In [ ]:
```python
def leer_lineas_de_archivo(ruta_archivo):
  """Lee un archivo y devuelve una lista con las líneas del archivo.
  Args:
      ruta_archivo (str): La ruta del archivo a leer.
  Returns:
      list: Una lista donde cada elemento es una línea del archivo.
  """
  lineas = []
  with open(ruta_archivo, "r") as archivo:
    for linea in archivo:
      lineas.append(linea.strip())
  return lineas


ruta_archivo = r"c:\Evidencias_CASO2A\ips_del_pcap.txt"

# leer_lineas_de_archivo(ruta_archivo)


# print(leer_lineas_de_archivo(ruta_archivo))

lista_ips = leer_lineas_de_archivo(ruta_archivo)

print(lista_ips)
print(len(lista_ips))
```

In [ ]:
```python
# Para obtener Un subconjunto De las direcciones
def obtener_sublista(lista, indice_inicio, numero_d_items=4):
  """Obtiene n elementos de una lista a partir de un índice dado.
  Args:
      lista: La lista de la que se extraerán los elementos.
      indice_inicio: El índice del primer elemento a incluir en la sublista.
  Returns:
      list: Una nueva lista con los elementos extraídos, o una lista vacía si el
  """
  if indice_inicio < 0 or indice_inicio >= len(lista):
    return []
  indice_final = min(indice_inicio + numero_d_items, len(lista))  # Asegurar que
  return lista[indice_inicio:indice_final]


lista_ips_2 = obtener_sublista(lista_ips, 2 , 5)
print(lista_ips_2)
```

In [ ]:
```python
import urllib.request
import csv
import json
from urllib.error import URLError

def obtener_geolocalizacion(lista_ips_2, archivo_salida):
    """
    Obtiene información de geolocalización para una lista de IPs usando ip-api.c
    Args:
        lista_ips: Una lista de direcciones IP en formato de cadena.
        archivo_salida: La ruta del archivo CSV donde se guardarán los resultado
    """
    base_url = "http://ip-api.com/json/"
    with open(archivo_salida, 'w', newline='', encoding='utf-8') as csvfile:
```

```
        fieldnames = ['IP', 'País', 'Ciudad', 'Latitud', 'Longitud', 'Continente
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
        writer.writeheader()
        for ip in lista_ips_2:
            url = base_url + ip
            try:
                with urllib.request.urlopen(url) as response:  # Timeout de 10 s
                    if response.status == 200:
                        data = json.loads(response.read().decode())
                        if data["status"] == "success":
                            writer.writerow({
                                'IP': ip,
                                'País': data.get('country'),
                                'Ciudad': data.get('city'),
                                'Latitud': data.get('lat'),
                                'Longitud': data.get('lon'),
                                'Continente': data.get('continent'),
                                'Distrito': data.get('district'),
                                'Código Postal': data.get('zip'),
                                'Moneda': data.get('currency'),
                                'ISP': data.get('isp'),
                                'Organización': data.get('org'),
                                'AS': data.get('as'),
                                'Nombre AS': data.get('asname'),
                                'Reverse DNS': data.get('reverse'),
                                'Móvil': data.get('mobile'),
                                'Proxy': data.get('proxy'),
                                'Hosting': data.get('hosting')
                            })
                        else:
                            # Si el estado no es "success", escribir None para l
                            writer.writerow({'IP': ip} | {key: None for key in f
                    else:
                        print(f"Error al obtener información para la IP {ip}: {r
                        writer.writerow({'IP': ip} | {key: None for key in field
            # Manejar errores específicos
            # except timeout:
                # print(f"Tiempo de espera agotado para la IP {ip}")
                # writer.writerow({'IP': ip} | {key: None for key in fieldnames[
            except URLError as e:
                print(f"Error en la solicitud para la IP {ip}: {e.reason}")
                writer.writerow({'IP': ip} | {key: None for key in fieldnames[1:
            except Exception as e:  # Atrapar otros errores
                print(f"Error al procesar la IP {ip}: {e}")
                writer.writerow({'IP': ip} | {key: None for key in fieldnames[1:
```

In [ ]:

```
# Opcion desde colab

from google.colab import drive

ruta_archivo = 'ips-1.txt'
def escribir_ips_a_archivo(lista_ips, ruta_archivo):
    """Escribe una lista de direcciones IP en un archivo de texto en Google Driv
    Args:
        lista_ips (list): Una lista de direcciones IP.
        ruta_archivo (str): El nombre del archivo de texto de salida (e.g., 'arc
    """
    drive.mount('/content/drive')  # Monta Google Drive
    ruta_completa = f"/content/drive/MyDrive/LISP/{ruta_archivo}"
    with open(ruta_completa, 'w') as archivo:
```

```
        for ip in lista_ips:
            archivo.write(ip + '\n')
    print(f"Archivo guardado en Google Drive: {ruta_completa}")
```

In [ ]:
```
escribir_ips_a_archivo(lista_ips, ruta_archivo)
```

In [ ]:
```
obtener_geolocalizacion(lista_ips , archivo_salida)
```