OBSERVACIONES DEL LA PRACTICA

Sergio Soler Garzón Cod 202310103
Juan David Galan Vargas Cod 202111470

Preguntas de análisis

1) ¿Qué relación encuentra entre el número de elementos en el árbol y la altura del árbol?

   Rta: En principio debería de seguir una fórmula de logaritmo en base 2, la cual para los datos dados es de aproximadamente 10. Por lo tanto, podemos inferir que el árbol presenta un alto grado de desbalanceo.

   - Elementos del árbol: 1177
   - Altura del árbol: 29

2) ¿Si tuviera que responder esa misma consulta y la información estuviera en tablas de hash y no en un BST, cree que el tiempo de respuesta sería mayor o menor? ¿Por qué?

Rta: En el caso de responder esta consulta con una tabla de hash, se dependería en mayor medida de la llave de esta; si la llave es la fecha de cada crimen, sería necesario recorrer todos los crímenes N sino está organizada, o sería necesario implementar un algoritmo de ordenamiento que aumentaría la complejidad de la búsqueda. Por otro lado, si llave es el tipo de crimen, también es necesario recorrer los N crímenes. Finalmente, en el caso de lo arboles, en el peor de los casos también es necesario recorrer todos los elementos, sin embargo, debido a que los árboles son ordenados por la fecha, de estos solo se recorrer una fracción de estos, por lo que pese a tener el mismo orden de complejidad N en el peor caso, en la realidad los árboles terminan su búsqueda antes.

3) ¿Qué operación del TAD se utiliza para retornar una lista con la información encontrada en un rango de fechas?

   Rta: La siguiente imagen contiene la función en sí, que permite obtener la información dentro de un rango de fechas.

```
def values(map, keylo, keyhi):
    """
    Retorna todas los valores del arbol que se encuentren entre
    [keylo, keyhi]

    Args:
        map: La tabla de simbolos
        keylo: limite inferior
        keylohi: limite superiorr
    Returns:
        Las llaves en el rago especificado
    Raises:
        Exception
    """
    return map['datastructure'].values(map, keylo, keyhi)
```

Y en la ultima imagen la funcion 'getCrimesByRange' es quien recorre cada elemento para sacar el tamaño

```
def getCrimesByRange(analyzer, initialDate, finalDate):
    """
    Retorna el numero de crimenes en un rago de fechas.
    """
    lst = om.values(analyzer['dateIndex'], initialDate, finalDate)
    totcrimes = 0
    for lstdate in lt.iterator(lst):
        totcrimes += lt.size(lstdate['lstcrimes'])
    return totcrimes
```