

Materia: Computación Tolerante a Fallas.

NRC: 179961

Maestro: Lopez Franco, Michael Emanuel

Aula: X-02

Sección: D06

Alumno: Zashuvath López Moreno, Ethan Israel

Código: 216493953



Índice

Introducción	3
Objetivo:	3
Contenido.....	3
Bibliotecas	3
Funciones	3
Configuración de la interfaz gráfica:	5
Inicialización de variables:	6
Bucle principal:.....	6
Ejemplo	6

Introducción

Generar un programa que sea capaz de restaurar el estado de ejecución

Objetivo:

Conocer los conceptos básicos en sistemas tolerantes a fallas.

Contenido

Bibliotecas

Las bibliotecas utilizadas fueron “tkinter” para la interfaz gráfica. “messagebox” para mostrar mensajes emergentes y “pacle” para la serialización de datos.

```
1 import tkinter as tk
2 from tkinter import messagebox
3 import pickle
```

Funciones

- Función imprimir_tablero():
Esta función se utiliza para crear y mostrar el tablero de juego en la interfaz gráfica. Utiliza un bucle anidado para generar un botón para cada celda del tablero.

```
5 def imprimir_tablero():
6     for i in range(3):
7         for j in range(3):
8             btn = tk.Button(frame, text=tablero[i][j], width=10, height=3,
9                             command=lambda row=i, col=j: hacer_movimiento(row, col))
10            btn.grid(row=i, column=j)
```

- Función hacer_movimiento(row, col): Esta función se llama cuando un jugador hace clic en una celda del tablero. Verifica si la celda está vacía y, en ese caso, coloca la ficha del jugador actual en la celda, actualiza el tablero y verifica si el jugador ha ganado o si el juego ha terminado en empate. Luego, cambia el turno del jugador y guarda el estado del tablero en un archivo.

```
12 def hacer_movimiento(row, col):
13     global turno
14     if tablero[row][col] == " ":
15         tablero[row][col] = turno
16         imprimir_tablero()
17         if verificar_ganador(turno):
18             messagebox.showinfo("Ganador", f"¡Jugador {turno} ha ganado!")
19             reiniciar_juego()
20         elif all(tablero[i][j] != " " for i in range(3) for j in range(3)):
21             messagebox.showinfo("Empate", "¡Es un empate!")
22             reiniciar_juego()
23         turno = "O" if turno == "X" else "X"
24         guardar_tablero()
```

- Función verificar_ganador(jugador): Esta función verifica si el jugador actual ha ganado el juego al comprobar si hay una línea vertical, horizontal o diagonal de sus fichas en el tablero.

```
26 def verificar_ganador(jugador):
27     for i in range(3):
28         if all(tablero[i][j] == jugador for j in range(3)):
29             return True
30         if all(tablero[j][i] == jugador for j in range(3)):
31             return True
32     if all(tablero[i][i] == jugador for i in range(3)) or all(tablero[i][2 - i] == jugador for i in range(3)):
33         return True
34     return False
```

- Funciones guardar_tablero() y cargar_tablero():
guardar_tablero() guarda el estado actual del tablero en un archivo llamado "checkpoint.txt" utilizando el módulo pickle.

cargar_tablero() intenta cargar el tablero desde el archivo mencionado. Si el archivo no existe, crea un tablero vacío.

```
36 def guardar_tablero():
37     with open("checkpoint.txt", "wb") as f:
38         pickle.dump(tablero, f)
39
40 def cargar_tablero():
41     try:
42         with open("checkpoint.txt", "rb") as f:
43             return pickle.load(f)
44     except FileNotFoundError:
45         return [[" " for _ in range(3)] for _ in range(3)]
```

- Función reiniciar_juego():

Esta función restablece el tablero a su estado inicial, reinicia el turno del jugador y guarda el tablero vacío en el archivo.

```
47 def reiniciar_juego():
48     global tablero, turno
49     tablero = [[" " for _ in range(3)] for _ in range(3)]
50     turno = "X"
51     guardar_tablero()
52     imprimir_tablero()
```

- Funciones `juego_nuevo()` y `continuar_juego()`:

`juego_nuevo()` inicia un juego nuevo restableciendo el tablero y el turno.

`continuar_juego()` carga un juego anterior si existe un archivo "checkpoint.txt" y continúa desde donde se quedó.

```
54 ~ def juego_nuevo():
55     global tablero, turno
56     tablero = [[" " for _ in range(3)] for _ in range(3)]
57     turno = "X"
58     guardar_tablero()
59     imprimir_tablero()
60
61 ~ def continuar_juego():
62     global tablero, turno
63     tablero = cargar_tablero()
64     imprimir_tablero()
```

Configuración de la interfaz gráfica:

El programa crea una ventana de juego con un título y un marco donde se mostrará el tablero.

También agrega un menú que permite a los jugadores comenzar un nuevo juego, continuar un juego anterior o salir del programa.

```
69 root = tk.Tk()
70 root.title("Juego de Gato")
71
72 frame = tk.Frame(root)
73 frame.pack()
74
75 menu = tk.Menu(root)
76 root.config(menu=menu)
77
78 file_menu = tk.Menu(menu)
79 menu.add_cascade(label="Archivo", menu=file_menu)
80 file_menu.add_command(label="Nuevo Juego", command=juego_nuevo)
81 file_menu.add_command(label="Continuar Juego", command=continuar_juego)
82 file_menu.add_separator()
83 file_menu.add_command(label="Salir", command=root.quit)
```

Inicialización de variables:

El programa inicializa el tablero y el turno del jugador. El tablero se carga desde el archivo "checkpoint.txt" si existe.

```
66 tablero = cargar_tablero()
67 turno = "X"
```

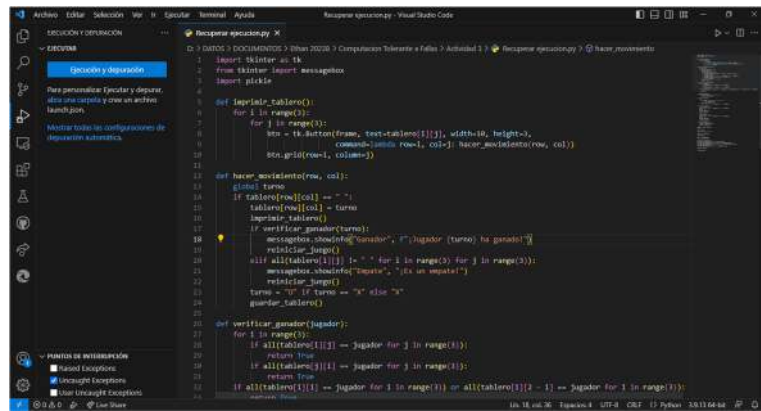
Bucle principal:

```
87 root.mainloop()
```

Ejemplo

Apertura	
Juego	
Archivos	

Ejecución cerrada



```
1 import tkinter as tk
2 from tkinter import messagebox
3 import pickle
4
5 def imprimir_tablero():
6     for i in range(3):
7         for j in range(3):
8             btn = tk.Button(frame, text=tablero[i][j], width=10, height=1,
9                             command=lambda row=i, col=j: hacer_movimiento(row, col))
10            btn.grid(row=i, column=j)
11
12 def hacer_movimiento(row, col):
13     global turno
14     if tablero[row][col] == " ":
15         tablero[row][col] = turno
16         imprimir_tablero()
17         if verificar_ganador(turno):
18             messagebox.showinfo("Mensaje", f"¡Jugador {turno} ha ganado!")
19             reiniciar_juego()
20         elif all(tablero[i][j] != " " for i in range(3) for j in range(3)):
21             messagebox.showinfo("Mensaje", "¡Es un empate!")
22             reiniciar_juego()
23         turno = "O" if turno == "X" else "X"
24         guardar_tablero()
25
26 def verificar_ganador(jugador):
27     for i in range(3):
28         if all(tablero[i][j] == jugador for j in range(3)):
29             return True
30         if all(tablero[j][i] == jugador for j in range(3)):
31             return True
32     if all(tablero[i][i] == jugador for i in range(3)) or all(tablero[i][2-i] == jugador for i in range(3)):
33         return True
34     return False
```

Ultimo respaldo



Juego de G...		
Archivo		
X		O
O	O	X
X	X	O