in Python Explained | Built In

📚Recursos:

LIBRO muy recomendado de la editorial O'Reilly →
https://drive.google.com/file/d/1UoxD_LwgGPx4ON3FOvJaCaA7r57n4r5B/view?usp=drive_link
Ejemplos de otro libro de Python de O'reilly https://allendowney.github.io/ThinkPython/
Contiene Google Colabs con ejemplos de cada capítulo del libro (mucho más complejo y extenso, pero pueden verlo, tomamos muchos ejemplos de allí para el colab)

*La editorial O'reilly es muy utilizada en el área y contiene material de calidad y altamente recomendable para toda su carrera, cuanto antes se acostumbren a utilizar **libros dedicados a desarrollo** verán una mejora significativa en su desempeño y se podrán adaptar mejor a los años superiores.*
*Eso si… estan todos en **inglés, ya conocen nuestro consejo sobre lo necesario que es el inglés en nuestra industria tanto para aprender como para trabajar***

otros links de cursos con otros ejemplos:
https://j2logo.com/python/tutorial/
https://developers.google.com/edu/python?hl=es-419
https://www.w3schools.com/python/default.asp
https://www.kaggle.com/learn/python