**1. Containers**

**Sequential Containers**

* **vector**
  + Keywords: push\_back, pop\_back, insert, erase, clear, size, capacity, resize, empty, front, back, begin, end, emplace, emplace\_back, shrink\_to\_fit
* **deque**
  + Keywords: push\_front, push\_back, pop\_front, pop\_back, insert, erase, clear, size, resize, empty, front, back, begin, end, emplace, emplace\_front, emplace\_back
* **list**
  + Keywords: push\_front, push\_back, pop\_front, pop\_back, insert, erase, clear, size, empty, front, back, begin, end, emplace, emplace\_front, emplace\_back, sort, reverse, merge, splice, unique
* **array**
  + Keywords: size, at, empty, front, back, data, begin, end, fill, swap
* **forward\_list**
  + Keywords: push\_front, pop\_front, insert\_after, erase\_after, clear, size, empty, front, emplace\_after, sort, reverse, merge, splice\_after, unique

**Associative Containers**

* **set**
  + Keywords: insert, erase, find, count, clear, size, empty, begin, end, lower\_bound, upper\_bound, equal\_range
* **multiset**
  + Keywords: Same as set with allowance for duplicate elements.
* **map**
  + Keywords: insert, erase, find, count, clear, size, empty, begin, end, at, operator[], lower\_bound, upper\_bound, equal\_range
* **multimap**
  + Keywords: Same as map with allowance for duplicate keys.

**Unordered Containers**

* **unordered\_set**
  + Keywords: insert, erase, find, count, clear, size, empty, begin, end, bucket\_count, load\_factor, rehash, reserve
* **unordered\_multiset**
  + Keywords: Same as unordered\_set with allowance for duplicate elements.
* **unordered\_map**
  + Keywords: insert, erase, find, count, clear, size, empty, begin, end, at, operator[], bucket\_count, load\_factor, rehash, reserve
* **unordered\_multimap**
  + Keywords: Same as unordered\_map with allowance for duplicate keys.

**Container Adapters**

* **stack**
  + Keywords: push, pop, top, size, empty
* **queue**
  + Keywords: push, pop, front, back, size, empty
* **priority\_queue**
  + Keywords: push, pop, top, size, empty

**2. Iterators**

* **Types**: begin, end, rbegin, rend, cbegin, cend, crbegin, crend
* **Iterator Adaptors**: reverse\_iterator, back\_insert\_iterator, front\_insert\_iterator, insert\_iterator

**3. Algorithms**

* **Searching and Sorting**
  + sort, stable\_sort, partial\_sort, nth\_element, binary\_search
* **Permutation**
  + next\_permutation, prev\_permutation
* **Numeric**
  + accumulate, partial\_sum, adjacent\_difference
* **Set Operations**
  + set\_union, set\_intersection, set\_difference, set\_symmetric\_difference, includes
* **Heap**
  + make\_heap, push\_heap, pop\_heap, sort\_heap
* **Partitioning**
  + partition, stable\_partition, is\_partitioned, partition\_copy
* **Searching**
  + find, find\_if, find\_if\_not, adjacent\_find, search, search\_n
* **Modification**
  + copy, copy\_if, move, swap, swap\_ranges, transform, replace, replace\_if, remove, remove\_if, unique
* **Other**
  + count, count\_if, equal, mismatch, lexicographical\_compare, for\_each

**4. Utilities**

* **Pair and Tuple**
  + pair, make\_pair, tie, tuple, make\_tuple, get
* **Functional**
  + function, bind, mem\_fn, not1, not2, ref, cref
* **Type Traits**
  + is\_same, enable\_if, conditional, remove\_reference, add\_const