Universidad de Guadalajara

CUCEI – Centro Universitario de Ciencias Exactas e Ingenierías

Mtro. Michel Emanuel López Franco

Computación tolerante a fallas

Reporte 7

Velasco Hernandez Victor Manuel

Código: 216598879

| D06 | 2022A |

|21/Febrero/2022|

Checkpointing

Universidad de Guadalajara
CUCEI
Computación tolerante a fallas

Victor Manuel Velasco Hernández
|Código: 216598879 | | 2022A |
|21/ Febrero /2022 | |D06|

Reporte 7- Checkpointing

Objetivo:
(Application checkpointing)
Checkpointing is a technique that provides fault tolerance for computing systems.

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

Introducción:
El manejo de puntos de control en los sistemas de software o de computo resultan ser fundamentales en el aspecto de la tolerancia a fallos, lo cuál trae muchos beneficios al usuario y a los programadores evitando graves consecuencias, por lo que el estudio de estrategias para su implementación es fundamental, sobre todo en esta época en la que abundan distintos tipos de tecnologías que pueden mejorar la funcionalidad en un futuro.

Desarrollo:
Este programa es basado en el programa anterior de excepciones por lo que representa una mejora al sistema desarrollado gracias a la implementación de un Checkpoint con la librería Pickle, para este programa se crearon 2 funciones para trabajar el guardado de información, saveProcess() y recoverProcess(), uno para guardar información y la otra para leer:

```
def saveProcess():
    pickle_out=open("dict.pickle","wb")
    pickle.dump(checkpoint,pickle_out)
    pickle_out.close()

def recoverProcess():
    try:
        with open("dict.pickle", "r") as data:
            #checkpoint.readline()
            print("Se ha encontrado un Checkpoint")
            pickle_in = open("dict.pickle", "rb")
            data = pickle.load(pickle_in)
            #print(data)
            pickle_in.close()
            return data
    except FileNotFoundError:
        print("No se ha encontrado un Checkpoint")
        return {}
```

El siguiente fragmento de código muestra un menú, el cuál puede ser suprimido en caso de que haya datos guardados en el Checkpoint, en caso positivo, se regresara a la ultima fase en la que el usuario dejo el programa, cuando se tengan guardados los tres valores, se le pregunta al usuario si para el siguiente proceso desea usarlos, de ser caso contrario, el diccionario utilizado se vacía:

```
finalizado=True
checkpoint=recoverProcess()
opc=0
while True:
    finalizado=True
    if opc==5:
        print("Hasta la proxima!")

        break
    print("Bienvenido a la Calculadora 3000- Tolerante a fallos con Checkpoint \U0001F4BB \U0001F511")
    if 'NUM2' in checkpoint:
        decision = int(input("Se han detectado de un proceso ya finalizado, desea volver a cargarlos? 0==Si o 1==No\n"))
        if decision == 1:
            checkpoint = {}
    if checkpoint == {}:
        print("Escoge el proceso que desees realizar:\n1)Suma\n2)Resta\n3)Multiplicacion\n4)Division\n5)Salir")
    else:
        print("Restaurando punto de control")
```

En esta parte del código se muestra que en caso de que se haya guardado la opción de proceso, simplemente evita preguntarle al usuario y se limita a moverlo al proceso que correspondiente, en caso de que haya introducido uno o dos números, se realizara el proceso:

```
while finalizado:
    try:
        if checkpoint:
            opc = checkpoint['OPC']
            if opc==1:
                print("Volviendo al proceso de suma")
            elif opc==2:
                print("Volviendo al proceso de resta")
            elif opc==3:
                print("Volviendo al proceso de multiplicacion")
            elif opc==4:
                print("Volviendo al proceso de Division")
            elif opc==5:
                finalizado==False
                False
        else:
            opc = int(input("Introduce tu opcion:"))
```

Cada vez que se introduce un número con el teclado es guardado en un diccionario de datos para el Checkpoint.

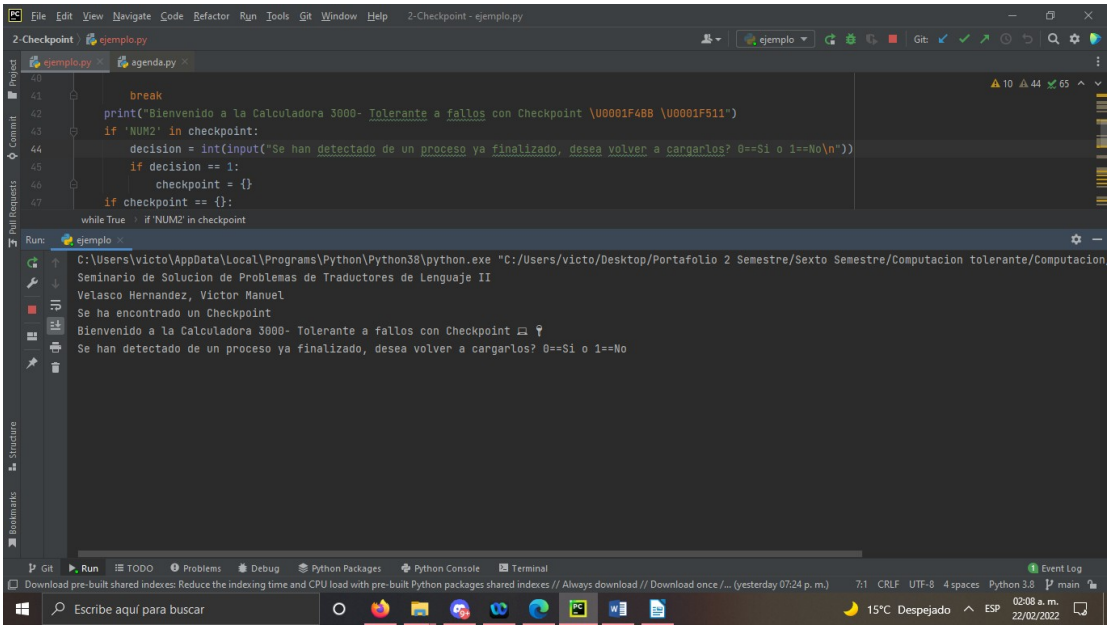
```
else:
    checkpoint['OPC'] = opc
    saveProcess()
    while finalizado:
        try:
            if 'NUM1' in checkpoint:
                    num1=checkpoint['NUM1']
                    if 'NUM2' in checkpoint:
                        num2=checkpoint['NUM2']
                    else:
                        num2 = int(input("Introduce un segundo numero:"))
                        checkpoint['NUM2'] = num2
                        saveProcess()
            else:
                num1 = int(input("Introduce el primer numero:"))
                checkpoint['NUM1'] = num1
                saveProcess()
                num2 = int(input("Introduce un segundo numero:"))
                checkpoint['NUM2'] = num2
                saveProcess()
```

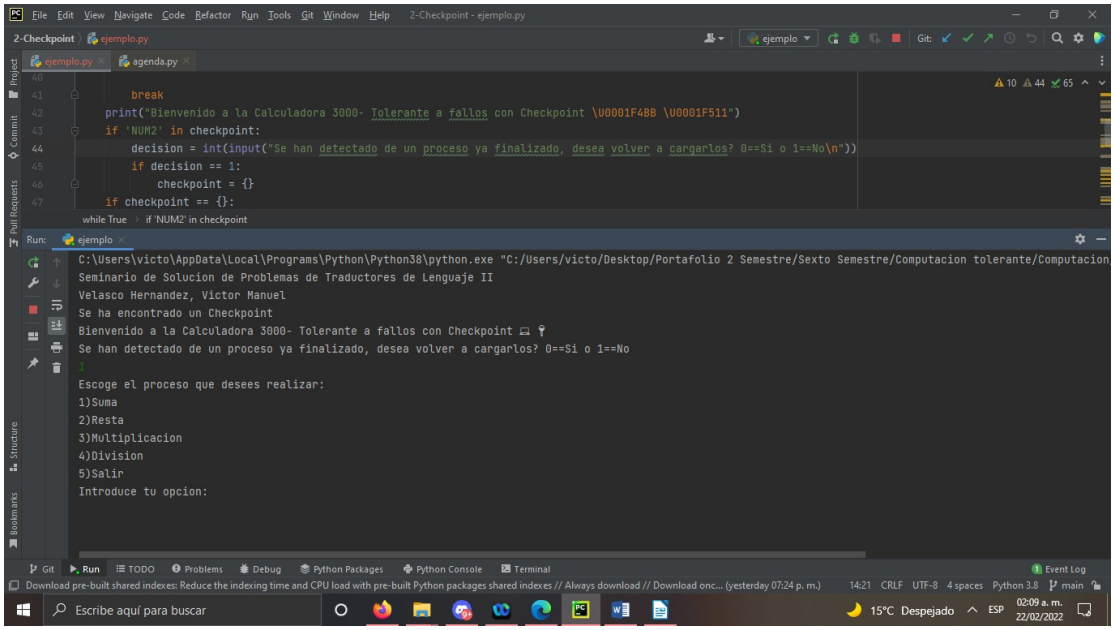
Cuando los 3 datos se han introducido correctamente, se realiza el proceso:

```
if opc==1:
    print("El resultado de la suma de "ˌnum1ˌ" + "ˌnum2ˌ" ="ˌnum1+num2)
    time.sleep(3)
    finalizado=False
    False
elif opc == 2:
    print("El resultado de la resta de ", num1, " - ", num2, " =", num1 -num2)
    time.sleep(3)
    finalizado = False
    False
elif opc == 3:
    print("El resultado de la multiplicacion de ", num1, " * ", num2, " =", num1 * num2)
    time.sleep(3)
    finalizado = False
    False
elif opc == 4:
    print("El resultado de la division de ", num1, "/", num2, " =", num1 / num2)
    time.sleep(3)
    finalizado = False
    False
```

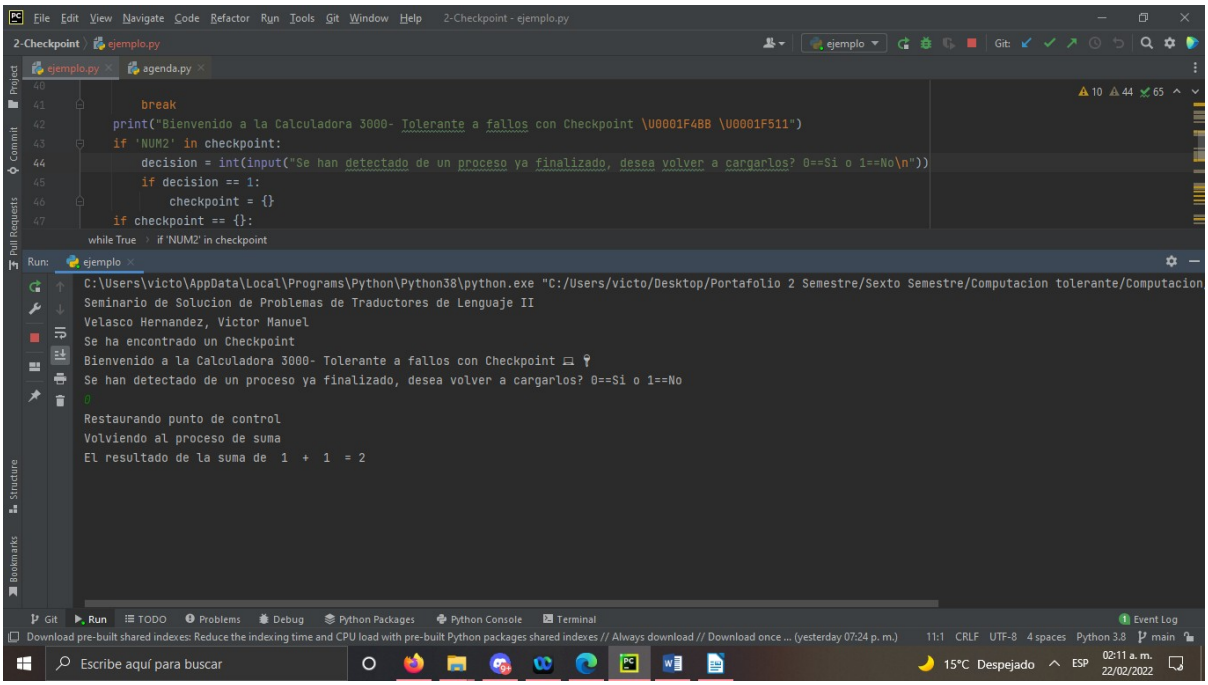
	Universidad de Guadalajara **CUCEI** **Computación tolerante a fallas**	**Victor Manuel Velasco Hernández** **|Código: 216598879 | | 2022A |** **|21/ Febrero /2022 | |D06|**

Capturas de programa:





	Universidad de Guadalajara **CUCEI** **Computación tolerante a fallas**	**Victor Manuel Velasco Hernández** **|Código: 216598879 | | 2022A |** **|21/ Febrero /2022 | |D06|**



Conclusiones:

El manejo de Checkpoints es muy útil al momento de querer preservar información que se encontraba en memoria primaria, lo cuál ayuda a hacer un sistema tolerante a fallos, debido a que prevé muchos aspectos y se base en la misma naturaleza en la que los dispositivos pueden recuperarse de un fallo inesperado, por lo que resulta ser una excelente implementación, por lo que para conocer la naturaleza de estos mecanismos es importante conocer funciones que garantizan el respaldo de la información, además de ayudar a tener una mejor comprensión de las eventualidades que se presentan en las computadoras, esto hace que podamos evitar posibles problemas en el futuro.