1. array：大小固定不能改变，支持随机访问。
2. vector：动态数组，支持随机访问。
3. list：双向循环链表，不支持随机访问。
4. deque：多个连续的存储快，支持随机访问。
5. set和multiset：红黑树(平衡二叉树)，元素有序(支持二分查找)
6. map和multimap：红黑树，元素有序
7. 哈希容器，unordered\_set, undordered\_multiset, unordered\_map和unordered\_multimap：哈希表，内部是一个很大的数字，元素无序。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 插入/删除 | 访问/修改 | 查找 |
| array | X |  |  |
| vector | 尾部为：  其它为： |  |  |
| list |  | 头尾为：  其它为： |  |
| deque | 头尾为：  其它为： |  |  |
| set/multiset |  | X(只能先删除再插入) |  |
| map/multimap |  | X(只能先删除再插入) |  |
| unordered\_set/ undordered\_multiset unordered\_map/unordered\_multimap | 最坏：  其它： | X | 最坏：  其它： |

注：哈希容器中，最坏的情况假如所有元素都被散列到同一位置，分离链表法解决冲突时，就会出现的时间复杂度

容器适配器：

priority\_queue：二叉堆，元素有序，不支持随机访问。

删除堆顶元素时间复杂度为：，删除其它位置的元素无意义(Java中有类似的方法，时间复杂度为)。

插入元素的时间复杂度为：，先将元素插入堆中数组的最后，然后进行调整。

访问堆顶元素的时间复杂度为 ：。

堆不支持修改操作，如果希望修改堆顶元素，只有先删除掉，然后插入一个新的值。

创建堆（自底向上建堆）的时间复杂度为：而不是。并非是将n个元素插入堆中数组，从而计算出。

二叉堆是一个完全二叉树，假设有n个元素，其高度为h，则。如下图所示的二叉堆，总共有n=7个元素，高度为2，



最底层元素的高度为0，个数为，不需要往下调整，次底层元素的高度为1，个数为，最多需要调整的次数也为，则



其中s表示高度，高度为0的不需要下滤，高度为1的最多下滤1次，高度为2的最多下滤2次，则：



由得：



将带入上式，可得：



所以建堆的时间复杂度为：