Presentación del Proyecto: Piedra, Papel o Tijera

PROYECTO DE PROGRAMACIÓN EN PYTHON

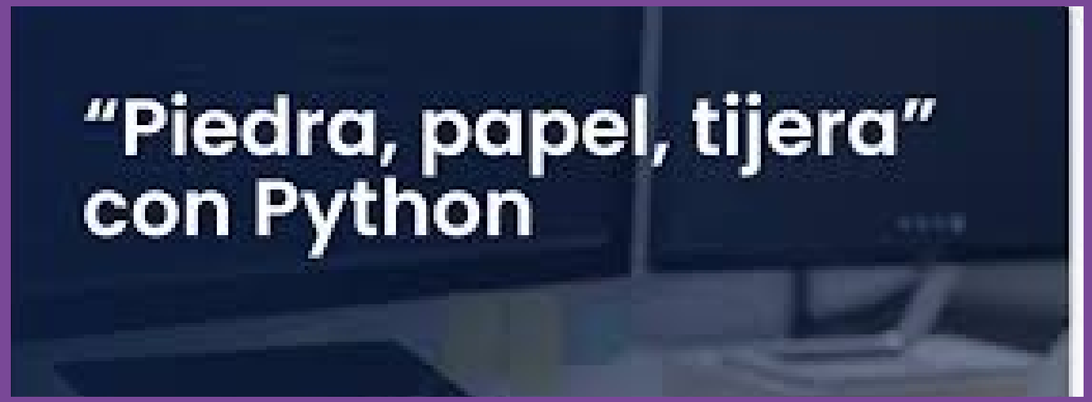
PRESENTADO POR : JHONNATAN SALAZAR

TABLA DE CONTENIDO

INTRODUCCIÓN

Objetivo del Proyecto:

- Diseñar un juego interactivo en Python que simule el juego de piedra, papel o tijera.
- Aplicar conceptos básicos de programación como estructuras de control, funciones y bucles.




```
# Define una función que genera y devuelve
def obtener_eleccion_computadora():
    opciones = ["piedra", "papel", "tijera
    return random.choice(opciones)
```

HERRAMIENTAS UTILIZADAS

En este proyecto utilizamos Python con la librería random para las elecciones aleatorias, desarrollando en Visual Studio Code. Aplicamos conceptos clave como condicionales, bucles y funciones, apoyándonos en documentación oficial y recursos educativos de esta materia

LENGUAJE DE PROGRAMACIÓN

Python 3.13.1

ENTORNO DE DESARROLLO

Visual Studio Code
Facilitan la edición, ejecución y depuración del código.

LIBRERIA RANDOM

Libreria Random
Utilizada para las elecciones aleatorias

ARQUITECTURA DEL JUEGO

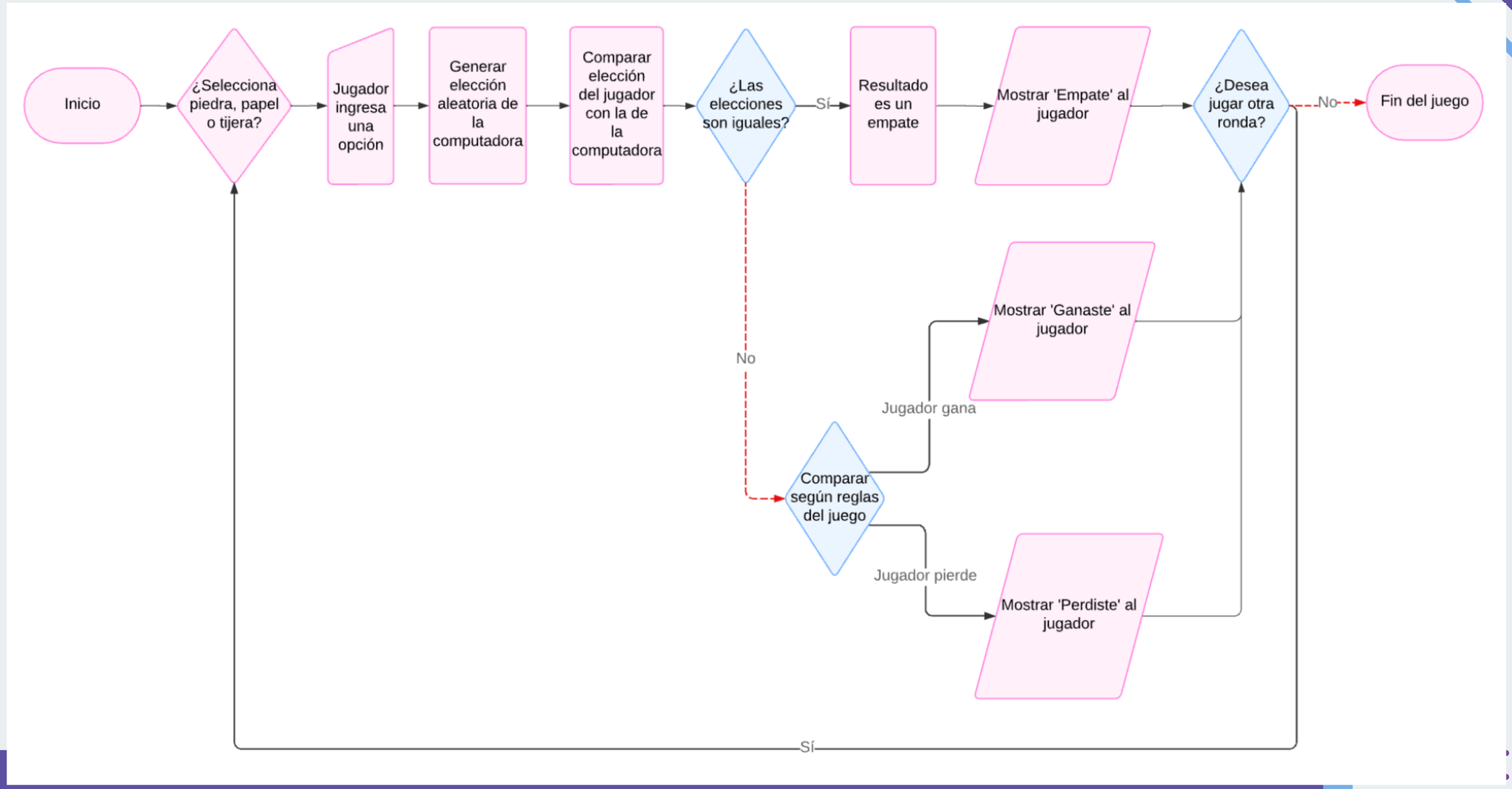
ELEMENTOS CLAVE

- Entrada del jugador (elección entre piedra, papel o tijera).
- Elección aleatoria de la computadora.
- Comparación de elecciones para determinar el resultado.
- Posibilidad de jugar varias rondas.

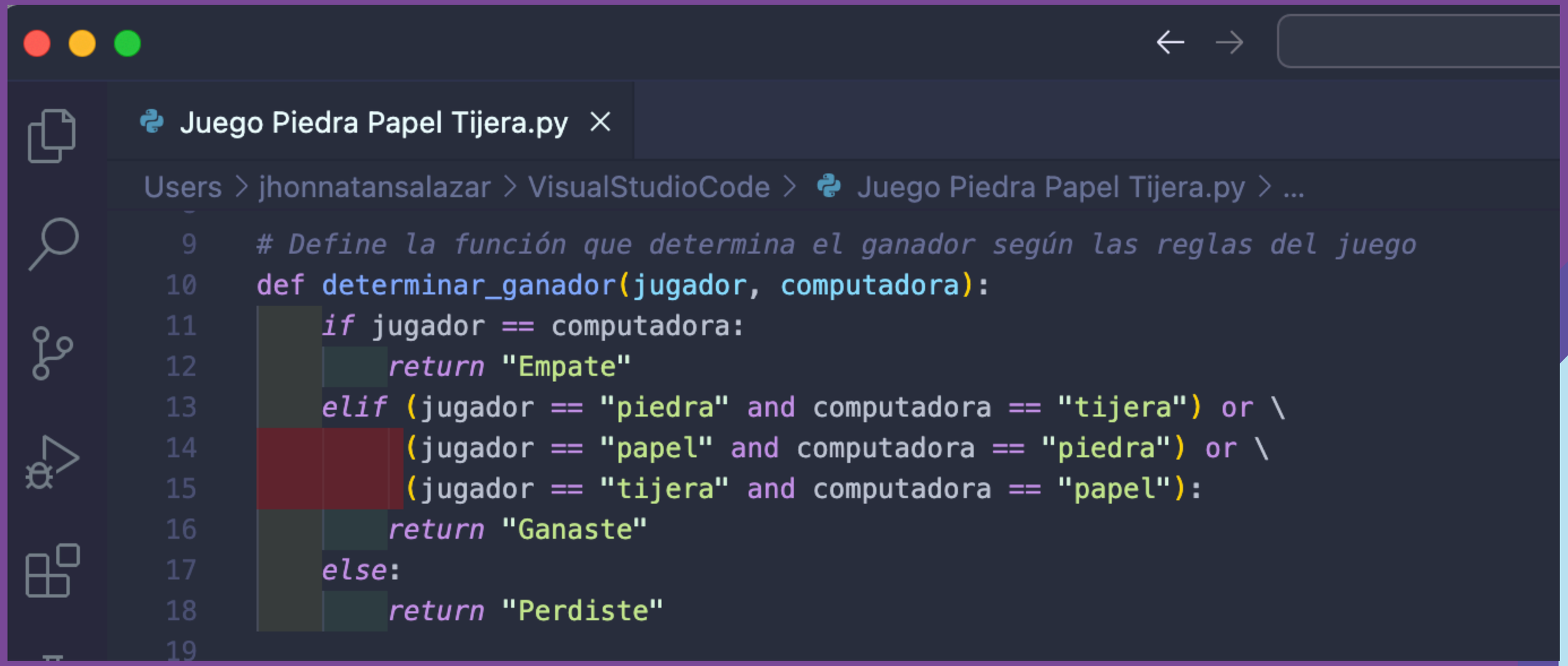
FUNCIONES PRINCIPALES

1. obtener_eleccion_computadora()
2. determinar_ganador(jugador, computadora)
3. jugar()

DIAGRAMA DE FLUJO



FRAGMENTO DE CÓDIGO



EJEMPLO DE EJECUCIÓN

En el ejemplo de ejecución, el usuario elige una opción como "piedra", mientras que la computadora genera aleatoriamente otra, como "tijera". Luego, el programa compara ambas elecciones y determina el resultado: en este caso, el usuario gana. Finalmente, se pregunta si desea jugar otra vez, mostrando un flujo simple y dinámico del juego en consola.

01
USUARIO = PIEDRA

02
PC = TIJERA

03
COMPARA REGLAS

04
RESULTADO = GANASTE

05
JUGAR OTRA VEZ

06
(SI / NO)

07
(JUEGA / SALE)

INTEGRACIÓN DE CONOCIMIENTO

USO DE OPERADORES RACIONALES

Los operadores racionales en el código son:
== (igual a): Compara si dos valores son iguales.

```
if jugador == computadora:
```

USO DE OPERADORES LÓGICOS

Se usan para combinar múltiples condiciones en una misma expresión

- and
- or

```
"piedra" and computadora == "tijera") or
"papel" and computadora == "piedra") or \
"tijera" and computadora == "papel"):
```

USO DE CONDICIONALES IF, ELIF, ELSE

Se utilizan para tomar decisiones basadas en las condiciones

```
if jugador == computadora:
    return "Empate"
elif (jugador == "piedra" and compu
     (jugador == "papel" and comput
     (jugador == "tijera" and compu
    return "Ganaste"
else:
    return "Perdiste"
```

BUCLE “WHILE”

Mantiene el juego en ejecución de forma continua hasta que el usuario indique que no quiere jugar más

```
while True:
    # Elección del usuario
    jugador = input("Elige piedra

    # Validar entrada del usuario
    if jugador not in ["piedra",
        print("Entrada no válida.
        continue
```

ESTRUCTURA DE DATOS “LISTAS”

Es una lista que contiene las tres opciones del juego: "piedra", "papel" y "tijera"

```
opciones = ["piedra", "papel", "tijera"]
```

CONCLUSIÓN Y FUTURO DEL PROYECTO

CONCLUSIÓN

- Este proyecto demostró cómo se pueden aplicar conceptos de programación para crear un juego divertido y funcional.
- Refleja cómo los conceptos básicos pueden ser la base para proyectos más complejos en el futuro.

MEJORAS FUTURAS

- Crear una versión con interfaz gráfica.
- Implementar un sistema de puntuación.

GRACIAS POR SU ATENCIÓN