

Presentación del Proyecto: Piedra, Papel o Tijera

PROYECTO DE PROGRAMACIÓN EN PYTHON

PRESENTADO POR : JHONNATAN SALAZAR



TABLA DE CONTENIDO



01 INTRODUCCIÓN

05 FRAGMENTO DEL
CÓDIGO

02 HERRAMIENTAS
UTILIZADAS

06 EJEMPLO DE EJECUCIÓN

03 ARQUITECTURA DEL
JUEGO

07 INTEGRACIÓN DE
CONOCIMIENTO

04 DIAGRAMA DE FUJO

08 CONCLUSIÓN Y FUTURO
DEL PROYECTO

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.

**“Piedra, papel, tijera”
con Python**

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

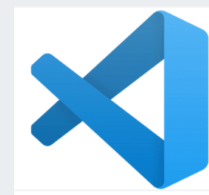
HERRAMIENTAS UTILIZADAS

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

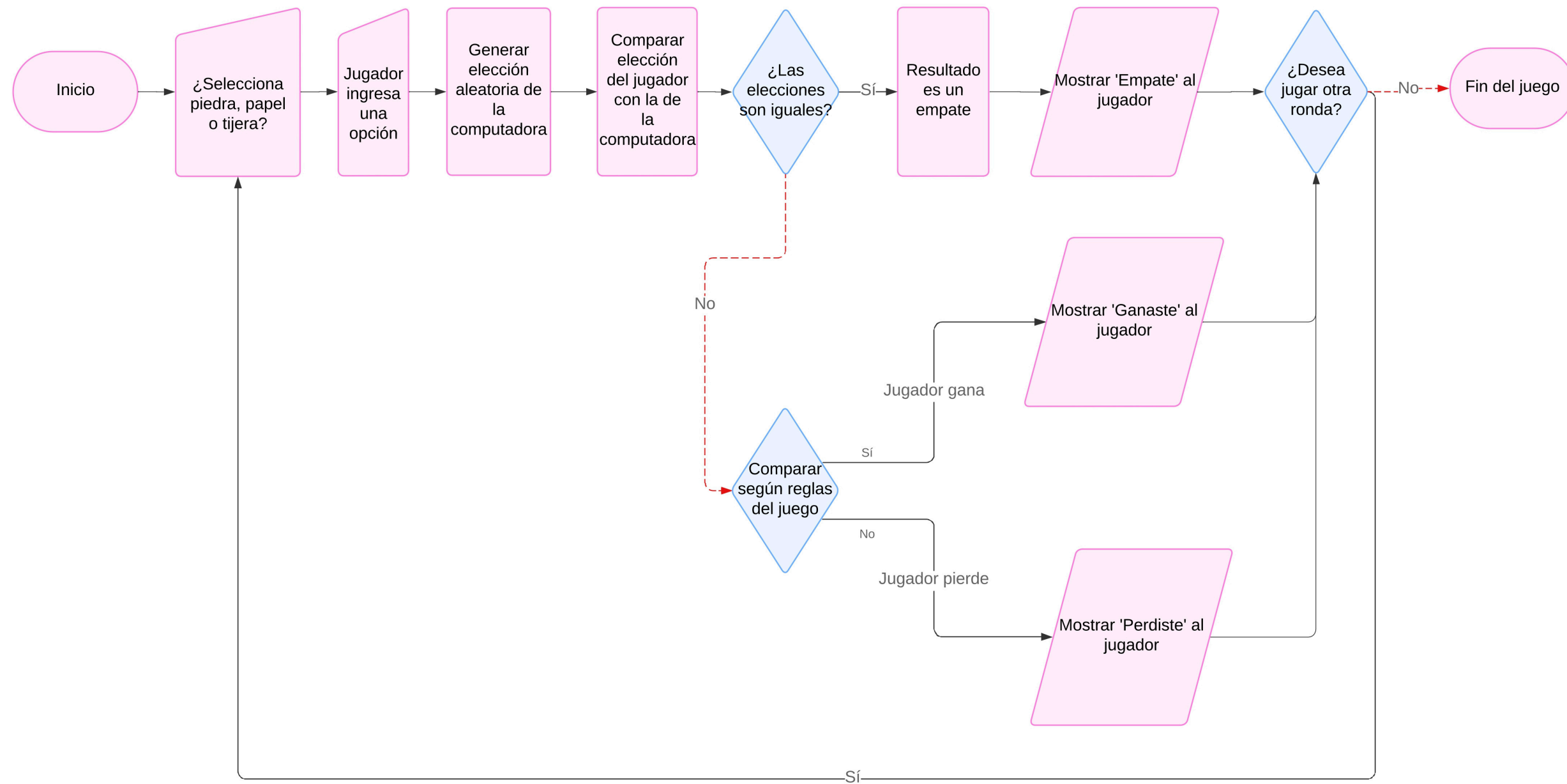
- 1.obtener_eleccion_computadora()
- 2.determinar_ganador(jugador, computadora)
- 3.jugar()



DIAGRAMA DE FLUJO

Diagrama de flujo

Jhonnatan Salazar | December 18, 2024



FRAGMENTO DE CÓDIGO

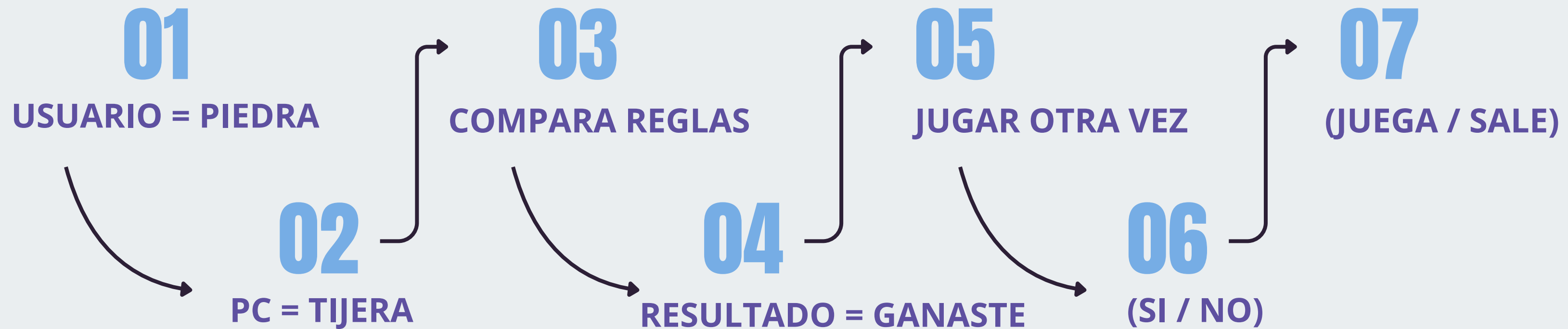
A screenshot of a code editor window with a dark theme. The window title is 'Juego Piedra Papel Tijera.py'. The breadcrumb path is 'Users > jhonnatansalazar > VisualStudioCode > Juego Piedra Papel Tijera.py > ...'. The code is in Python and defines a function 'determinar_ganador' that takes 'jugador' and 'computadora' as arguments. It uses conditional logic to determine the winner based on the rules of Rock Paper Scissors. The code is syntax-highlighted, with comments in grey, keywords in blue, and strings in yellow. Line numbers 9 through 19 are visible on the left side of the code block.

```
9  # Define la función que determina el ganador según las reglas del juego
10 def determinar_ganador(jugador, computadora):
11     if jugador == computadora:
12         return "Empate"
13     elif (jugador == "piedra" and computadora == "tijera") or \
14          (jugador == "papel" and computadora == "piedra") or \
15          (jugador == "tijera" and computadora == "papel"):
16         return "Ganaste"
17     else:
18         return "Perdiste"
19
```

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.



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.
- Multijugador





**GRACIAS
POR SU
ATENCIÓN**