

**Universidad de Guadalajara
Sistema de Educación Media Superior
Centro Universitario de Ciencias Exactas e Ingenierías**



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

D06 2023 B

Alumno: Esquivel Barbosa Diego Humberto

Código: 211401635

Carrera: INCO

Fecha: 03/09/2023

Github:<https://github.com/DiegoSargent/Computacion-Tolerante-a-Fallas/blob/1c95a9c4d7aba4eba62eb86ba1aac4f8650b3189/Checkpoint.py>

Introducción

Se nos solicita crear un programa con la opción de tener un punto de restauración o también conocido como un checkpoint para cuando la aplicación se llegue a cerrar o se presente algún error se pueda reanudar desde donde se quedó.

Para este ejemplo decidí usar el popular juego de Gato para dos jugadores en Python con la capacidad de guardar y cargar partidas en cualquier momento como si se tratara de un juego en consola donde restauras el punto de control.

Desarrollo

Función `imprimir_tablero(tablero)`

Esta función se encarga de imprimir el estado actual del tablero del juego. Utiliza una matriz para representar el tablero y lo muestra en la consola en un formato legible.

Función `verificar_ganador(tablero, jugador)`

Esta función verifica si un jugador ha ganado el juego al comprobar las filas, columnas y diagonales del tablero.

Funciones `guardar_partida()` y `cargar_partida()`

Estas funciones se encargan de guardar y cargar el estado actual del juego en un archivo binario utilizando el módulo `pickle`. Esto permite a los jugadores guardar su progreso y continuar el juego más tarde.

Función `jugar_gato()`

La función principal del juego administra el flujo del juego. Permite a los jugadores tomar turnos, realiza comprobaciones de victoria y empate, y ofrece la opción de guardar la partida en cualquier momento.

Código

```
import pickle

# Función para imprimir el tablero
def imprimir_tablero(tablero):
    for fila in tablero:
        print(" | ".join(fila))
        print("-" * 9)

# Función para verificar si alguien ha ganado
def verificar_ganador(tablero, jugador):
    for fila in tablero:
        if all(cell == jugador for cell in fila):
            return True

    for col in range(3):
        if all(tablero[row][col] == jugador for row in range(3)):
```

```

        return True

    if all(tablero[i][i] == jugador for i in range(3)) or all(tablero[i][2 - i] == jugador for i in range(3)):
        return True

    return False

# Función para guardar la partida
def guardar_partida(tablero, jugador_actual):
    partida = {'tablero': tablero, 'jugador_actual': jugador_actual}
    with open('partida_guardada.pickle', 'wb') as archivo:
        pickle.dump(partida, archivo)

# Función para cargar la partida
def cargar_partida():
    try:
        with open('partida_guardada.pickle', 'rb') as archivo:
            partida = pickle.load(archivo)
            return partida['tablero'], partida['jugador_actual']
    except FileNotFoundError:
        return [[' ' for _ in range(3)] for _ in range(3)], 'X' # Inicializa el tablero si no se encuentra una
partida guardada

# Función principal
def jugar_gato():
    tablero, jugador_actual = cargar_partida()

    while True:
        imprimir_tablero(tablero)
        print(f"Turno del jugador {jugador_actual}")

        fila = int(input("Ingresa el número de fila (0, 1, 2): "))
        columna = int(input("Ingresa el número de columna (0, 1, 2): "))

        if tablero[fila][columna] == ' ':
            tablero[fila][columna] = jugador_actual
        else:
            print("Esa casilla ya está ocupada. Inténtalo de nuevo.")
            continue

        if verificar_ganador(tablero, jugador_actual):
            imprimir_tablero(tablero)
            print(f"¡El jugador {jugador_actual} ha ganado!")
            break

        if ' ' not in [cell for row in tablero for cell in row]:
            imprimir_tablero(tablero)
            print("¡Empate!")
            break

    jugador_actual = 'O' if jugador_actual == 'X' else 'X'

```

```
guardar_partida(tablero, jugador_actual)
```

```
if __name__ == "__main__":  
    jugar_gato()
```

```
  |  |  
-----  
  |  |  
-----  
  |  |  
-----  
Turno del jugador X  
Ingresa el número de fila (0, 1, 2): 1  
Ingresa el número de columna (0, 1, 2): 1  
  |  |  
-----  
  | X |  
-----  
  |  |  
-----  
Turno del jugador O  
Ingresa el número de fila (0, 1, 2): 1  
Ingresa el número de columna (0, 1, 2): 1  
Esa casilla ya está ocupada. Inténtalo de nuevo.  
  |  |  
-----  
  | X |  
-----
```




```
Turno del jugador O  
Ingresa el número de fila (0, 1, 2): 1  
Ingresa el número de columna (0, 1, 2): 2  
  |  |  
-----  
  | X | O  
-----  
  |  |  
-----  
Turno del jugador X  
Ingresa el número de fila (0, 1, 2): 1  
Ingresa el número de columna (0, 1, 2): 0  
  |  |  
-----  
X | X | O  
-----  
  |  |  
-----  
Turno del jugador O  
Ingresa el número de fila (0, 1, 2): guardar
```

```

Traceback (most recent call last):
  File "c:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint\Checkpoint.py", line 71, in <module>
    jugar_gato()
  File "c:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint\Checkpoint.py", line 47, in jugar_gato
    fila = int(input("Ingresa el número de fila (0, 1, 2): "))
ValueError: invalid literal for int() with base 10: 'guardar'
PS C:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint> c;; cd 'c:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint'; & 'C:\Users\Diego\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\Diego\.vscode\extensions\ms-python.python-2023.8.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '59878' '--' 'c:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint\Checkpoint.py'
|
-----
X | X | O
|
-----
|

```

Si cerramos el programa o pasa un error, podemos volver a iniciarlo y retomara la actividad.

	<u>__pycache__</u>	03/09/2023 16:09	Carpeta de archivos	
	Checkpoint	03/09/2023 16:14	Archivo de origen ...	3 KB
	<u>partida_guardada.pickle</u>	03/09/2023 16:23	Archivo PICKLE	1 KB

Conclusión

Con este código se dio una demostración de como funciona un punto de restauración en diferentes programas ya sea para evitar algún tipo de perdida de información o para prevenir errores mayores durante una ejecución.

Se mostro con un ejemplo de un juego ya que pienso que es la forma mas sencilla de hacer la demostración por que estamos acostumbrados a verlos de forma cotidiana como cuando estas realizando una actividad escolar y por algún motivo debes suspender la actividad y retomarla más tarde.

Es lo mismo que se intenta demostrar en la actividad, pero enfocado en un programa de Python.

Bibliografía

- Python. (2023). Documentación oficial de Python. <https://docs.python.org/>
- Python Software Foundation. (2023). Módulo pickle. <https://docs.python.org/3/library/pickle.html>