



COMPUTACION TOLERANTE A FALLAS



**Actividad 03 Generar un programa que restaure el
estado de ejecución**

Universidad de Guadalajara

Nombre: Marco Aurelio Domínguez Amezcua

Código: 216818534

Carrera: Ingeniería en computación.

Fecha: 04/09/23

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

Para esta primera parte importamos las librerías pickle y curses. La primera de ellas nos va a servir para leer y escribir datos en un archivo, la información es serializada y poco legible para las personas, y por el lado de 'curses', la cual crea una interfaz de consola más interactiva. Además, se tiene una variable que se refiere al archivo que usaremos para guardar la información, y funciones las cuales nos servirán para guardar, leer y guardar automáticamente los cambios que se vayan realizando.

```
pick.py > ...
1  import pickle
2  import curses
3
4  # Nombre del archivo donde se guarda la información
5  archivo_datos = 'datos.pkl'
6
7  # Función para cargar la información existente o crear una nueva lista vacía
8  def cargar_o_crear_lista(nombre_archivo):
9      try:
10         with open(nombre_archivo, 'rb') as archivo:
11             datos = pickle.load(archivo)
12     except (FileNotFoundError, EOFError):
13         datos = []
14     return datos
15
16 # Función para guardar la lista en el archivo
17 def guardar_lista(nombre_archivo, lista):
18     with open(nombre_archivo, 'wb') as archivo:
19         pickle.dump(lista, archivo)
20
21 # Función para guardar automáticamente la lista cada vez que se modifica
22 def guardar_automáticamente(lista):
23     guardar_lista(archivo_datos, lista)
24     stdscr.addstr(0, 0, "Información guardada automáticamente.")
25     stdscr.refresh()
26
```

En este bloque debemos configurar la pantalla de la interfaz, y posteriormente cargar una lista existente o crear una nueva lista, además de mostrar la información que este guardada actualmente en nuestro archivo.

```

27 # Configuración de la pantalla de curses
28 stdscr = curses.initscr()
29 curses.cbreak()
30 stdscr.keypad(True)
31 curses.noecho()
32
33 # Cargar la lista de datos existente o crear una nueva
34 lista_datos = cargar_o_crear_lista(archivo_datos)
35
36 # Mostrar la información actual
37 stdscr.addstr(1, 0, "Información actual: " + ''.join(lista_datos))
38 stdscr.refresh()
39

```

Para la ejecución del programa utilizaremos un bucle el cual siga repitiéndose cada vez que la tecla ingresada sea diferente de 'q', ya que al presionar esta se terminara el programa. Cabe mencionar, que el programa es algo simple y solamente muestra la idea del autoguardado en un programa.

```

40 while True:
41     # Obtener el caracter ingresado
42     caracter = stdscr.getch()
43
44     # Salir del bucle si el usuario presiona 'q'
45     if caracter == ord('q'):
46         break
47
48     # Convertir el valor numérico del caracter en un carácter ASCII
49     caracter = chr(caracter)
50
51     # Agregar el caracter a la lista de datos
52     lista_datos.append(caracter)
53
54     # Llamar a la función para guardar automáticamente
55     guardar_automaticamente(lista_datos)
56
57     # Mostrar la información actualizada
58     stdscr.addstr(1, 0, "Información actual: " + ''.join(lista_datos))
59     stdscr.refresh()
60
61 # Finalizar curses y cerrar el programa
62 curses.endwin()
63 print("Saliendo del programa.")
64

```

Resultados

En esta línea se ingresa texto, el cual se guarda automáticamente en el documento 'pkl'.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    POLYGLOT NOTEBOOK

Información guardada automáticamente.
Información actual: Marco Aurel
```

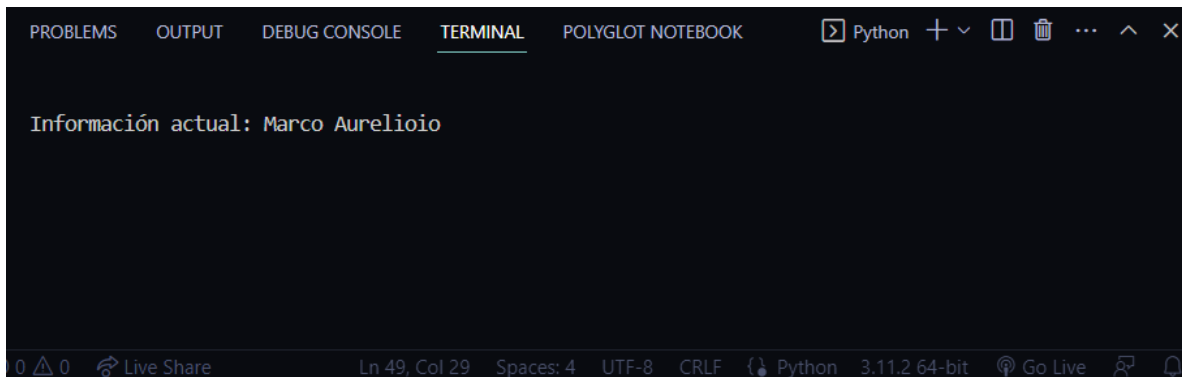
Así es como se visualiza el archivo 'pkl' después de haber recibido información

```
datos.pkl x
```

```
datos.pkl  
1 [EOT4NULNULNULNULNULNUL] (SOHMSOHaSOHrSOHC SOHO SOH SOHA SOHu hETX)
```

Si nosotros cerramos el programa, ya tendremos lista la información que ingresamos para poder continuar con el para la vez que se vuelva a ejecutar.

Al volverlo a ejecutar, tenemos la misma información que teníamos antes de cerrar nuestro programa.



The image shows a terminal window from a Jupyter Notebook. The window has a dark background and a light-colored border. At the top, there is a menu bar with the following items: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and POLYGLOT NOTEBOOK. To the right of the menu bar, there is a tab labeled 'Python' with a plus sign, a dropdown arrow, a window icon, a trash icon, a three-dot menu, an up arrow, and a close icon. The main area of the terminal displays the text 'Información actual: Marco Aurelioio'. At the bottom of the terminal, there is a status bar with the following information: 0 errors, 0 warnings, a Live Share icon, the current line and column (Ln 49, Col 29), the number of spaces (Spaces: 4), the encoding (UTF-8), the line ending (CRLF), the Python version (Python 3.11.2 64-bit), a Go Live icon, and a bell icon.

Finalmente, así es cómo funciona el auto guardado de información en Python.

Conclusión

La verdad no es un gran programa, debido a falta de tiempo no pude crear algo más complejo, pero me quedo con el aprendizaje de esta práctica y el conocimiento de saber que se puede auto guardar información en cualquier programa para que el usuario pueda estar tranquilo de que, si por alguna razón deja de funcionar el programa, este siga en el punto que se quedó, espero posteriormente aprender más a profundidad del tema y así poder aplicarlo a mis futuros proyectos.