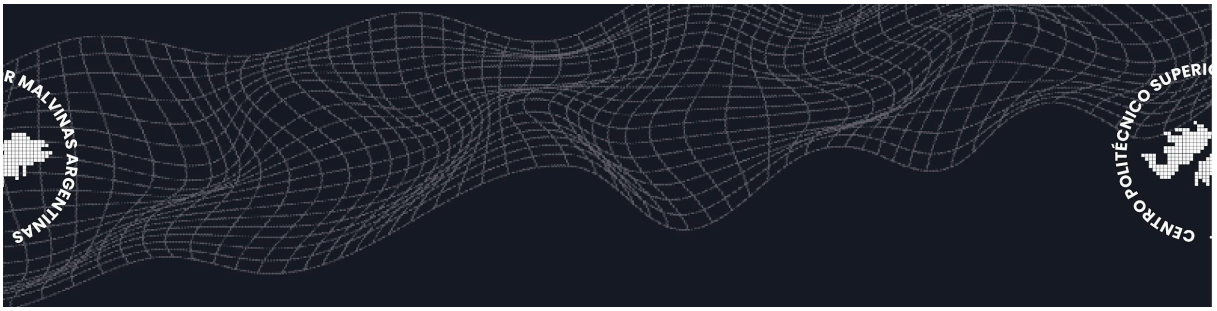


Clase 5: Flujo de control II - Estructuras repetitivas

Contenido

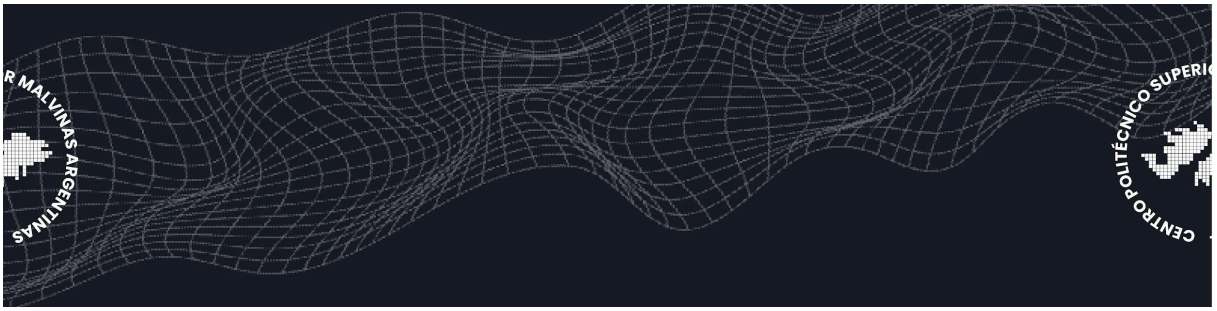
- Estructuras repetitivas,
- Estructura Mientras (while),
- Estructura Hacer mientras (do-while), Diferencias entre mientras (while) y hacer-mientras (do-while), Estructura repetir (repeat), Estructura desde/para (for), Salidas internas de los bucles, Sentencias de salto interrumpir (break) y continuar (continue).
-

1. Presentación:

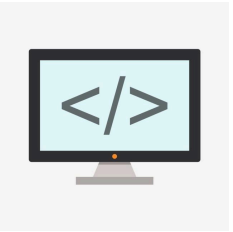
¡Bienvenidos al apasionante mundo de las estructuras repetitivas en Python!

Aquí, descubrirás cómo optimizar tus programas utilizando bucles "for" y "while". Aprenderás a automatizar tareas al ejecutar instrucciones una y otra vez, según condiciones específicas. Estas estructuras te permitirán tomar decisiones informadas y procesar datos de manera eficiente, llevando tus habilidades de programación al siguiente nivel. ¡Prepárate para dominar el arte de la repetición y la eficiencia en Python!


(queens):
i in range(BOARD_SIZE):
test_queens = queens
try:
validate(test_queens)
if len(test_queens)
return test_queens
else:
return



2. Desarrollo:



Estructuras repetitivas

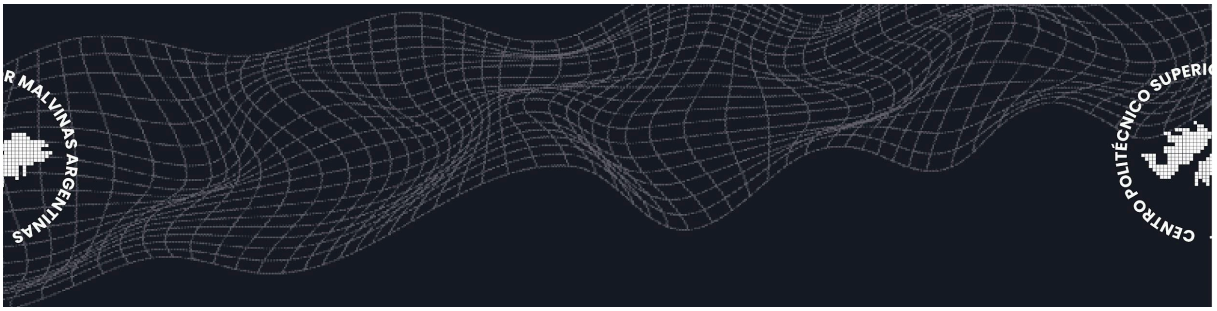
Un concepto importante en programación son las "estructuras repetitivas", también conocidas como bucles o ciclos. Estas estructuras permiten que un bloque de código se ejecute múltiples veces, lo que es esencial para automatizar tareas y procesar conjuntos de datos de manera eficiente. Hay dos tipos principales de estructuras repetitivas: el bucle "for" y el bucle "while".

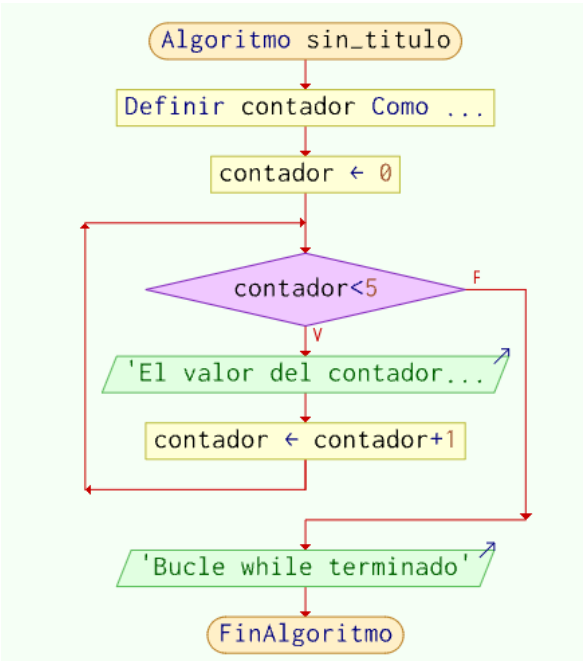
Estructura mientras (while)

while es una estructura de control que permite ejecutar un bloque de código repetidamente mientras se cumpla una

condición específica. Básicamente, el bloque de código se ejecutará una y otra vez hasta que la condición dada sea evaluada como falsa. A continuación un concepto más detallado sobre el bucle while en Python:

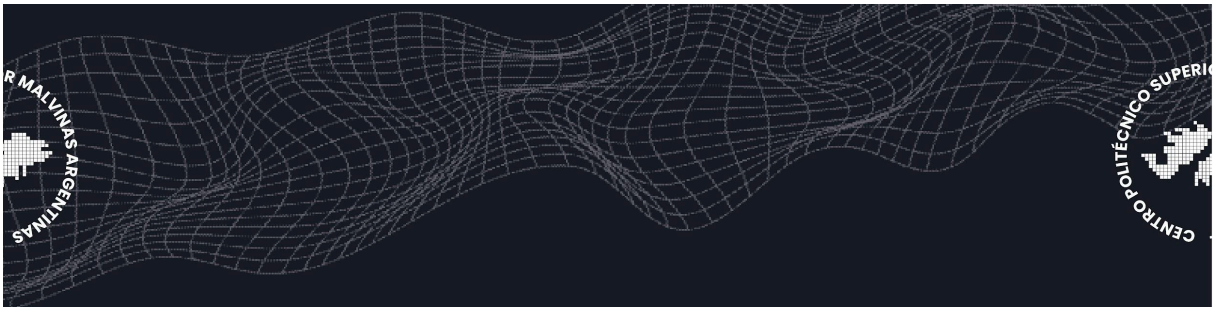
```python
contador = 0

while contador < 5:
    print("El valor del contador es:", contador)
    contador += 1

print("Bucle while terminado")
```





Estructura hacer-mientras (do-while)

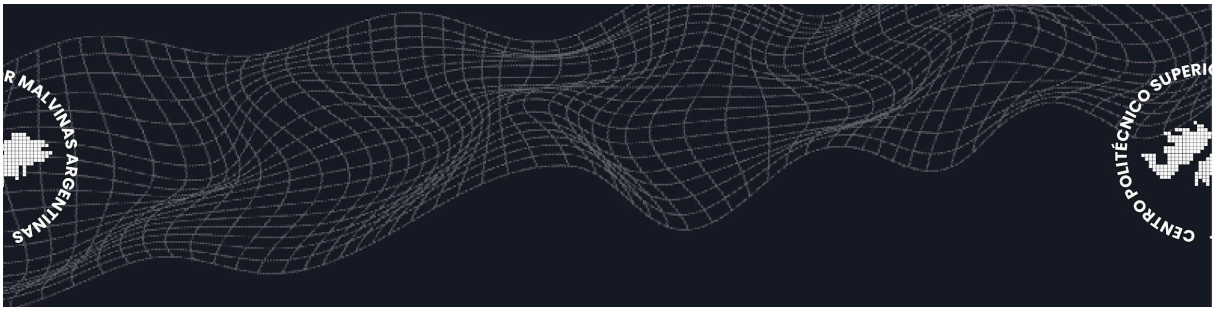
En Python, no hay una construcción de bucle do-while como en algunos otros lenguajes de programación. Sin embargo, puedes lograr el mismo efecto utilizando un bucle while junto con una condición de finalización al final(lo veremos posteriormente) del bucle. Aquí hay un ejemplo de cómo puedes hacerlo:



```python
while True:
    # Código que se ejecutará al menos una vez
    # ...

    # Verificar la condición de salida
    respuesta = input("¿Desea continuar? (s/n): ")
    if respuesta.lower() != 's':
        break  # Salir del bucle si la respuesta no es 's'
```

Diferencia entre while y do-while

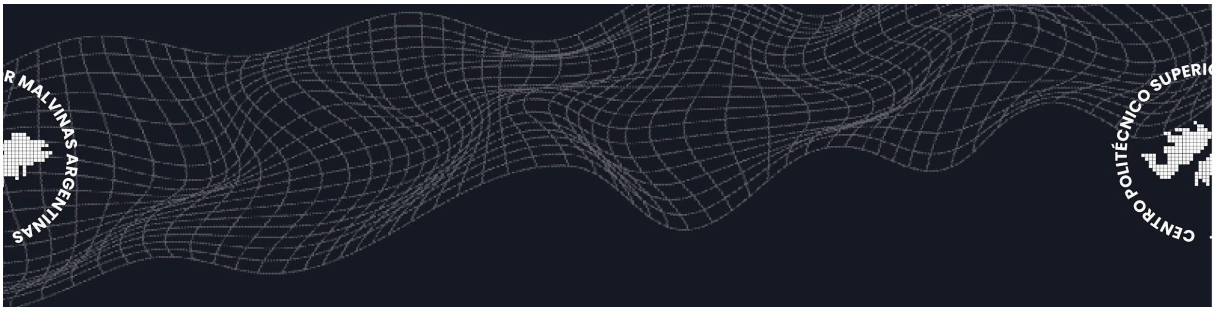
La principal diferencia es cuándo se verifica la condición: en el caso de while, se verifica antes de cada ejecución del bloque de código, mientras que en el caso de do-while, se verifica después de cada ejecución del bloque de código. Esto puede tener un impacto en cómo se comporta el bucle, especialmente si se necesita asegurar que el bloque de código se ejecute al menos una vez.

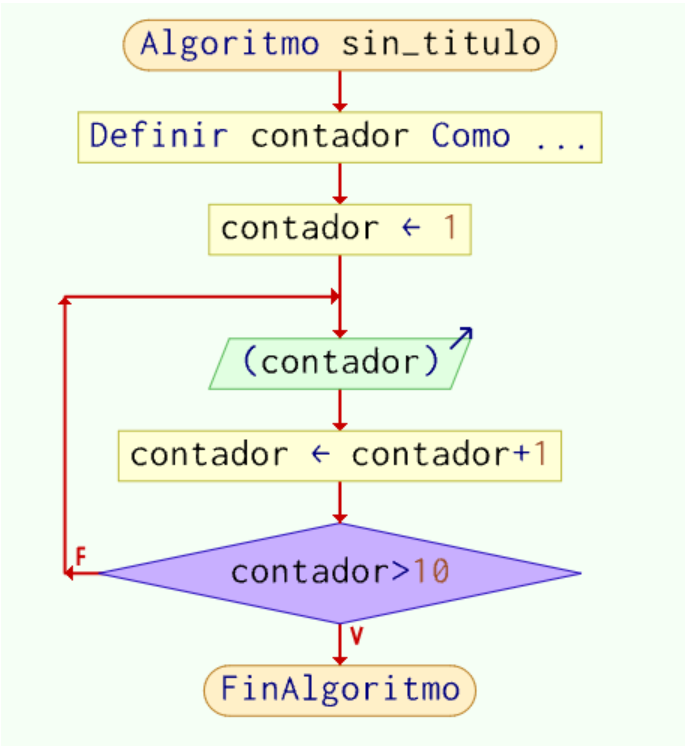


Estructura repetir (repeat)

repeat en programación se refiere a una estructura o comando que permite ejecutar un bloque de código o instrucciones múltiples veces de manera iterativa. También puede ser conocido como bucle o ciclo. La idea detrás del uso de "repeat" es automatizar la ejecución repetida de un conjunto de acciones sin tener que escribir el mismo código una y otra vez.

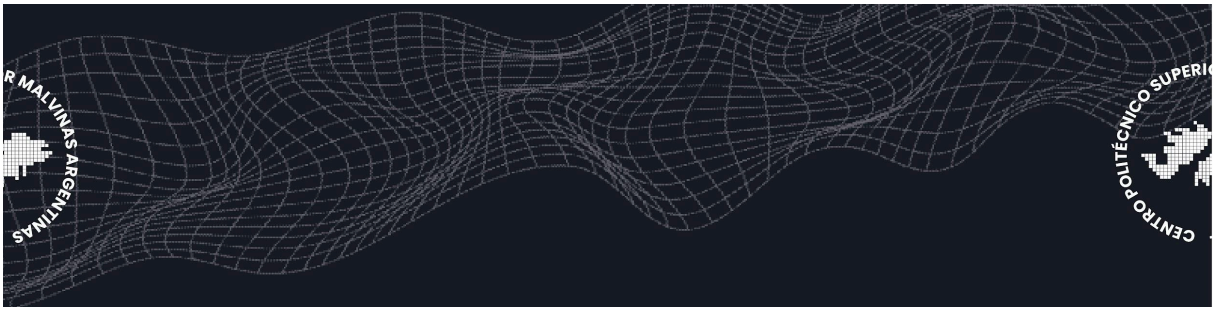
En muchos lenguajes de programación, existen diferentes tipos de bucles, como el "for", el "while" y el "do-while", que son ejemplos de cómo se puede implementar la funcionalidad de "repeat". Estos bucles permiten controlar la cantidad de repeticiones y la condición de continuación, lo que proporciona flexibilidad para adaptar el comportamiento del programa según sea necesario.

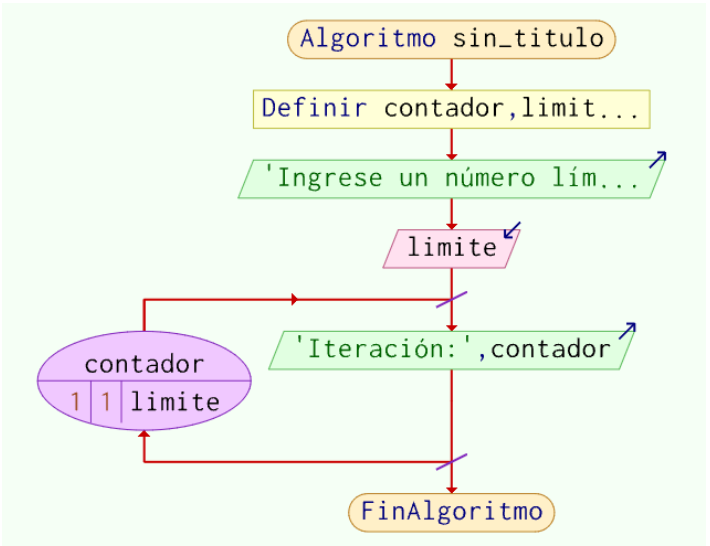




Estructura desde/para (for)

Un bucle for es una estructura de control utilizada en programación para repetir un bloque de código un número específico de veces. Este bucle es especialmente útil cuando se sabe de antemano cuántas veces se debe ejecutar un conjunto de instrucciones.

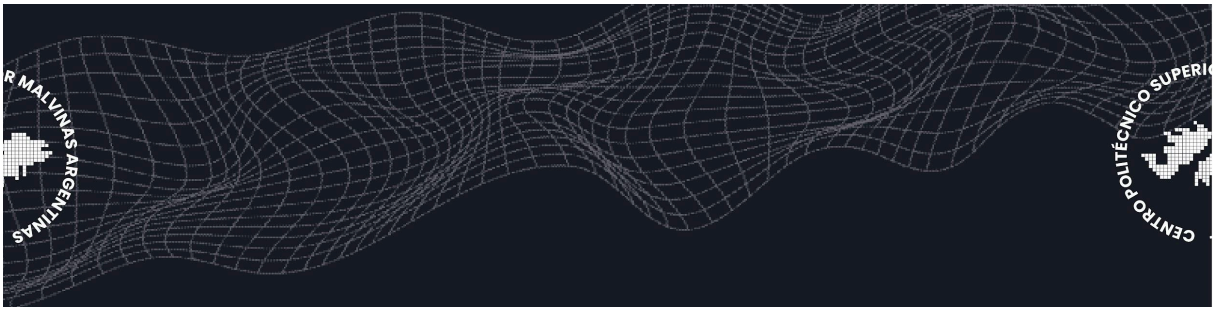





```python
limite = int(input("Ingrese un número límite: "))

for contador in range(1, limite + 1):
    print("Iteración:", contador)
```

Salidas internas de los bucles

Las "salidas internas de los bucles" en programación se refieren a condiciones o instrucciones que permiten salir anticipadamente de un bucle antes de que se haya completado su iteración total. Los bucles son estructuras

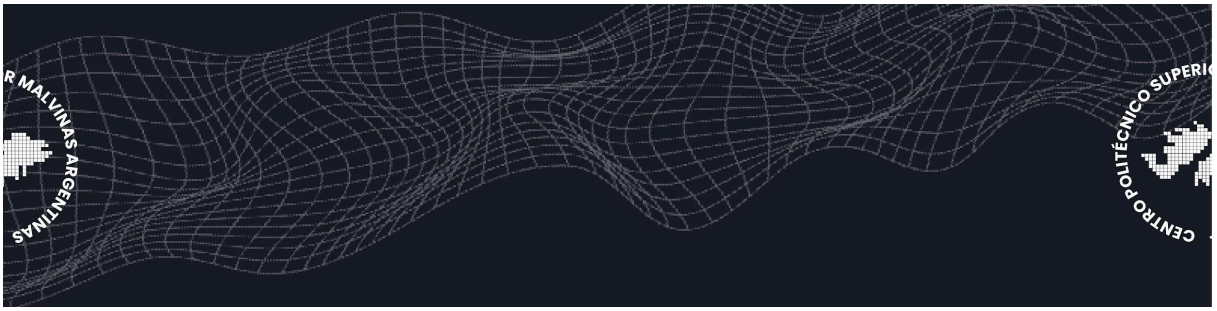


fundamentales en la programación que permiten repetir un bloque de código mientras se cumpla una condición específica.

Hay varias situaciones en las que puede ser útil salir de un bucle antes de que alcance su final natural. Las "salidas internas de los bucles" se utilizan para optimizar el rendimiento del programa, mejorar la legibilidad del código o para manejar situaciones específicas.

Sentencia break

En Python, al igual que en muchos otros lenguajes de programación, la sentencia "break" se utiliza para interrumpir la ejecución de un bucle (como "for" o "while") antes de que la condición de finalización normal se cumpla. Cuando se encuentra una sentencia "break" dentro de un bucle, el bucle se detiene inmediatamente y la ejecución del programa continúa con la primera instrucción después del bucle.



Ejemplo de cómo se usa la sentencia "break" en un bucle "while" en Python:

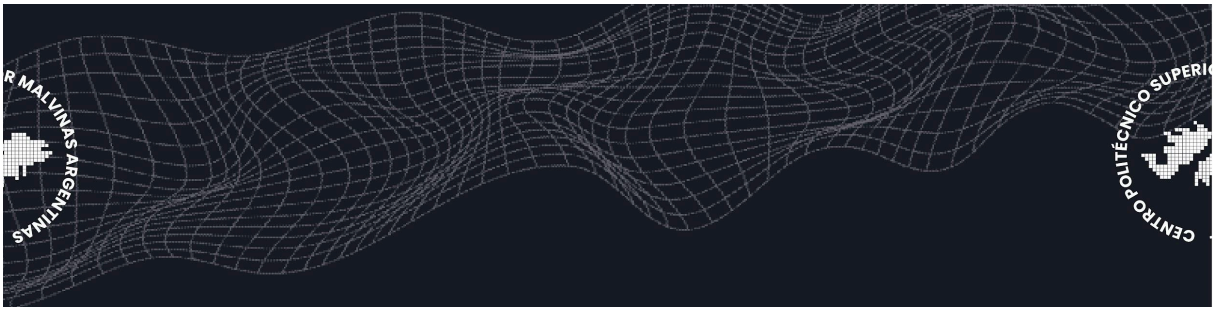
```python
i = 0
while i < 10:
    print(i)
    if i == 5:
        break
    i += 1
```

En este ejemplo, el bucle "while" imprimirá los números del 0 al 5 y luego se interrumpirá cuando "i" sea igual a 5 debido a la sentencia "break".

Sentencia continue

En Python, la sentencia continue es una instrucción que se utiliza dentro de bucles (como for y while) para controlar el flujo del programa. Cuando se ejecuta la instrucción continue, se omite el resto del código dentro de la iteración actual del bucle y se pasa a la siguiente iteración.

En otras palabras, si se encuentra la sentencia continue en medio de un bucle, el programa dejará de ejecutar el código



que sigue a esa sentencia en esa iteración en particular y pasará directamente a la siguiente iteración del bucle.

Aquí tienes un ejemplo para ilustrar cómo funciona la sentencia continue en un bucle for:

```python
for i in range(1, 6):
    if i == 3:
        print("Saltando la iteración", i)
        continue
    print("Iteración", i)
```

En este ejemplo, cuando i es igual a 3, la sentencia continue se ejecuta y salta la impresión de "Iteración 3", pasando directamente a la siguiente iteración. Como resultado, la línea que imprime "Saltando la iteración 3" se ejecuta solo cuando i es igual a 3, y el bucle continúa con las demás iteraciones.

3. Actividades

A continuación deberás utilizar el pensamiento computacional y resolver los siguientes ejercicios:

1- Validación de entrada de usuario:

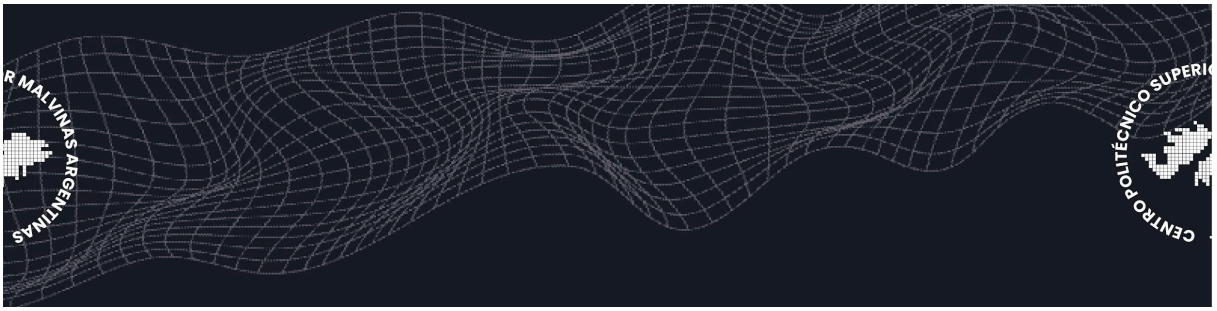
Solicitar al usuario que ingrese un número entre 1 y 100, y luego asegurate de que la entrada esté dentro del rango deseado.

2- Conteo regresivo:

Imprimir números en orden descendente desde un número inicial hasta cero.

3- Recorrido de una cadena de texto:

Contar cuántas veces aparece una letra específica en una cadena de texto.



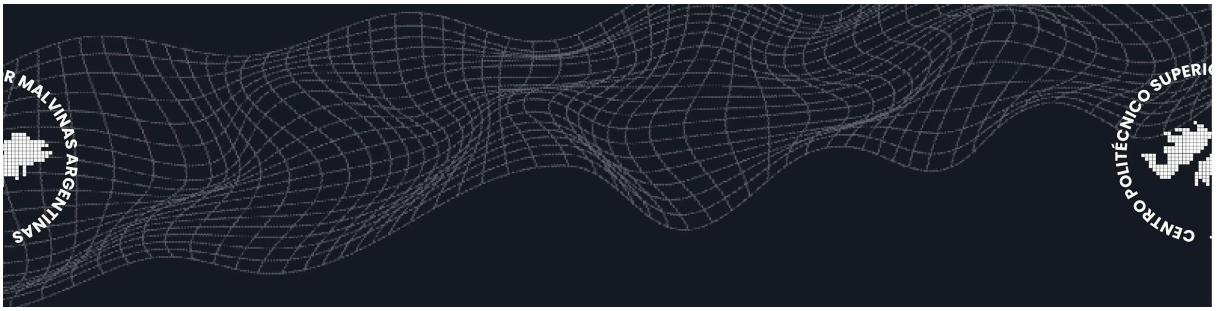
4. Cierre

En esta clase, nos enfocamos en las estructuras repetitivas: Estructura Mientras (while), Estructura Hacer mientras (do-while), Diferencias entre mientras (while) y hacer-mientras (do-while), Estructura repetir (repeat), Estructura desde/para (for), Salidas internas de los bucles, Sentencias de salto interrumpir (break) y continuar (continue).

Material de ayuda:

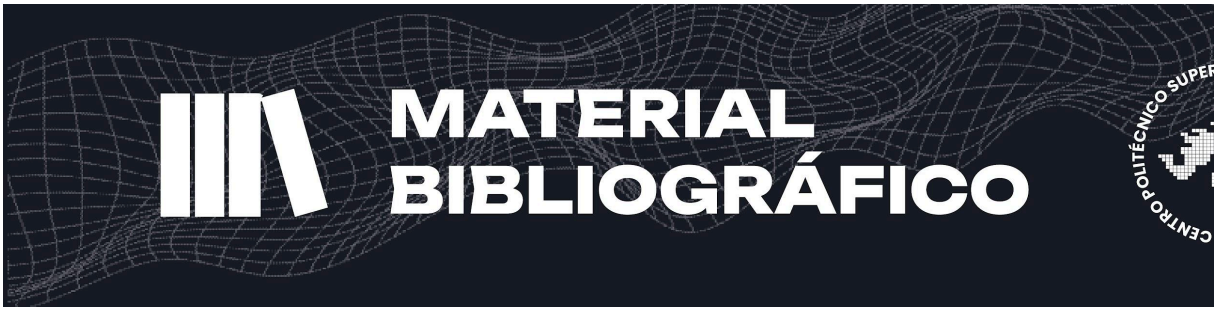
En el siguiente video tutorial se explica el uso de while con python(ProgramacionATS):

https://www.youtube.com/watch?v=YEWxlbffgxE



En el siguiente video tutorial se explica el uso de for con python(ProgramacionATS):

https://www.youtube.com/watch?v=mRI8C2ZhDkg

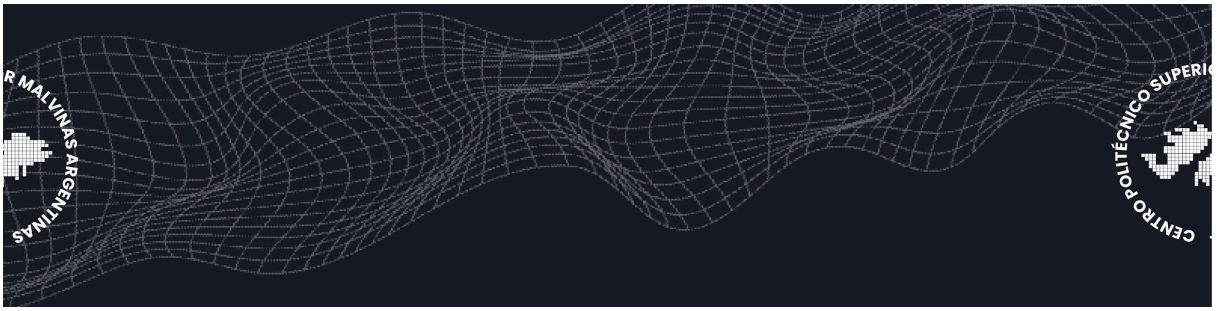


Bibliografía obligatoria

- FUNDAMENTOS DE PROGRAMACIÓN. Algoritmos, estructura de datos y objetos-Cuarta edición - Luis Joyanes Aguilar-McGrawHill - 2008 - ISBN 978-84-481-6111-8

Bibliografía sugerida de la unidad

- Python 3 – Los fundamentos del lenguaje - Es un libro completo, bien escrito, claro, aunque un poco extenso, por lo que no es apto para lectores apurados. La intención del autor es brindar absolutamente todo lo que se necesita
-



para que el lector aprenda a programar en Python desde cero.

- Python para todos: explorando datos en Python 3 - Este libro es ideal para estudiantes y programadores que buscan especializarse en la exploración y el análisis de datos.