PROYECTO INTEGRADOR

LÓGICA DE PROGRAMACIÓN

JUEGO PIEDRA PAPEL O TIJERA

Sebastián Rojas

Índice

OBJETIVO

Desarrollar un software funcional que permita aplicar conocimientos de programación estructurada, como condicionales, bucles y estructuras lógicas, a través de un juego interactivo.

DESCRIPCIÓN

El juego fue desarrollado en Python y permite que dos jugadores compitan eligiendo entre piedra, papel o tijera. El sistema decide el ganador según reglas tradicionales, guarda el puntaje y permite jugar múltiples rondas.

DIAGRAMA DE CASOS DE USO(UML):

Representa cómo los usuarios interactúan con el juego y qué funcionalidades existen (inicio, elección, resultado, reinicio).

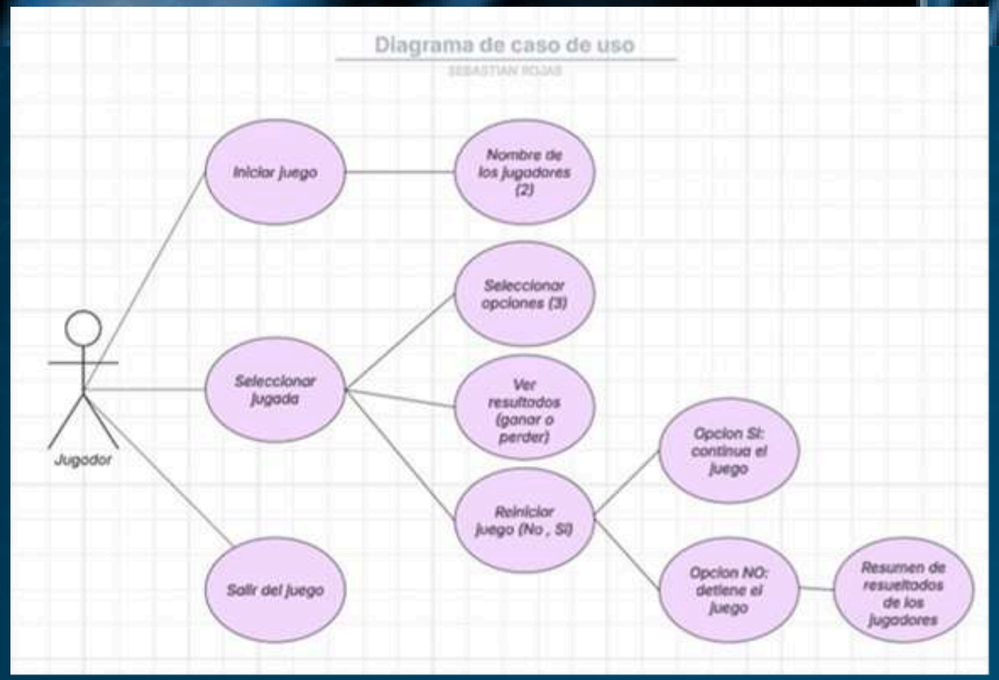
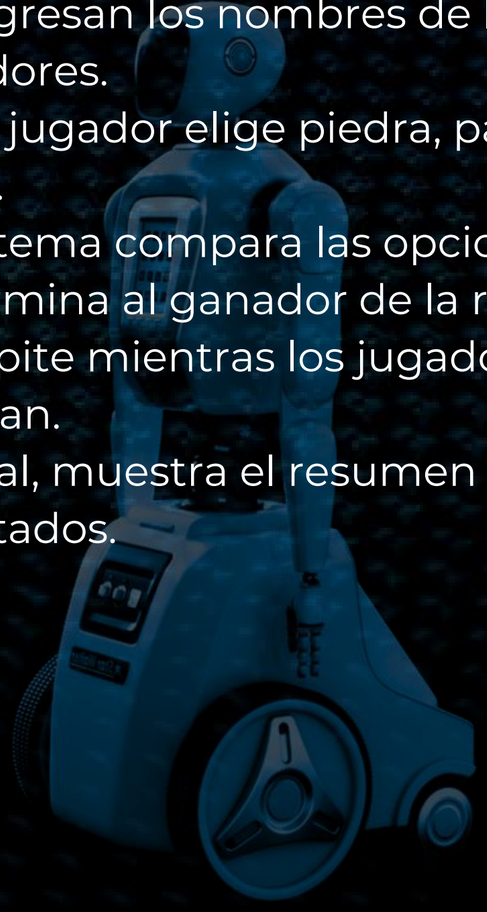


DIAGRAMA DE ARQUITECTURA POR CAPAS:

Organiza el programa en niveles, separando la lógica del juego de la interfaz del usuario.

¿Cómo funciona internamente?:

- Se ingresan los nombres de los jugadores.
- Cada jugador elige piedra, papel o tijera.
- El sistema compara las opciones y determina al ganador de la ronda.
- Se repite mientras los jugadores quieran.
- Al final, muestra el resumen de resultados.



```
Algoritmo PiedraPapelTijera
    Definir nombre1, nombre2, opcion1, opcion2, seguir Como Caracter
    Definir gana1, gana2, empates Como Entero
    gana1 ← 0
    gana2 ← 0
    empates ← 0
    Escribir 'Ingrese el nombre del jugador 1:'
    Leer nombre1
    Escribir 'Ingrese el nombre del jugador 2:'
    Leer nombre2
    Repetir
        Escribir nombre1, ', elige: Piedra, Papel o Tijera'
        Leer opcion1
        opcion1 ← Minusculas(opcion1)
        Mientras opcion1<>'piedra' Y opcion1<>'papel' Y opcion1<>'tijera' Hacer
            Escribir 'Opcion invalida. Intenta de nuevo:'
            Leer opcion1
            opcion1 ← Minusculas(opcion1)
        FinMientras
        Escribir nombre2, ', elige: Piedra, Papel o Tijera'
        Leer opcion2
        opcion2 ← Minusculas(opcion2)
        Mientras opcion2<>'piedra' Y opcion2<>'papel' Y opcion2<>'tijera' Hacer
            Escribir 'Opcion invalida. Intenta de nuevo:'
            Leer opcion2
            opcion2 ← Minusculas(opcion2)
        FinMientras
        Si opcion1=opcion2 Entonces
            Escribir 'Empate!'
            empates ← empates+1
        SiNo
            Si (opcion1='piedra' Y opcion2='tijera') O (opcion1='papel' Y opcion2='piedra') O (opcion1='tijera' Y opcion2='papel') Entonces
                Escribir nombre1, ' gana esta ronda!'
                gana1 ← gana1+1
            SiNo
                Escribir nombre2, ' gana esta ronda!'
                gana2 ← gana2+1
            FinSi
        FinSi
        Escribir '¿Quieren seguir jugando? (si/no)'
        Leer seguir
        seguir ← Minusculas(seguir)
    Hasta Que seguir<>'no'
    Escribir 'Resumen de resultados:'
    Escribir '-----------------------------------'
    Escribir nombre1, ' gano ', gana1, ' veces.'
    Escribir nombre2, ' gano ', gana2, ' veces.'
    Escribir 'Empates: ', empates
    Escribir '-----------------------------------'
    Si gana1>gana2 Entonces
        Escribir nombre1, ', ganaste, ¡felicidades!'
    SiNo
        Si gana2>gana1 Entonces
            Escribir nombre2, ', ganaste, ¡felicidades!'
        SiNo
            Escribir 'Hubo un empate general entre ', nombre1, ' y ', nombre2
        FinSi
    FinSi
FinAlgoritmo
```

ENTORNO Y HERRAMIENTAS USADAS

- **Lenguaje**: Python
- **Editor:** Visual Studio Code
- **Control de versiones:** GitHub
- **Colaboración:** Repositorio para documentar avances y código

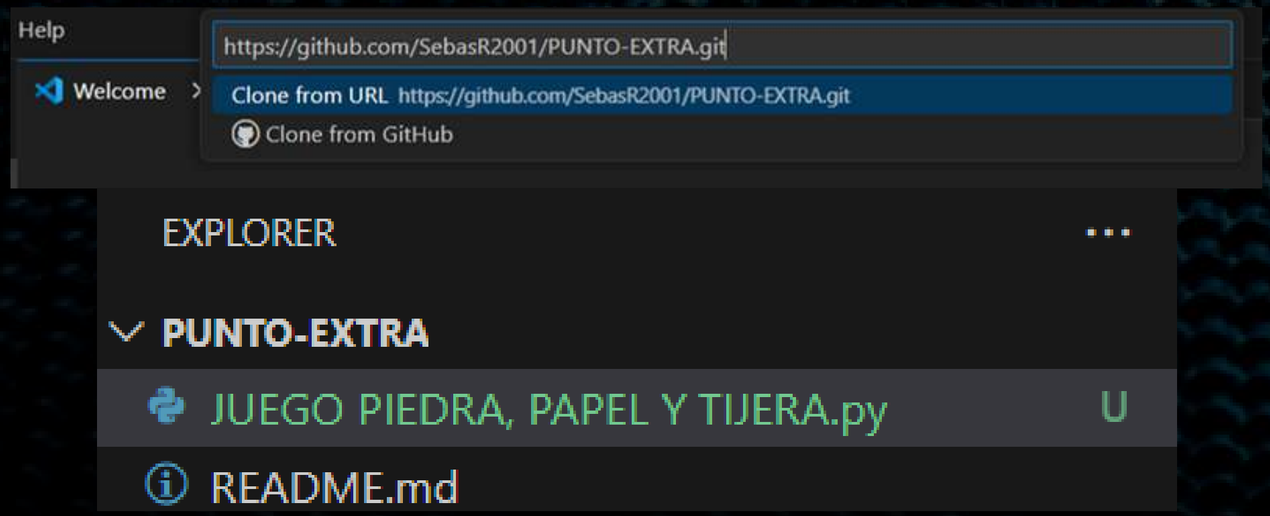


DIAGRAMA DE FLUJO

El diagrama representa la lógica del juego Piedra, Papel o Tijera. El programa inicia solicitando los nombres de los jugadores y entra en un ciclo que se repite mientras deseen continuar. En cada ronda, ambos jugadores ingresan su opción y el sistema valida las entradas. Luego compara las elecciones para determinar si hubo un empate o qué jugador ganó, actualizando los puntajes. Al finalizar, muestra un resumen de resultados y declara al ganador general. Si los jugadores desean seguir, el ciclo se repite; de lo contrario, el programa finaliza.

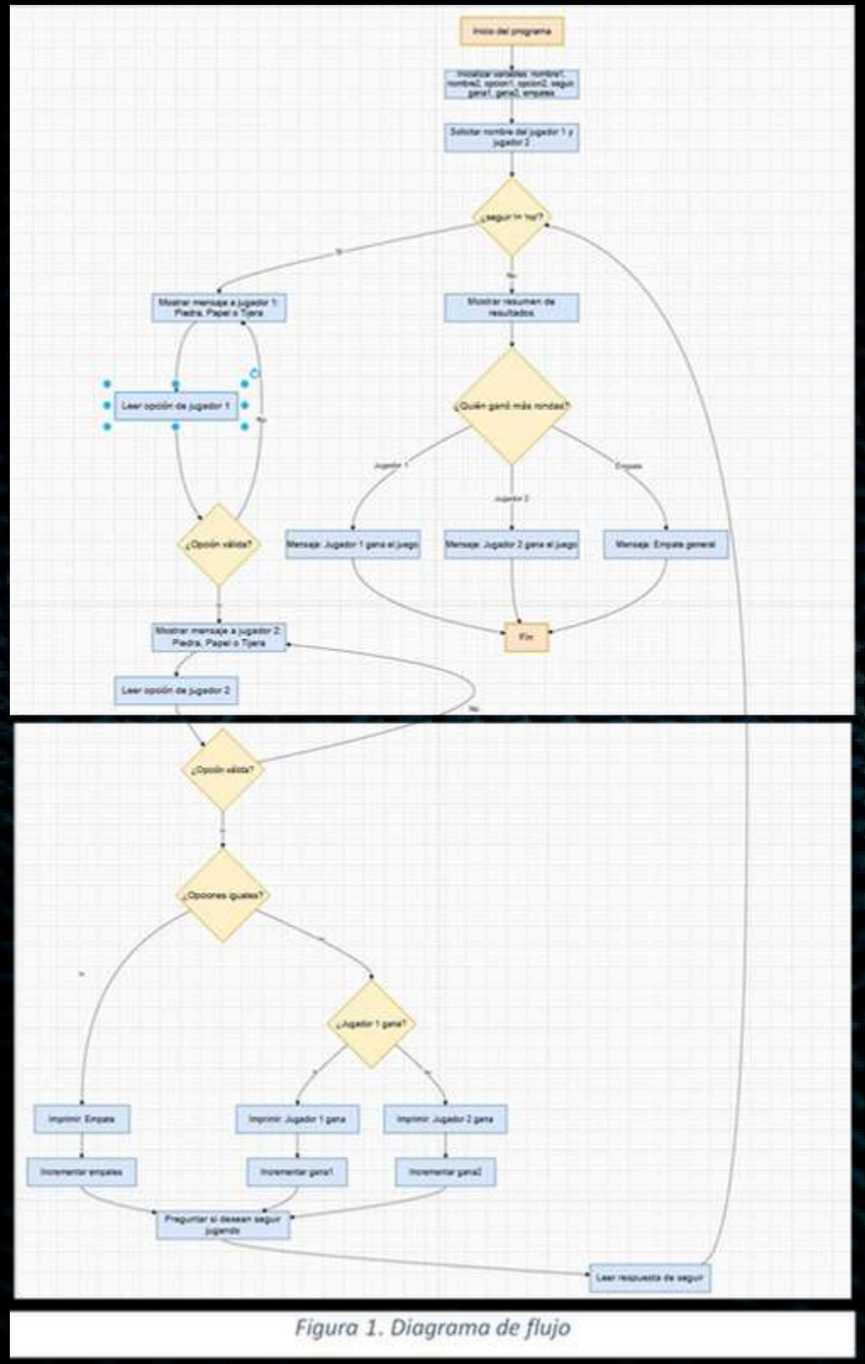


Figura 1. Diagrama de flujo

RESULTADOS Y CONCLUSIONES

Resultados obtenidos:

- Se creó un juego funcional, con estructura clara y reutilizable.
- Se aplicaron condicionales, bucles y comentarios para facilitar su comprensión.
- Se presentaron videos explicativos y funcionales como parte de los entregables.

Conclusión:

Este proyecto demuestra cómo una idea sencilla puede convertirse en una herramienta útil para aprender lógica de programación y desarrollar software funcional con impacto educativo.

```python
# Definir variables para almacenar nombres, opciones y resultados del juego
nombre1 = ""  # Nombre del jugador 1
nombre2 = ""  # Nombre del jugador 2
opción1 = ""  # Opción elegida por el jugador 1 (Piedra, Papel o Tijera)
opción2 = ""  # Opción elegida por el jugador 2 (Piedra, Papel o Tijera)
seguir = ""   # Variable para controlar si los jugadores desean seguir jugando
gana1 = 0     # Contador de victorias del jugador 1
gana2 = 0     # Contador de victorias del jugador 2
empates = 0   # Contador de empates

# Solicitar nombres de los jugadores
nombre1 = aporte("Ingrese el nombre del jugador 1: ")
nombre2 = aporte("Ingrese el nombre del jugador 2: ")

# Iniciar el juego y repetir mientras los jugadores deseen seguir jugando
mientras seguir != 'No':
    # Jugador 1 elige una opción (Piedra, Papel o Tijera)
    imprimir(nombre1 + ', elige: Piedra, Papel o Tijera')
    opción1 = aporte().más bajo()

    # Validar la opción elegida por el jugador 1
    mientras opción1 != 'piedra' y opción1 != 'papel' y opción1 != 'tijera':
        imprimir('Opción inválida. Intenta de nuevo:')
        opción1 = aporte().más bajo()

    # Jugador 2 elige una opción (Piedra, Papel o Tijera)
    imprimir(nombre2 + ', elige: Piedra, Papel o Tijera')
    opción2 = aporte().más bajo()

    # Validar la opción elegida por el jugador 2
    mientras opción2 != 'piedra' y opción2 != 'papel' y opción2 != 'tijera':
        imprimir('Opción inválida. Intenta de nuevo:')
        opción2 = aporte().más bajo()

    # Determinar el resultado de la ronda
    si opción1 == opción2:
        imprimir('¡Empate!')
        empates += 1  # Incrementar el contador de empates si las opciones son iguales
    elif(opción1 == 'piedra' y opción2 == 'tijera')o\
        (opción1 == 'papel' y opción2 == 'piedra')o\
        (opción1 == 'tijera' y opción2 == 'papel'):
        imprimir(nombre1 + '¡Gana esta ronda!')
        gana1 += 1  # Incrementar el contador de victorias del jugador 1 si gana
    demás:
        imprimir(nombre2 + '¡Gana esta ronda!')
        gana2 += 1  # Incrementar el contador de victorias del jugador 2 si gana

    # Preguntar a los jugadores si desean seguir jugando
    seguir = aporte('¿Quieres seguir jugando? (si/no)').más bajo()

# Mostrar resumen de resultados al finalizar el juego
imprimir(Resumen de resultados:)
imprimir('-----------------------------------')
imprimir(nombre1 + ' ganó ' + cadena(gana1)+ ' a veces.')
imprimir(nombre2 + ' ganó ' + cadena(gana2)+ ' a veces.')
imprimir('Empates:' + cadena(empates))
imprimir('-----------------------------------')

# Determinar y mostrar al ganador general del juego
si gana1 > gana2:
    imprimir(nombre1 + '¡Ganaste, felicidad!')
elif gana2 > gana1:
    imprimir(nombre2 + '¡Ganaste, felicidad!')
demás:
    imprimir('Hubo un empate general entre ' + nombre1 + ' y ' + nombre2)
```

THANK YOU