**Стек (stack) -**

Это структура данных, в которой элементы добавляются и удаляются в вершине стека. Массив элементов организован по принципу LIFO. Элементы забирают и добавляют сверху. Поскольку операции вставки и извлечения осуществляются только с одного конца массива, то для моделирования стека можно использовать лист, который имеет аналогичные методы.

**Очередь (Queue) -**

Это структура данных, которой добавление элементов возможно лишь в конец очереди возможно лишь в конец очереди, а выборка либо взятие элемента только из начала очереди при этом выбранный элемент из очереди удаляется. Массив элементов организованн по принципу FIFO. Операции и методы немного изменены и также находятся в презентации.

**Дек (deque) -**

Двусвязная очередь. Это структура данных представляющая из себя список элементов в котором добавление новых элементов и удаление существующих производится в обоих концов массива. Это позволяет многие задачи решать непосредственно решать с помощью дека, абстрагируясь до нужной структуры данных. Такие структуры как очередь и стек создаются на основе дека. В питоне он организованн в collection.deque . Поскольку тип дек не является встроенным типом, а реализован в модуле, то перед использованием дека модуль надо импортировать.

Реализация в Питоне:

Встроенная в питон структура данных list поставляется в комплекте с методами для имитации (в чистом виде можно только через коллекцию) операций стек и очереди.

Встроенная структура list может использоваться в качестве стека. Вместо метода push( ) можно использовать append( ) для добавления новых элементов в верхнюю часть стека в то время, как pop( ) удаляет элементы в порядке LIFO. У list() есть несколько недостатков по сравнению с другими структурами данных. Проблема заключается в том, что он может столкнуться по скорости при увеличении размера данных. Если стек становится больше чем блок памяти в котором находится на данный момент, то питон должен сделать некоторое дополнительное выделение памяти, это может привести к тому что некоторые вызовы метода append будут занимать намного больше времени чем другие.

Функция в питон - это объект принимающий аргументы и вовращающий значение. . Функция определяется с помощью инструкции **def** (вызов функции) **func** (имя функции) **(x , y)** (параметры). Инструкция return говорит, что нужно вернуть значение. В функции можно использовать неограниченное количество параметров, но число аргументов должно точно соответствовать числу параметров. Эти параметры представляют собой позиционные аргументы. Параметр - это имя в списке параметров в первой строке определения функции, он получает своё значение при вызове. Аргумент - это реальное значение или ссылка на него, переданная функции при вызове