1. Немодифицирующие операции.
   * for\_earch() - выполняет операции для каждого элемента последовательности
   * find() - находит первое вхождение значения в последовательность
   * find\_if() - находит первое соответствие предикату в последовательности
   * count() - подсчитывает количество вхождений значения в последовательность
   * count\_if() - подсчитывает количество выполнений предиката в последовательности
   * search() - находит первое вхождение последовательности как подпоследовательности
   * search\_n()- находит в последовательности подпоследовательность, состоящую из n повторений и возвращает её первое вхождение.
2. Модифицирующие операции.
   * copy() - копирует последовательность, начиная с первого элемента
   * swap() - меняет местами два элемента
   * replace() - заменяет элементы с указанным значением
   * replace\_if() - заменяет элементы при выполнении предиката
   * replace\_copy() - копирует последовательность, заменяя элементы с указанным значением
   * replace\_copy\_if() - копирует последовательность, заменяя элементы при выполнении предиката
   * fill() - заменяет все элементы данным значением
   * remove() - удаляет элементы с данным значением
   * remove\_if() - удаляет элементы при выполнении предиката
   * remove\_copy() - копирует последовательность, удаляя элементы с указанным значением
   * remove\_copy\_if() - копирует последовательность, удаляя элементы при выполнении предиката
   * reverse() - меняет порядок следования элементов на обратный
   * random\_shuffle() - перемещает элементы согласно случайному равномерному распределению ("тасует" последовательность)
   * transform() - выполняет заданную операцию над каждым элементом последовательности
   * unique() - удаляет равные соседние элементы
   * unique\_copy() - копирует последовательность, удаляя равные соседние элементы
3. Сортировка.
   * sort() - сортирует последовательность с хорошей средней эффективностью
   * partial\_sort() - сортирует часть последовательности
   * stable\_sort() - сортирует последовательность, сохраняя порядок следования равных элементов
   * lower\_bound() - находит первый элемент, меньший чем заданное значение
   * upper\_bound() - находит первый элемент, больший чем заданное значение
   * binary\_search() - определяет, есть ли данный элемент в отсортированной последовательности
   * merge() - сливает две отсортированные последовательности
4. Работа с множествами.
   * includes() - проверка на вхождение
   * set\_union() - объединение множеств
   * set\_intersection() - пересечение множеств
   * set\_difference() - разность множеств
5. Минимумы и максимумы.
   * min() - меньшее из двух
   * max() - большее из двух
   * min\_element() - наименьшее значение в последовательности
   * max\_element() - наибольшее значение в последовательности
6. Перестановки.
   * next\_permutation() - следующая перестановка в лексикографическом порядке
   * prev\_permutation() - предыдущая перестановка в лексикографическом порядке