

Análisis de complejidad temporal y espacial.

```
public int convertStringToNumbers(K key) {  
    int values = 0;  
    String s = key.toString();  
    for (int i = 0; i < s.length() ; i++)  
        values += s.charAt(i) * (i + 1);  
    return values;  
}
```

Instrucciones	Veces que se repite (Notación Big O)
1. int values = 0;	1
2. String s = key.toString();	1
3. for (int i = 0; i < s.length() ; i++)	n + 1
4. values += s.charAt(i) * (i + 1);	n
5. return values;	1

Tipo	Variable	Tamaño de 1 valor atómico	Cantidad de valores atómicos
Entrada	key	64 bits	1
Auxiliar	s	64 bits	1
	i	32 bits	1
Salida	values	32 bits	1

Complejidad espacial total = Entrada + Auxiliar + Salida = 1 + 2 + 1 = 4 = $O(4) = O(1)$.

Complejidad espacial auxiliar = Auxiliar = 2 = $O(2) = O(1)$.

Complejidad espacial auxiliar + salida = Auxiliar + Salida = 2 + 1 = 3 = $O(3) = O(1)$.

```

public void enqueue(T data) {
    Node<T> element = new Node<>(data);
    if( isEmpty() ) {
        first = element;
        last = first;
    } else {
        Node<T> temporal = first;

        while( temporal.getNext() != null )
            temporal = temporal.getNext();

        temporal.setNext(element);
        last = element;
    }
}

```

Instrucciones	Veces que se repite (Notación Big O)
1. Node<T> element = new Node<>(data);	1
2. If(isEmpty()) {	1
3. first = element;	1
4. last = first;	1
5. } else {	1
6. Node<T> temporal = first;	1
7. while(temporal.getNext() != null)	n + 1
8. temporal = temporal.getNext();	n
9. temporal.setNext(element);	1
10. last = element;	1

Tipo	Variable	Tamaño de 1 valor atómico	Cantidad de valores atómicos
Entrada	data	64 bits	1
Auxiliar	temporal	64 bits	1
Salida	Ninguno	0	0

Complejidad espacial total = Entrada + Auxiliar + Salida = 1 + 1 + 0 = 2 = $O(2)$ = $O(1)$.

Complejidad espacial auxiliar = 1 = $O(1)$.

Complejidad espacial auxiliar + salida = Auxiliar + Salida = 1 + 0 = 1 = $O(1)$.