**12. Стандартные контейнерные типы. Последовательные контейнеры. Оценка сложности основных операций.**

**Стандартные последовательные контейнерные типы –** структуры данных, представляющие собой последовательность значение, определенным образом связанные.

**Виды:**

* **Массив(std::array) #include <array>**
  + Фиксированная по длине последовательность, неизменяемая.
  + Хранится в стеке непрерывным блоком.
  + Изображение выглядит как текст, снимок экрана, Шрифт, линия

    Автоматически созданное описаниеНе путай array со стандартным массивом(int arr[3] = {1, 2, 3}). Array имеет расширенный функционала, в отличие простого упрощенного(напр, arr.back() определение последнего элемента)
* **Вектор(std::vector) #include <vector>**
  + Динамический последовательность, позволяющая добавлять элементы в любое место.
  + Изображение выглядит как текст, Шрифт, снимок экрана, линия

    Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт

    Автоматически созданное описаниеХранится в куче непрерывным блоком. При изменении размера находит другое свободное пространство и переносится(происходит релокация) туда, а старая последовательность затирается.
* **Двусвязные список(std::list)**
  + Динамический последовательность, позволяющая добавлять элементы только в начало. Представляет собой последовательность узлов, каждый из которых помимо значения хранит указатель на предыдущий и следующий элементы.
  + Хранится в куче, узлы могут быть разбросаны в разных участках памяти, из-за наличия указателей.
  + Такая структура позволяет эффективно добавлять(удалять) элементы в любое место без необходимости смещать всю последовательность. Минус такой структуры в необходимости больше памяти, для хранения указателей.
  + Изображение выглядит как текст, дисплей, снимок экрана, программное обеспечение

    Автоматически созданное описаниеИз-за того, что список хранит только указатели на начало и конец, обращение по индексу невозможно.
* **Односвязный список()**
  + Динамический последовательность, позволяющая добавлять элементы только в начало. Как двусвязный список, только связь идет на следующий элемент, из-за чего добавлять элементы можно только в начало и середину. Т.к. конечный элемент не имеет указателя ни на что.
  + Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

    Автоматически созданное описаниеХранится в куче, узлы могут быть разбросаны в разных участках памяти, из-за наличия указателей.
* **Двусторонняя очередь(std::deque).** 
  + Изображение выглядит как диаграмма, текст, линия, Технический чертеж

    Автоматически созданное описаниеДинамический последовательность, представляющая собой страницы, которых хранятся кусочно-непрерывно. Т.е. последние и первые элементы хранятся в отдельных страницах, чтобы мочь эффективно и быстро добавлять новый элемент, без необходимости релокации и копировании старых элементов.
  + Изображение выглядит как текст, Шрифт, снимок экрана

    Автоматически созданное описаниеХранится в куче страницами, не обязательно цельным блоком страниц. По сути дела, работает схоже с вектором.

**Сложность:**

* **Массив:** Доступ по индексу: O(1)
* **Вектор:** 
  + Вставка/Удаление в конец: O(1)
  + Вставка/удаление элемента в середину/начало: O(n)
  + Доступ по индексу : O(1)
* **Двусвязный список:**
  + Вставка/удаление элемента в любое место: O(1)
  + Доступ по индексу : Отсутствует(только полный перебор, который O(n))
* **Односвязный список**
  + Вставка/удаление элемента в начало: O(1)
  + Вставка/удаление элемента в середину/конец: O(n)
  + Доступ к элементу по индексу: Отсутствует (требуется обход списка)
* **Двусторонняя очередь:**
  + Вставка/удаление элемента в начало/конец: O(1)
  + Вставка/удаление элемента в середину: O(n)
  + Доступ к элементу по индексу: O(1)