1. Множество:
   * Основные операции: добавление элемента, удаление элемента, поиск элемента.
   * Сложность выполнения:
     + Добавление элемента: в худшем случае O(n) (если нужно перехешировать всю таблицу), в среднем O(1).
     + Удаление элемента: в худшем случае O(n) (если нужно перехешировать всю таблицу), в среднем O(1).
     + Поиск элемента: в худшем случае O(n) (если все элементы попали в одну ячейку), в среднем O(1).
   * Подходит для задач, где требуется быстрый поиск и удаление элементов без учета их порядка.
2. Массив:
   * Основные операции: доступ (чтение и запись) к элементу по индексу, добавление элемента (в начало, середину или конец массива), удаление элемента (из начала, середины или конца массива).
   * Сложность выполнения:
     + Доступ к элементу по индексу: O(1).
     + Добавление элемента: в худшем случае O(n) (если нужно перенести все элементы), в среднем O(n/2) (также нужно учитывать сдвиг элементов).
     + Удаление элемента: в худшем случае O(n) (если нужно перенести все элементы), в среднем O(n/2) (также нужно учитывать сдвиг элементов).
   * Подходит для задач, где известен заранее максимальный размер последовательности и требуется быстрый доступ к элементам по индексу.
3. Стек:
   * Основные операции: добавление элемента (помещение на вершину стека), удаление элемента (изъятие из вершины стека), проверка пустоты стека.
   * Сложность выполнения:
     + Добавление элемента: O(1).
     + Удаление элемента: O(1).
     + Проверка пустоты стека: O(1).
   * Подходит для задач, где необходимо придерживаться принципа LIFO (Last-In, First-Out), например, при обработке рекурсивных вызовов, сохранении состояния программы или реализации алгоритма обхода графа в глубину.
4. Очередь:
   * Основные операции: добавление элемента (в конец очереди), удаление элемента (из начала очереди), проверка пустоты очереди.
   * Сложность выполнения:
     + Добавление элемента: O(1).
     + Удаление элемента: O(1).
     + Проверка пустоты очереди: O(1).
   * Подходит для задач, где нужно соблюдать принцип FIFO (First-In, First-Out), например, при обработке задач или запросов, имитации работы системы или реализации алгоритма обхода графа в ширину.
5. Связный список:
   * Основные операции: добавление элемента, удаление элемента, поиск элемента по значению, вставка элемента по индексу, удаление элемента по индексу.
   * Сложность выполнения:
     + Вставка элемента: O(1) - добавление элемента в начало или конец списка.
     + Удаление элемента: O(1) - удаление элемента из начала или конца списка.
     + Поиск элемента по значению: O(n) - линейная сложность, так как требуется просмотреть все элементы списка.
     + Вставка элемента по индексу: O(n) - линейная сложность, так как требуется просмотреть все элементы до нужной позиции.
     + Удаление элемента по индексу: O(n) - линейная сложность, так как требуется просмотреть все элементы до нужной позиции.
   * Подходит для задач, где требуется частое добавление или удаление элементов в начало или конец списка. Он также удобен для задач, где необходимо сохранить порядок элементов и иметь возможность быстро вставлять или удалять элементы по определенным позициям.