

16 DE NOVIEMBRE DE 2025

EJERCICIO TRES EN RAYA PYTHON

ENRIQUE GONZÁLEZ



CAMPUSFP
1º DAM GETAFE

INDICE

Tres en raya (con funciones y módulos).....	2
Objetivo.....	2
Requisitos.....	2
Estructura mínima.....	2
APP.PY.....	3
CÓDIGO:.....	3
TTT_CORE.PY (FUNCIONES SALVO MENU).....	4
CÓDIGO:.....	4
COMPROBACIÓN:.....	8
VS JUGADOR.....	8
VS CPU:.....	10

Tres en raya (con funciones y módulos)

Objetivo

Desarrollar un juego de **Tres en raya (Tic-Tac-Toe)** en consola, usando:

- **Funciones** para separar responsabilidades (lógica del juego, entrada/salida).
- **Módulos** (archivos `.py`) para organizar el proyecto.
- **NumPy** para representar el tablero como una matriz 3×3.

Requisitos

1. El tablero será una matriz 3×3 de enteros:
 - 0 = casilla vacía, 1 = jugador X, 2 = jugador O.
2. Debe permitir **Jugador vs Jugador** y **Jugador vs CPU (aleatoria)**.
3. Validar entradas: filas/columnas entre 0 y 2 y casillas vacías.
4. Detectar **victoria** (filas, columnas, diagonales) y **empate** (tablero lleno).
5. Código **sin estructuras avanzadas**: usar `while`, `for`, `if`, listas, tuplas, NumPy básico.
6. Sin usar `break` ni `continue`; los bucles se controlan por condición.

Estructura mínima

```
tres_en_raya/  
  ttt_core.py      # lógica del juego (funciones puras y  
utilidades)  
  app.py           # interfaz de consola y bucles de juego
```

APP.PY

CÓDIGO:

```
from ttt_core import *

def main():
    jugando=True
    while jugando==True:
        print("--- MODOS DE JUEGO ---")
        print("1. JUGADOR VS JUGADOR")
        print("2. JUGADOR VS CPU")
        opcion=input("¿A que quieres jugar?: ")
        if opcion=="1":
            jugador_vs_jugador()
        elif opcion=="2":
            jugador_vs_cpu()
        else:
            print("Opción inválida.")
        otra=input("¿Quieres echarte otra partiduki? (s/n): ").lower()
        if otra != "s":
            jugando=False
        print("¡Nos vemos!")

main()
```

TTT_CORE.PY (FUNCIONES SALVO MENU)

CÓDIGO:

```
import numpy as np
import random

#FUNCIONES DEL TABLERO TTT
def crear_tablero():
    return np.zeros((3,3), dtype=int)

def movimiento_valido(tablero, fila, columna):
    #SI LA CASILLA ES VALIDA Y ESTÁ VACÍA, DEVUELVE TRUE
    if fila < 0 or fila > 2:
        return False
    if columna < 0 or columna > 2:
        return False
    return tablero[fila, columna]==0

def aplicar_movimiento(tablero, fila, columna, jugador):
    #COLOCA FICHA Y MODIFICA TABLERO BASE
    tablero[fila, columna]=jugador
    return tablero

def mostrar_tablero(tablero):
    print("\n   0   1   2")
    fichas={0:" ",1:"X",2:"O"}
    fila=0
    while fila<3:
        print(str(fila)+" "+fichas[tablero[fila,0]]+" | "+fichas[tablero[fila,1]]+" | "+fichas[tablero[fila,2]])
        if fila < 2:
            print(" ---+---+---")
            fila=fila +1
    print()

#VICTORIA O EMPATE
def hay_ganador(tablero, jugador):
    #FILAS
    fila=0
    while fila < 3:
        if (tablero[fila, 0]==jugador and
            tablero[fila, 1]==jugador and
            tablero[fila, 2]==jugador):
            return True
        fila=fila +1
```

```
#COLUMNAS
col=0
while col <3:
    if (tablero[0, col]==jugador and
        tablero[1, col]==jugador and
        tablero[2, col]==jugador):
        return True
    col=col +1
#DIAGONALES(PRINCIPAL Y SECUNDARIA)
if (tablero[0, 0]==jugador and
    tablero[1, 1]==jugador and
    tablero[2, 2]==jugador):
    return True
if (tablero[0, 2]==jugador and
    tablero[1, 1]==jugador and
    tablero[2, 0]==jugador):
    return True
#EMPATE
def tablero_lleno(tablero):
    #DEVUELVE TRUE CUANDO NO QUEDEN CASILLAS VACIAS
    return np.all(tablero != 0)

#CAMBIAR DE JUGADOR
def cambiar_turno(jugador_actual):
    if jugador_actual ==1:
        return 2
    return 1

#PEDIR MOVIMIENTO AL USUARIO
def pedir_jugada(tablero, jugador):
    valida=False
    fila=-1
    col=-1
    while valida==False:
        print("TURNO DEL JUGADOR", jugador)
        fila_correcta=False
        while fila_correcta==False:
            entrada=input("Fila (0-2): ")
            try:
                f=int(entrada)
                if f>=0 and f<=2:
                    fila=f
                    fila_correcta=True
            except:
                print("La fila debe ser 0,1 o 2")
                print("Introduce un numero válido, colega.")
        col_correcta=False
```

```
    while col_correcta==False:
        entrada=input("Columna (0-2): ")
        try:
            c=int(entrada)
            if c>=0 and c<=2:
                col=c
                col_correcta=True
            else:
                print("La fila debe ser 0,1 o 2")
        except:
            print("Introduce un numero válido, colega.")
    if movimiento_valido(tablero, fila, col):
        valida=True
    else:
        print("Esa casilla está ya marcada.")
    return fila, col

##MODOS DE JUEGO
#JUGADOR VS JUGADOR
def jugador_vs_jugador():
    tablero=crear_tablero()
    jugador_actual=1
    terminado=False
    print("--- MODO JUGADOR VS JUGADOR ---")
    while terminado==False:
        mostrar_tablero(tablero)
        fila,col=pedir_jugada(tablero, jugador_actual)
        aplicar_movimiento(tablero, fila, col, jugador_actual)
        if hay_ganador(tablero, jugador_actual):
            mostrar_tablero(tablero)
            print("¡GANA EL JUGADOR", jugador_actual, "!")
            terminado=True
        else:
            if tablero_lleno(tablero):
                mostrar_tablero(tablero)
                print("EMPATE")
                terminado=True
            else:
                jugador_actual=cambiar_turno(jugador_actual)
#JUGADOR VS CPU
def jugador_vs_cpu():
    tablero=crear_tablero()
    jugador_actual=1
    terminado=False
    print("--- MODO JUGADOR VS CPU ---")
    while terminado==False:
        mostrar_tablero(tablero)
        if jugador_actual==2:
            print("TURNO DE LA CPU")
```

```
        fila,col=elegir_movimiento_cpu(tablero)
        print("CPU ELIGE LA CASILLA:", fila, col)
    else:
        fila,col=pedir_jugada(tablero, jugador_actual)
        aplicar_movimiento(tablero, fila, col, jugador_actual)
        if hay_ganador(tablero, jugador_actual):
            mostrar_tablero(tablero)
            if jugador_actual==2:
                print("¡HA GANADO LA CPU!")
            else:
                print("¡GANA EL JUGADOR!")
            terminado=True
        else:
            if tablero_lleno(tablero):
                mostrar_tablero(tablero)
                print("EMPATE")
                terminado=True
            else:
                jugador_actual=cambiar_turno(jugador_actual)

#MOVIMIENTOS CPU
def elegir_movimiento_cpu(tablero):
    import random
    fila=-1
    col=-1
    movimiento_encontrado=False
    #BUCLE QUE BUSCA CASILLA VACIA
    while movimiento_encontrado==False:
        fila_int= random.randint(0, 2)
        col_int= random.randint(0, 2)
        if tablero[fila_int, col_int]==0:
            fila=fila_int
            col=col_int
            movimiento_encontrado=True
    return fila,col
```


COMPROBACIÓN:

VS JUGADOR

```
PS C:\Users\kiikesiito> python -u "c:\Users\kiikesiito\Desktop\Tres en raya\tempCodeRunnerFile.py"
--- MODOS DE JUEGO ---
1. JUGADOR VS JUGADOR
2. JUGADOR VS CPU
¿A que quieres jugar?: 1
--- MODO JUGADOR VS JUGADOR ---

  0  1  2
0  |  |  |
---+---+---
1  |  |  |
---+---+---
2  |  |  |

TURNO DEL JUGADOR 1
Fila (0-2): 1
Columna (0-2): 1

  0  1  2
0  |  |  |
---+---+---
1  | X |  |
---+---+---
2  |  |  |

TURNO DEL JUGADOR 2
Fila (0-2): 2
Columna (0-2): 1

  0  1  2
0  |  |  |
---+---+---
1  | X |  |
---+---+---
2  | 0 |  |
```

TURNO DEL JUGADOR 1

Fila (0-2): 2

Columna (0-2): 2

	0	1	2
0			
---+---+---			
1		X	
---+---+---			
2		O	X

TURNO DEL JUGADOR 2

Fila (0-2): 0

Columna (0-2): 1

	0	1	2
0		O	
---+---+---			
1		X	
---+---+---			
2		O	X

TURNO DEL JUGADOR 1

Fila (0-2): 0

Columna (0-2): 0

	0	1	2
0	X		O
---+---+---			
1		X	
---+---+---			
2		O	X

¡GANA EL JUGADOR 1 !

¿Quieres echarte otra partiduki? (s/n):

VS CPU:

```
¿Quieres echarte otra partiduki? (s/n): S
--- MODOS DE JUEGO ---
1. JUGADOR VS JUGADOR
2. JUGADOR VS CPU
¿A que quieres jugar?: 2
--- MODO JUGADOR VS CPU ---

  0  1  2
0  |  |
---+---+---
1  |  |
---+---+---
2  |  |

TURNO DEL JUGADOR 1
Fila (0-2): 1
Columna (0-2): 1

  0  1  2
0  |  |
---+---+---
1  | X |
---+---+---
2  |  |

TURNO DE LA CPU
CPU ELIGE LA CASILLA: 0 1

  0  1  2
0  | O |
---+---+---
1  | X |
---+---+---
2  |  |
```

```
TURNO DE LA CPU
CPU ELIGE LA CASILLA: 2 2

  0  1  2
0  X | O |
  ---+---+---
1  | X |
  ---+---+---
2  | | O

TURNO DEL JUGADOR 1
Fila (0-2): 2
Columna (0-2): 0

  0  1  2
0  X | O |
  ---+---+---
1  | X |
  ---+---+---
2  X | | O

TURNO DE LA CPU
CPU ELIGE LA CASILLA: 1 2

  0  1  2
0  X | O |
  ---+---+---
1  | X | O
  ---+---+---
2  X | | O

TURNO DEL JUGADOR 1
Fila (0-2): 0
Columna (0-2): 2

  0  1  2
0  X | O | X
  ---+---+---
1  | X | O
  ---+---+---
2  X | | O

¡GANA EL JUGADOR!
¿Quieres echarle otra partiduki? (s/n): █
```