Guía de laboratorio

Laboratorio de Contenido 1: Merge Sort en análisis de logs de seguridad

Explicación básica
Merge Sort es un algoritmo basado en la técnica *Divide y Vencerás*. Divide el arreglo en mitades recursivamente, ordena cada mitad y luego las fusiona manteniendo el orden. Tiene complejidad O(n log n) en todos los casos y es estable.

Objetivos

- Comprender el funcionamiento recursivo de Merge Sort.
- Analizar su complejidad temporal y espacial.
- Aplicarlo al ordenamiento de registros de seguridad.

Caso práctico
Un sistema SOC recibe registros con marcas de tiempo desordenadas y necesita organizarlos cronológicamente para análisis forense.

Problema
Ordenar la siguiente lista de timestamps (en segundos):
[45, 12, 78, 34, 23, 90, 11]

Ejemplo paso a paso
División:
[45,12,78] [34,23,90,11]
Subdivisión:
[45] [12,78]
[34,23] [90,11]
Ordenar y fusionar:
[12,45,78]
[11,23,34,90]
Resultado final:
[11,12,23,34,45,78,90]

Implementación en Python

```
def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left = merge_sort(arr[:mid])
```

```
    right = merge_sort(arr[mid:])

    return merge(left, right)

def merge(left, right):
    result = []
    i = j = 0

    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1

    result.extend(left[i:])
    result.extend(right[j:])

    return result

datos = [45, 12, 78, 34, 23, 90, 11]
print(merge_sort(datos))
```

Conclusiones

Merge Sort garantiza rendimiento estable incluso en grandes volúmenes de datos. Es ideal cuando la estabilidad es crítica, como en el ordenamiento de logs con misma prioridad o misma marca temporal.