**complejidad temporal**

| **Operación** | **Complejidad temporal** | **Número de veces que se ejecuta** | **Contribución total al tiempo de ejecución** |
| --- | --- | --- | --- |
| Insertar valor booleano en firstClass | O(1) | 30 \* 6 = 180 | O(180) = constante |
| Insertar valor booleano en businessClass | O(1) | 10 \* 6 = 60 | O(60) = constante |
| Agregar cola de prioridad en economyClass | O(1) | 15 \* 6 = 90 | O(90) = constante |
| Total | O(1) | 270 | O(270) = constante |

Método revisado:

public static void initializeSeats(MyMap<String, Boolean> firstClass,  
 MyMap<String, Boolean> businessClass,  
 MyMap<String, MyPriorityQueue<Date>> economyClass) {

| **Operación** | **Complejidad temporal** | **Número de veces que se ejecuta** | **Contribución total al tiempo de ejecución** |
| --- | --- | --- | --- |
| Asignación de valor a element | O(1) | 1 | O(1) |
| Cálculo de la mitad de size | O(1) | 1 | O(1) |
| Comprobación de la condición del bucle | O(1) | log2(n) | O(log n) |
| Cálculo del índice del hijo izquierdo | O(1) | log2(n) | O(log n) |
| Asignación de valor a child | O(1) | log2(n) | O(log n) |
| Cálculo del índice del hijo derecho | O(1) | log2(n) | O(log n) |
| Comparación de child y elements[rightChildIndex] | O(1) | log2(n) | O(log n) |
| Asignación de valor a childIndex | O(1) | log2(n) | O(log n) |
| Comparación de element y child | O(1) | log2(n) | O(log n) |
| Asignación de valor a elements[index] | O(1) | log2(n) | O(log n) |
| Asignación de valor a index | O(1) | log2(n) | O(log n) |
| Asignación de valor a elements[index] | O(1) | 1 | O(1) |
| Total | O(log n) | n | O(n log n) |

private void downHeap(int index) {