UIDE Powered by Arizona State University®	**Universidad Internacional del Ecuador** **Escuela de Ciencias de la Computación** Carrera Ingeniería en Sistemas de Información	ASU

Nombre: Wendy Nayeli Pañora Guevara	**Trabajo:** Aprendizaje Autónomo P2
Materia: Logitud de programación	**Semestre:** Primero
Fecha: 03/07/2025	**Profesor: ESTEFANIA VANESSA HEREDIA JIMENEZ**

Juego de Piedra, Papel y tijeras

1. Menú de opciones
2. Validación de entradas
3. Lógica del juego
4. Posibilidad de jugar múltiples rondas

Líneas de códigos utilizados

1. import random
   - Es una librería (o módulo) estándar de Python que proporciona funciones para generar números pseudoaleatorios y realizar selecciones al azar.
     ¿Para qué sirve la librería random?
     Principalmente para:
   - Generar números aleatorios.
   - Mezclar listas (barajar).
   - Seleccionar elementos al azar de una secuencia.
     Importa el módulo random para generar elecciones aleatorias de la computadora.

2. lista= [“Piedra”, “Papel”, “Tijera”]
   - En Python, las listas son una de las estructuras de datos más versátiles y utilizadas. Sirven para almacenar múltiples elementos en una sola variable
   - ¿Para qué sirven las listas?

Almacenar colecciones de datos (números, textos, objetos, etc.).
Mantener un orden (cada elemento tiene un índice numérico, empezando desde 0).
Modificar datos (agregar, eliminar o cambiar elementos).
Iterar sobre ellos (usando bucles como for).
Realizar operaciones complejas (ordenar, filtrar, buscar, etc.).

- Crea una lista con las tres opciones posibles del juego.

3. print ("Bienvenidos al juego de Piedra, Papel o Tijeras")
    - Muestra un mensaje de bienvenida al juego.

4. while True:

    En Python es un bucle infinito que se ejecuta indefinidamente hasta que se encuentre una instrucción break o se interrumpa el programa manualmente.

    ¿Para qué sirve?

    - Se usa principalmente en situaciones donde:
    - No sabes cuántas veces se repetirá el bucle (ej: hasta que el usuario ingrese un valor válido).
    - Necesitas mantener un proceso activo (ej: un servidor, un menú interactivo).
    - Quieres controlar manualmente la salida con break.

    Inicia un bucle infinito que mantendrá el juego en ejecución hasta que el usuario decida salir.

5. print ("Opciones a elegir:")
   print ("(1) Piedra")
   print ("(2) Papel")
   print ("(3) Tijera")
   print ("(4) Salir del juego")
    - Muestra el menú de opciones al jugador.

6. ```
   try:
       jugador = int (input ("Seleccione una opción: "))
   except ValueError:
       print("Por favor, Ingrese un número válido")
       continue
   ```
   ```

- El try en Python es parte de la estructura try-except, que se utiliza para manejar errores (excepciones) en tiempo de ejecución. Su objetivo es evitar que el programa se detenga abruptamente cuando ocurra un error y, en su lugar, permitirte manejarlo de forma controlada.
- Intenta convertir la entrada del usuario a un número entero. Si falla (porque el usuario ingresó texto), muestra un mensaje de error y vuelve al inicio del bucle.

Los condicionales if, else y elif en Python son estructuras fundamentales para controlar el flujo de un programa, permitiendo ejecutar diferentes bloques de código según se cumplan (o no) ciertas condiciones.

7. if jugador == 4:
   print("Fin del juego, gracias por partipar.¡Adiós!")
   break
   - Si el usuario elige 4 (Salir), muestra un mensaje de despedida y termina el juego.

8. if jugador not in [1, 2, 3]:
   print("La opción seleccionada es invalida, por favor escoge una opción válida")
   continue
   - Verifica que la opción seleccionada sea válida (1, 2 o 3). Si no lo es, muestra un mensaje de error y vuelve al inicio del bucle

9. jugador_opcion = lista[jugador-1]
   - Convierte la opción numérica del jugador (1-3) en la cadena correspondiente ("Piedra", "Papel" o "Tijera").

10. computador_opcion = random.choice(lista)
    - random.choice() es una función de la librería random de Python que selecciona un elemento aleatorio de una secuencia (como una lista, tupla o string). Es útil cuando necesitas elegir un valor al azar de un conjunto de opciones.

    - La computadora elige aleatoriamente una de las tres opciones.

11. print("Opción del jugador: ", jugador_opcion)
    print("Opción del computador: ", computador_opcion)

- Muestra las elecciones de ambos jugadores.

12. if jugador_opcion == computador_opcion:
        print (“¡La partida es Empate!”)

- Si ambas elecciones son iguales, declara un empate.

13. elif (jugador_opcion == "Piedra" and computador_opcion == "Tijera") or \
        (jugador_opcion == "Papel" and computador_opcion == "Piedra") or \
        (jugador_opcion == "Tijera" and computador_opcion == "Papel"):
        print ("¡Ganaste la partida!")

- Operador and (Y lógico)
  Evalúa si todas las condiciones son verdaderas (True).
  Regla: True solo si ambas condiciones son True.
- Operador or (O lógico)
  Evalúa si al menos una condición es verdadera (True).
  Regla: True si cualquiera de las condiciones es True.

- Verifica las combinaciones ganadoras donde el jugador vence a la computadora.

14. else:
        print("¡Perdiste la partida!")

- Si no es empate ni victoria del jugador, entonces el jugador pierde.

15. while True:
        seguir = input("¿Quieres seguir jugando? (si/no): ").strip().lower()
        if seguir in ["si", "no"]:
            break
        else:
            print("Por favor, ingrese 'si' para seguir jugando o 'no' para terminar el juego")

- Pregunta al jugador si quiere continuar, validando que la respuesta sea "si" o "no".

16. if seguir! = 'si':
        print("Fin del juego, gracias por partipar.¡Adiós!")
        break

- Si el jugador no quiere continuar ("no"), muestra un mensaje de despedida y termina el juego.