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 solicito crear un programa con la opción de tener un punto de restauración o tambien 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)):
```

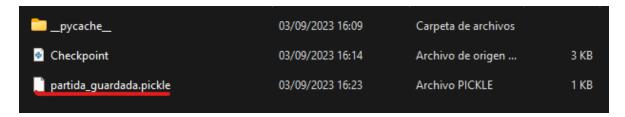
```
return True
  if all(tablero[i][i] == jugador for i in range(3)) or all(tablero[i][2 - i] == jugador for i in range(3)):
     return True
  return False
# Función para guardar la partida
def guardar partida(tablero, jugador actual):
  partida = {'tablero': tablero, 'jugador_actual': jugador_actual}
  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
       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()
```

```
| Traceback (most recent call last):
| File "c:\Users\Diego\Desktop\Computacion Tolerante a Fallas\Checkpoint\Checkpoint.py", line 71, in \( \text{module} \)
| \( \text{jugar_gato} \)
| \( \text{pigo} \text{past}(0) \)
| \( \text{pigo} \text{pigo
```

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



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