



# UNIVERSIDAD DE GUADALAJARA - CUCEI

ALUMNO: ARIEL HUMBERTO VALLE ESCOTO  
PROFESOR: MICHEL EMANUEL LOPEZ FRANCO

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

```
"C:\Users\PC Budgie\anaconda\python.exe" "C:/Users/PC Budgie/Desktop/estado de recuperacion practica/main.py"
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 1
Introduzca el primer dato: 345
Introduzca el segundo dato: 678
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 2
Los datos del estado de ejecución son:
dato1: 345
dato2: 678
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 1
Introduzca el primer dato: 1
Introduzca el segundo dato: 2
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 2
Los datos del estado de ejecución son:
dato1: 1
dato2: 2
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada:
```

Este es el menú donde se hacen checkpoints de los datos almacenados en el programa. Introduje en el primer dato 345 y 678 y los restauré por así decirse los guardé en el archivo .pk1 y luego ingresé otros números el 1 y el 2 e hice el mismo proceso de restauración (checkpoint).

```

import pickle
# Esta es la clase que se almacenará en el archivo pickle
class EstadoEjecucion:
    def __init__(self, dato1, dato2):
        self.dato1 = dato1
        self.dato2 = dato2

# Función para guardar el estado de ejecución
def guardarEstado(estados):
    archivo = open('estado.pk1', 'wb')
    pickle.dump(estados, archivo)
    archivo.close()

# Función para restaurar el estado de ejecución
def restaurarEstado():
    archivo = open('estado.pk1', 'rb')
    estado = pickle.load(archivo)
    archivo.close()
    return estado

# Menú principal
while True:
    print("Menu principal")
    print("1. Crear un nuevo estado de ejecución")
    print("2. Restaurar un estado de ejecución")
    print("3. Salir")
    opcion = int(input("Introduzca la opción deseada: "))
    if opcion == 1:
        dato1 = int(input("Introduzca el primer dato: "))
        dato2 = int(input("Introduzca el segundo dato: "))
        estado = EstadoEjecucion(dato1, dato2)
        guardarEstado(estado)
    elif opcion == 2:
        estado = restaurarEstado()
        print("Los datos del estado de ejecución son:")
        print("dato1:", estado.dato1)
        print("dato2:", estado.dato2)
    elif opcion == 3:
        break

```

Utilicé la clase **PICKLE**, esta almacena los datos en un archivo .pk1 donde hace los checkpoints de los datos ingresados en el programa y a la hora de cerrar o terminar el programa una vez vuelto a iniciar se quedan los últimos datos que se ingresaron

Enseguida adjunto las capturas:

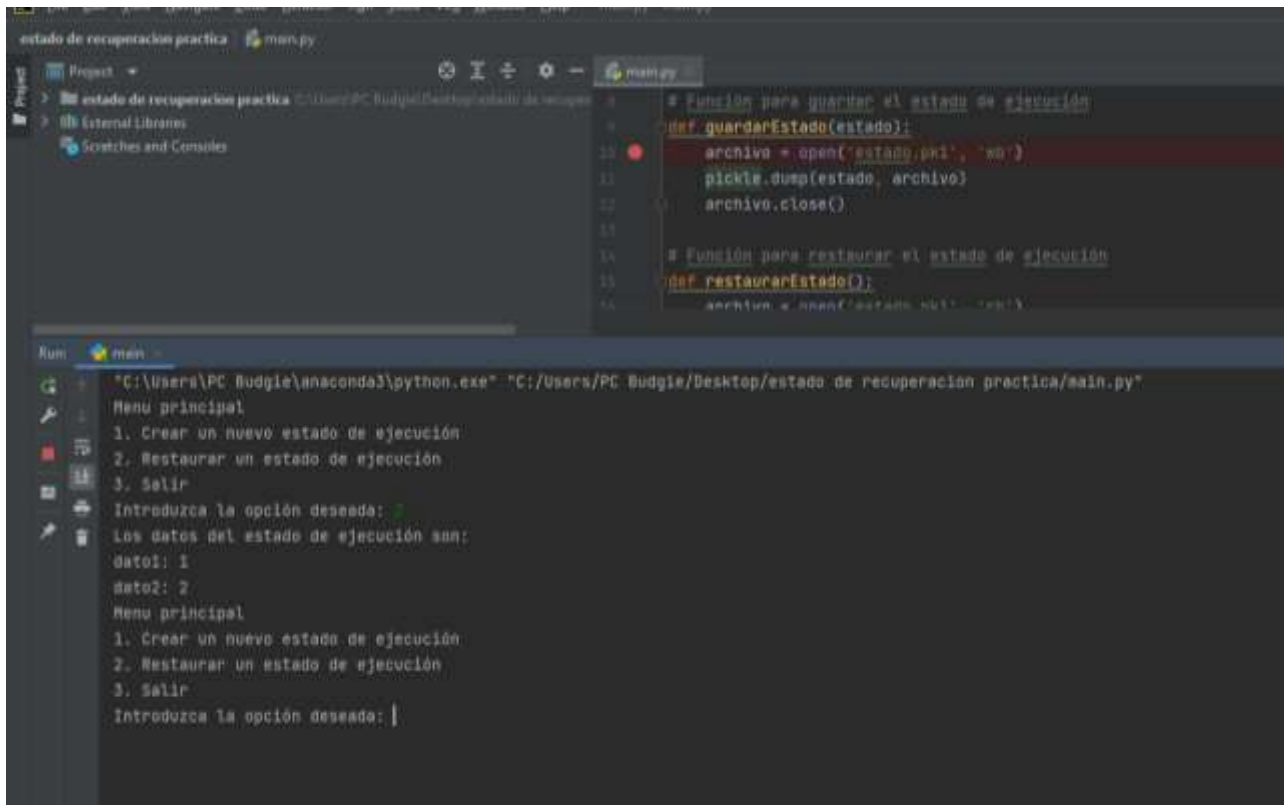
```
# Función para guardar el estado de ejecución
def guardarEstado(estado):
    archivo = open('estado.pkl', 'wb')
    pickle.dump(estado, archivo)
    archivo.close()

# Función para restaurar el estado de ejecución
def restaurarEstado():
    archivo = open('estado.pkl', 'rb')
```

main.py

```
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 1
Los datos del estado de ejecución son:
dato1: 345
dato2: 678
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 2
Introduzca el primer dato: 1
Introduzca el segundo dato: 2
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 1
Los datos del estado de ejecución son:
dato1: 1
dato2: 2
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 2
Process finished with exit code 0
```

Aquí ya finalicé el programa y aquí se puede ver que los últimos valores ingresados fueron los números 1 y 2.



```
estado de recuperacion practica main.py
Project
> estado de recuperacion practica C:\Users\PC Budgie\Desktop\estado de recuperacion practica
> External Libraries
> Scratches and Consoles

# Función para guardar el estado de ejecución
def guardarEstado(estado):
    archivo = open("estado.pkl", "wb")
    pickle.dump(estado, archivo)
    archivo.close()

# Función para restaurar el estado de ejecución
def restaurarEstado():
    archivo = open("estado.pkl", "rb")
```

```
Run main
"C:\Users\PC Budgie\anaconda3\python.exe" "C:\Users\PC Budgie\Desktop\estado de recuperacion practica/main.py"
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: 2
Los datos del estado de ejecución son:
dato1: 1
dato2: 2
Menu principal
1. Crear un nuevo estado de ejecución
2. Restaurar un estado de ejecución
3. Salir
Introduzca la opción deseada: |
```

Y aquí volví a correr el programa y restauré los datos y aparece en pantalla los últimos datos ingresados que eran 1 y 2 como había comentado anteriormente.

## Conclusiones:

Pude notar que este tipo de librerías para guardar datos sirven demasiado para próximos programas mas extensos.

## Bibliografía:

*pickle* — *Python object serialization*. (s. f.). Python documentation.

<https://docs.python.org/3/library/pickle.html>