	Institución Educativa José Eusebio Caro **Tecnología e Informática**
Docentes	Jesús Eduardo Madroñero Ruales
Propósito del taller	Identificar un conjunto de pasos e instrucciones para realizar una tarea. Crear programas con la comprensión de múltiples estrategias para implementar ciclos. Codificar los ciclos y/o bucles del lenguaje hablado, en algoritmos.
Competencias	Utilizo adecuadamente herramientas informáticas de uso común para la búsqueda y procesamiento de la información y la comunicación de ideas.

Los Ciclos en Programación

Un **ciclo** (o bucle), es una estructura de control que permite ejecutar un conjunto de instrucciones varias veces. Los ciclos son una forma eficiente de automatizar tareas repetitivas en un programa.

Tipos de ciclos en programación

(1) Ciclo "para" (for): Esta estructura repite un conjunto de instrucciones un número específico de veces. Es ideal para:

- Repetir una acción un número determinado de veces.
- Recorrer listas o rangos de números.
- Realizar acumulaciones o búsquedas sistemáticas.

Ejemplo: Este programa muestra los números: del 1 al 5.

Ejemplo en PseInt

```
Algoritmo Ciclopara
Definir i como entero
Para i ← 1 Hasta 5
  Escribir "número: ", i
FinPara
FinAlgoritmo
```

Ejemplo en Python

```
for i in range(1, 6):
    print("Número:", i)
```

(2) Ciclo "mientras" (while): Esta estructura repite un conjunto de instrucciones, mientras se cumpla una determinada condición. La condición se evalúa al principio de cada iteración, y si es verdadera, es ejecuta el cuerpo del ciclo. Si es falsa, el ciclo termina.

Ejemplo: Este programa muestra los números del 1 al 5.

Ejemplo en PseInt

```
Algoritmo Ciclomientras
Definir i como entero
i ← 1
Mientras i <= 5
  Escribir "número: ", i
FinMientras
FinAlgoritmo
```

Ejemplo en Python

```
i = 1
while i <= 5:
    print(i)
    i += 1
```

(3) Ciclo "repetir" (repeat): Esta estructura de control que permite ejecutar un bloque de instrucciones mientras se cumpla una condición determinada, pero a diferencia de otros ciclos como "mientras" o "para", primero se ejecuta el bloque de código y luego se evalúa la condición. Es decir, siempre se ejecuta al menos una vez.

Ejemplo: Este programa solicita un número entero, y genera su tabla de multiplicar del 1 al 10.

Ejemplo en PseInt	Ejemplo en Python

Proceso TablaMultiplicar
  Definir numero, i Como Entero
  i <- 1
  Escribir "Ingrese un número:"
  Leer numero
  Repetir
  Escribir numero, " x ", i, " = ", numero * i
  i <- i + 1
  Hasta Que i > 10
FinProceso
	numero = int(input("Ingrese un número: "))
	i = 1
	while True:
	  print(f"{numero} x {i} = {numero * i}")
	  i += 1
	  if i > 10:
	  break

Los ciclos son una herramienta fundamental en programación, ya que permiten automatizar tareas repetitivas y hacer más eficiente el código. Sin embargo, es importante utilizarlos con cuidado para evitar **bucles infinitos** o **errores de lógica**.

A continuación, se presenta una tabla comparativa entre los tres tipos de ciclos: Mientras, Repetir y Para.

Característica	Ciclo Mientras (while)	Ciclo Repetir (repeat)	Ciclo Para (for)
Evaluación de la condición	Al inicio del ciclo	Al final del ciclo	Al inicio del ciclo
Número de repeticiones	Desconocido	Desconocido (mínimo 1 vez)	Conocido
Se ejecuta al menos una vez	No necesariamente	Sí, siempre al menos una vez	No necesariamente
Control de variable de ciclo	Manual	Manual	Automático
Uso común	Repetir mientras una condición sea cierta	Repetir hasta que una condición se cumpla	Recorrer rangos o listas

Actividad conceptual

1. En su cuaderno realizar un resumen de lo descrito en el presente documento.
2. Explicar con ejemplos, las características de la tabla comparativa, entre los ciclos: **para**, **mientras** y **repetir**.
3. Consultar la sintaxis de la implementación de los ciclos **para**, **mientras** y **repetir**, para PseInt y Python.
4. Describir en sus propias palabras las diferencias claves entre los ciclos: **para**, **mientras** y **repetir.**
5. Teniendo en cuenta las estructuras de control (**para, mientras** y **repetir**), describir el algoritmo que resuelva los siguientes problemas:
   a. Pronunciar los números pares del 0 al 20.
   b. Correr 20 vueltas en la periferia de una cancha de futbol.
   c. Medir la estatura de 45 estudiantes de un salón de clase.

Actividad de codificación

Seguir las siguientes instrucciones para resolver las situaciones:

- Leer cada enunciado.
- Para cada situación, detallar los siguientes ítems: **variables y estructuras de control**: **condicionales y ciclos**.
- Escribir el código correspondiente en PseInt y Python.
- Responder las preguntas de reflexión al final.
- Utilizar la plataforma adjunta: https://psintplus.lat/, para codificar y ejecutar los algoritmos desarrollados.

Problemas propuestos:

1. Desarrollar un algoritmo que: solicite al usuario un número e imprima su tabla de multiplicar del 1 al 10.
2. Desarrollar un algoritmo que solicite 10 números al usuario y debe contar cuantos números positivos y negativos ingresa.
3. Desarrollar un algoritmo que solicite un número entero, y que, si el número es par presente los primeros 20 números pares, y si el número es impar, debe presentar los primeros 20 números impares.