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

Nombre: Diana Fernanda Castillo Rebolledo

Materia: Computación Tolerante a Fallas

Horario: L-I 11:00 am-1:00 pm

Profesor: Dr. Michel Emanuel López Franco

Sección: D06

Para resolver y completar este programa utilicé el método de serialización con Pickle, el cual convierte los objetos en flujos de bytes para guardarlos en un archivo pickle, que es un archivo binario, para poder volver a cargar los objetos cuando se necesite, transformándolos nuevamente en objetos de Python deserializados. Lo único que se tiene que hacer para empezar a trabajar con el módulo pickle es importarlo (import pickle).

El programa consta de una agenda telefónica que tiene las opciones de agregar contacto, ver contactos, buscar contacto y borrar todo. Para agregar un contacto, se crea un diccionario vacío que es donde se guardarán los datos del contacto, después se pregunta por su nombre, teléfono y correo, validando que el teléfono solo contenga números, y el correo contenga un símbolo de '@'. Una vez que los datos son correctos, estos se agregan al diccionario, y a su vez el diccionario del contacto se agrega a una lista definida al principio del programa llamada contactos.

```
AGENDA TELEFONICA

Escoge una opcion:
1) Agregar contacto
2) Mostrar contactos
3) Buscar contacto
4) Borrar todo
5) Salir
> 1
Introduce el nombre del contacto: Diana
Introduce su telefono: 3334985034
Introduce su correo: diana22@gmail.com
```

En seguida, se abre (o se crea si no existe) un archivo.pickle en escritura formato binario, y se serializa el objeto (diccionario) y se guarda en el archivo pickle, después cerramos el archivo y el contacto ya estará guardado. Con esto se garantiza que haya un respaldo de cada contacto agregado a la agenda por si ocurre algún error o se cierra el programa.

```
contacto['Nombre'] = nombre
contacto['Telefono'] = telefono
contacto['Correo'] = correo
contactos.append(contacto)

archivo = open('datos.pickle', 'ab')
pickle.dump(contacto, archivo)
archivo.close()
```

La segunda opción es para mostrar todos los contactos que haya en la lista, para ello, si hay al menos un elemento en la lista, se recorre toda la lista con un for y se imprime cada contacto con la ayuda de join para crear una cadena con los elementos del diccionario y darle un mejor formato a cada contacto.

```
> 2
Nombre: Diana
Telefono: 3334985034
Correo: diana22@gmail.com

Nombre: Luis
Telefono: 3311388419
Correo: luis1999@gmail.com
```

La tercera opción que es para buscar un contacto busca el nombre del contacto entre cada contacto de la lista de contactos y si lo encuentra, lo imprimirá.

```
> 3
Introduce el nombre del contacto: Diana
Nombre: Diana
Telefono: 3334985034
Correo: diana22@gmail.com
```

La cuarta opción sirve para eliminar todos los contactos si es que la lista de contactos no está vacía, también se elimina el archivo.pickle.

```
> 4
Contactos eliminados con exito.

Escoge una opcion:
1) Agregar contacto
2) Mostrar contactos
3) Buscar contacto
4) Borrar todo
5) Salir
> 2
No hay contactos.
```

Por el contrario, si no se eliminan los contactos, estos permanecerán guardados en el archivo pickle aunque la aplicación se cierre, y cuando se vuelva a abrir automáticamente los contactos se deserializarán para formar nuevamente diccionarios de Python, para esto antes de entrar al menú de opciones, el programa hace que se abra el archivo si es que existe, lo abre en lectura formato binario, y va deserializando cada objeto y agregándolo a la lista de contactos hasta que el archivo quede vacío, y se cierra el archivo.

```python
if os.path.exists('datos.pickle'):
    archivo = open('datos.pickle', 'rb')
    while True:
        try:
            contactos.append(pickle.load(archivo))
        except EOFError:
            break

    archivo.close()
```

```
PS C:\Users\user\Desktop\Serialización> python -u "c:\Use
AGENDA TELEFONICA

Escoge una opcion:
1) Agregar contacto
2) Mostrar contactos
3) Buscar contacto
4) Borrar todo
5) Salir
> 2
Nombre: Diana
Telefono: 3334985034
Correo: diana22@gmail.com

Nombre: Luis
Telefono: 3311388419
Correo: luis1999@gmail.com
```

Código en GitHub: https://github.com/diana-castillo/Application-checkpointing.git

Conclusiones:

La serialización me parece una opción bastante útil para realizar respaldos de información, en especial cuando se busca guardar datos de gran volumen que necesitan estar seguros en caso de que ocurra un error en la aplicación en la que están almacenados, para que de esta manera estos datos queden guardados y puedan restaurarse ya sea totalmente o al menos la mayoría una vez que la aplicación se inicie de nuevo para evitar errores o problemas futuros por la falta de esos datos. En Python hay varias formas de serialización, escogí utilizar Pickle ya que es una manera sencilla y eficiente de guardar y posteriormente restaurar objetos de varios tipos que hay en Python, incluso clases y funciones.

Referencias:

pickle — Python object serialization. (s. f.). Python Software Foundation. Recuperado 17 de febrero de 2022, de https://docs.python.org/3/library/pickle.html

Pickle. serializar datos con Python. (2019, 23 octubre). frankgalandev. Recuperado 17 de febrero de 2022, de https://frankgalandev.com/pickle-serializar-datos-con-python/

Los pickles de Python. (s. f.). Programación en Castellano. Recuperado 18 de febrero de 2022, de https://programacion.net/articulo/los_pickles_de_python_1860