# **El análisis de complejidad para los dos algoritmos no recursivos más largos y presenta su complejidad en Notación Big O y Complejidad Espacial**

**análisis de addLast(Data<K,V> data,int index) ubicado en la clase HashTable en el paquete model**

Texto

Descripción generada automáticamente

|  |  |  |
| --- | --- | --- |
| # | addLast(Data<K,V> data,int index) | Número de veces |
| 2. | if(lista[index]==null){ | 1 |
| 3. | lista[index]=data; | 1 |
| 4. | lista[index].setNext(data); | 1 |
| 5. | lista[index].setPrev(data); | 1 |
| 6. | }else{ | 1 |
| 7. | Data<K,V> tail= lista[index].getPrev(); | 1 |
| 8. | tail.setNext(data); | 1 |
| 9. | data.setNext(lista[index]); | 1 |
| 10. | lista[index].setPrev(data); | 1 |
| 11. | data.setPrev(tail);} | 1 |
| 12. | size++;} | 1 |

T(n)=1+1+1+1+1+1+1+1+1+1+1+1=12

**Complejidad: O(1)**

Complejidad Espacial:

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo | Variable | Tamaño de 1 valor | Cantidad |
| Entrada | Data, index |  | 1 |
| Auxiliar | Lista[] | 512 | n |
| Auxiliar | size | 32 | 1 |
|  |  |  |  |
|  |  |  |  |

Complejidad Espacial Total = Entrada+Auxuliar+ Auxiliar= 2+n: O(n)

**Análisis de eliminate(K key) ubicado en la clase HashTable en el paquete model**

**Texto

Descripción generada automáticamente**

|  |  |  |
| --- | --- | --- |
| # | eliminate(K key) | Número de veces |
|  | int call = getHashIndex(key); | 1 |
|  | Data<K, V> lista[call]; | 1 |
|  | while (aux != null) { | n+1 |
|  | if (aux.getKey() == key) { | n |
|  | Data<K, V> prev = aux.getPrev(); | n |
|  | Data<K, V> next = aux.getNext(); | n |
|  | if (queue.getKey() == lista[call]) { | n |
|  | lista[call] = next; | n |
|  | } else if (next != null) { | n |
|  | next.setPrev(prev); | n |
|  | } else if (prev != null) { | n |
|  | prev.setNext(next);}} | n |
|  | queue=queue.getNext();}} | n |

K(n)= 1+1+n+n+n+n+n+n+n+n+n+n=3+11n

Complejidad : O(n)

Auxiliar

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo | Variable | Tamaño de 1 valor | Cantidad |
| Entrada | key | 256 | 1 |
| Auxiliar | call | 112 | 1 |
| Auxiliar | queue | 112 | n |
| Auxiliar | aux | 512 | 1 |
| Auxiliar | Prev | 512 | 1 |
| Auxiliar | next | 512 | 1 |
|  |  |  |  |

Complejidad Espacial Total = Entrada+Auxuliar+ Auxiliar= 5+n: O(n)