Universidad Autónoma de Nuevo León

Facultad de Ciencias Físico Matemáticas

Licenciatura en Ciencias Computacionales

Programación Básica (073)

Diego González Flores – 2086068

Adrián Morales Vázquez - 2149220

Docente: Perla Marlene Viera Gonzalez

E3. Análisis Estadístico y Lectura de Datos

8/5/25

Una de las primeras cosas que decidimos cambiar de la estructura de datos es agregar los datos de conversión uno por uno al archivo de texto “conversion.txt” en vez de sobreescribirlo. Para esto, se utiliza el modo “a” en vez de “w”.

```
def convertirMonedas():
    try:
        fuente = input("Ingrese el código de moneda fuente: ").upper()
        blanco = input("Ingrese el codigo de moneda de destino: ").upper()
        cant = float(input("Ingrese la cantidad a convertir: "))
        url =
f'https://api.currencyfreaks.com/v2.0/convert/latest?apikey={apikey}&from={fuente}&to={blanco}&amount={cant}'
        respuesta = requests.get(url)
        if respuesta.status_code == 200:
            datos = respuesta.json()
            convertido = datos.get('convertedAmount')
            if convertido:
                print(f'{cant} {fuente} es igual a {convertido} {blanco}')
            else:
                print('Error: No se encontraron datos de conversión en la respuesta del API')
        else:
            print(f'Error: Codigo de estado: {respuesta.status_code}')
    except ValueError as e:
        print("Error: ", e)
    texto_json = json.dumps(datos, indent=2)
    with open("conversion.txt", "a") as archivo:
        archivo.write("\n")
        archivo.write(texto_json)
        archivo.write("\n")
    print("\n")
```

Se notó también que la impresión del menú está incorrecta, innecesariamente imprimiendo la palabra “print”, lo cual sucedió por la migración de un simple print a una función. Se corrigió:

```
def menu():
    print("""###Global###Currency###Masters###
###         ###           ###          ###
###         ###           ###          ###
###         ###           ###          ###
###         ###           ###          ###
1. Consultar tipos de monedas"
2. Convertir fuente a blanco
3. Consultar tasas de conversiones recientes
4. Salir""")
```

Se creó un segundo script llamado “GCMStats”, con la idea de que permitiría acceder a los archivos creados por el script principal (GlobalCurrencyMasters.py) e imprimir las tasas de cambio de las monedas convertidas, y para crear estadísticas.

9/5/25

Se implementó la parte de consulta de tasas en el script GCMStats.

```
import json
import numpy as np
import statistics as stats

def rates(ruta_archivo):
    with open(ruta_archivo, 'r') as archivo:
        contenido = archivo.read()
```

```
    bloques = [b for b in contenido.strip().split('\n\n') if b]
    #b for b - creará un lista solo con los bloques no vacíos en caso de lineas vacías
    #.split('\n\n') divide el texto en partes si existe una linea en blanco entre dos diccionarios
    #contenido.strip() elimina los saltos de linea en el principio y final del archivo

    for i, bloque in enumerate(bloques, 1):
        try:
            data = json.loads(bloque)
            print(f"Tasa {i}: {data.get('rate')}")
        except json.JSONDecodeError as e:
            print(f"Error al decodificar bloque {i}: {e}")

rates("conversion.txt")
```

Lo que hace este código es leer los contenidos del archivo “conversion.txt” como diccionario, e imprimir cada tasa de cambio.

Actualmente, en el momento de escribir esta entrada, el archivo de texto se ve así:


```
{
  "date": "2025-05-08 18:00:00+00",
  "from": "MXN",
  "to": "USD",
  "rate": "0.051105",
  "givenAmount": "450.0",
  "convertedAmount": "22.997"
}

{
  "date": "2025-05-08 18:00:00+00",
  "from": "USD",
  "to": "MXN",
  "rate": "19.567",
  "givenAmount": "15.0",
  "convertedAmount": "293.512"
}

{
  "date": "2025-05-09 23:00:00+00",
  "from": "PKR",
  "to": "CHILLGUY",
  "rate": "0.046095",
  "givenAmount": "340.0",
  "convertedAmount": "15.672"
}

{
  "date": "2025-05-09 23:00:00+00",
  "from": "CHILLGUY",
  "to": "MXN",
  "rate": "1.4955",
  "givenAmount": "600.0",
  "convertedAmount": "897.282"
}

{
  "date": "2025-05-09 23:00:00+00",
  "from": "MXN",
  "to": "RUB",
  "rate": "4.2433",
  "givenAmount": "500.0",
  "convertedAmount": "2121.630"
}
```

Al ejecutar GCMStats.py, se imprime cada tasa encontrada en el archivo:

```
Tasa 1: 0.051105
Tasa 2: 19.567
Tasa 3: 0.046095
Tasa 4: 1.4955
Tasa 5: 4.2433
```

Para incorporar el uso de abrir un archivo Python desde otro, se decidió crear una función que permite abrir el código de GCMStats.py mediante el script principal, GlobalCurrencyMasters.py. Se incorporó como una opción más en el menú principal.

```
def open_in_idle(file):
    idle_path = os.path.join(sys.base_prefix, 'Lib', 'idlelib', 'idle.pyw')
    command = [sys.executable, idle_path, file]
    subprocess.Popen(command)

def consultarTasas():
    open_in_idle("GCMStats.py")

def salir():
    print("Adios!")
    return False

def main():
    opciones ={
        1: consultarMonedas,
        2: convertirMonedas,
        3: consultarTasas,
        4: salir
    }
    menu()
    banderaMenu = True
    while banderaMenu:
        try:
            op = int(input("Elige el número de la opción que desea realizar: "))
            elegida = opciones.get(op)
            if elegida:
                resultado = elegida()
                if resultado is False:
                    banderaMenu = False
            else:
                print("Error: Opción inválida.")
        except ValueError:
            print("Error: Debes ingresar un número.")
main()
```

Hasta ahora, la función consultarTasas() simplemente abre el archivo GCMStats.py en IDLE, pero no corre su código. Como objetivo, tenemos pensado hacer que se ejecute el script de tasas mediante el script principal.

12/5/2025

Implementamos la parte de graficarComparacionMonedas()

```
def graficarComparacionMonedas():
    monedas = ['USD', 'EUR', 'MXN', 'JPY', 'ARS']
    colores = ['orange', 'red', 'darkorange', 'magenta', 'skyblue']
    valores = {}

    url = f"https://api.currencyfreaks.com/v2.0/rates/latest?apikey={apikey}"
```

```
respuesta = requests.get(url)

if respuesta.status_code != 200:
    print(f"Error al obtener datos. Código HTTP: {respuesta.status_code}")
    return

datos = respuesta.json()
rates = datos.get('rates', {})

try:
    mxn_rate = float(rates['MXN'])  # Cuánto vale 1 USD en MXN
except KeyError:
    print("No se encontró la tasa de MXN.")
    return

for moneda in monedas:
    try:
        if moneda == 'MXN':
            valores[moneda] = 1.0
        else:
            tasa = float(rates[moneda])  # Cuánto vale 1 USD en esa moneda
            valores[moneda] = mxn_rate / tasa  # Valor relativo en MXN
    except KeyError:
        print(f"No se encontró la tasa para {moneda}")
    except ValueError:
        print(f"Tasa inválida para {moneda}")

if len(valores) < 2:
    print("No hay suficientes tasas válidas para graficar.")
    return

cantidades = list(range(1, 101))
plt.figure(figsize=(12, 6))

for i, moneda in enumerate(monedas):
    if moneda in valores:
        conversiones = [cantidad * valores[moneda] for cantidad in cantidades]
        plt.plot(cantidades, conversiones, label=moneda, color=colores[i])

plt.title("Comparación del valor de monedas extranjeras en MXN (1–100 unidades)")
plt.xlabel("Cantidad de moneda extranjera")
plt.ylabel("Valor en pesos mexicanos (MXN)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Lo que hace esta parte es hacer la grafica de la conversion de pesos mexicanos a las monedas que elegimos para comparar el valor de cada moneda a dia de hoy.