**Cola de prioridad:**

***Agregar:***

Mejor caso:

Que los elementos que agreguen sean cada vez mas grandes haciendo que solo se modifique el primer nodo de la lista.

Complejidad:

O(1)

Peor caso:

Que los elementos que agreguen sean cada vez mas pequeños, pues aca ya tocaria recorrer toda la lista para agregarlos de ultimos.

Complejidad:

O(n)

***Eliminar maximo:***

Mejor caso:

Que la cola de prioridad le de prioridad al elemento mayor poniendolo de primero.

Complejidad:

O(1)

Peor caso:

Que la cola de prioridad le de prioridad al elemento menor haciendo que el mayor este de ultimo. Esto hace que se recorra toda la lista al eliminar el mayor.

Complejidad:

O(1)

**Heap:**

***Agregar:***

Mejor caso:

Que los elementos que agregue sean cada vez mas pequeños para evitar usar el swim. En este caso al agregar el nuevo hijo el heap sigue ordenado.

Complejidad:

O(1)

Peor caso:

Que los elementos que agregue sean cada vez mas grandes obligando a usar el swim.

Complejidad:

O(log n)

***Eliminar Maximo:***

Mejor caso:

Que el heap le de prioridad al elemento mayor poniendolo de primero.

Complejidad:

O(1)

Peor caso:

Que el heap le de prioridad al elemento menor haciendo que el mayor este de ultimo. Esto hace que se recorra toda la lista al eliminar el mayor.

Complejidad:

O(log n)

Para Cola de Prioridad: Tiempos en milisegundos para agregar y eliminar.

|  |  |  |
| --- | --- | --- |
|  | Agregar | Eliminar |
| 50000 | 8 | 0 |
| 100000 | 21 | 0 |
| 150000 | 30 | 0 |
| 200000 | 29 | 1 |
| 250000 | 37 | 0 |
| 300000 | 35 | 0 |
| 350000 | 40 | 0 |
|  |  |  |
|  |  |  |

Para Heap: Tiempos en milisegundos para agregar y eliminar.

|  |  |  |
| --- | --- | --- |
|  | Agregar | Eliminar |
| 50000 | 2 | 1 |
| 100000 | 2 | 3 |
| 150000 | 2 | 2 |
| 200000 | 1 | 3 |
| 250000 | 1 | 3 |
| 300000 | 2 | 4 |
| 350000 | 1 | 3 |