Container	Type	Avg. Time Complexity				Possible implementation	Comments
		Insertion	Deletion	Access	Search		
array		/	/	O(1)	O(n)	Wrapper around contiguous C-style array	Fixed in size
vector	Sequence	At the end: amortized O(1) O(n) elsewhere	At the end: amortized O(1) O(n) elsewhere	O(1)	O(n)	Wrapper around contiguous C-style array	• Insertion requires copy of all elements after vector • Size is dynamic and increases by power of 2 every time its full -> requires a full copy
deque (double-ended queue)		At the beginning & end: O(1) O(n) elsewhere	At the beginning & end: O(1) O(n) elsewhere	O(1)	O(n)	sequence of individually allocated fixed-size arrays	• Expanding storage is cheaper than vector since the block of memory does not need to be continous.
forward_list		O(1)	O(1)	O(n)	O(n)	singly linked list	• More space efficent than list if bidirectionality is not needed
list		O(1)	O(1)	O(n)	O(n)	doubly linked list	• Bidirectional
set		O(log(n))	O(log(n))	O(log(n))	O(log(n))	red-black tree	• contains a sorted set of unique objects
multiset		O(log(n))	O(log(n))	O(log(n))	O(log(n))	threaded red-black tree	• contains a sorted set of non-unique objects
map	Associative	O(log(n))	O(log(n))	O(log(n))	O(log(n))	red-black tree	• contains key-value pairs with unique keys
multimap		O(log(n))	O(log(n))	O(log(n))	O(log(n))	threaded red-black tree	• contains a sorted list of key-value pairs, while permitting multiple entries with the same key.
unordered_set	unordered associative	O(1)	O(1)	O(1)	O(1)	hash-set	• contains a set of unique objects • uses hash function • stores multiple in buckets referring to hashes • keeps track of load factor and doubles its number of buckets if necessary
unordered_multiset		O(1)	O(1)	O(1)	O(1)	hash-set	• same as unordered_set but allows non unique objects
unordered_map		O(1)	O(1)	O(1)	O(1)	hash-table	• same as unordered_set but stores key value pairs
unordered_multimap		O(1)	O(1)	O(1)	O(1)	hash-table	• same as unordered_map but allows non unique keys
stack		O(1)	O(1)	O(n)	O(n)	Wrapper on provided underlying container (default = deque)	• LIFO (last-in, first-out) data structure
queue	sequence adapters	O(1)	O(1)	O(n)	O(n)	Wrapper on provided underlying container (default = deque)	• FIFO (first-in, first-out) data structure
priority_queue		O(1)	O(1)	O(n)	O(n)	Wrapper on provided underlying container (default = deque)	• provides constant time lookup of the largest (by default) element