**Алгоритмы и структуры данных.**

**ШАД.**

Лекция 1.

Модели вычислений.

RAM-модель – модель оперативной памяти, которая состоит из ячеек и в каждой ячейке помещается слово длинной w. То есть мы можем в одну такую ячейку положить число определенной разрядности.

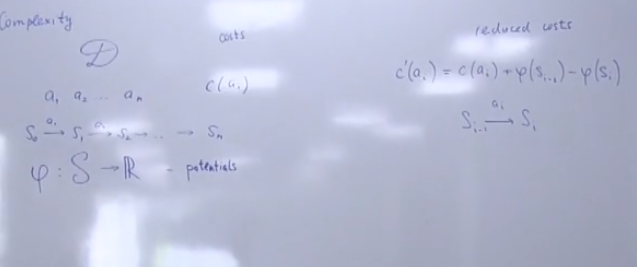
Разрешающие деревья (decision trees) – модель вычислений, отличающаяся от RAM, но позволяющая проводить сравнения и прочее и для которой подходит оценка алгоритмов.

Как эта модель устроена?   
Программа для данной модели – дерево (бинарное – у каждой вершины, кроме листьев, есть по два сына. Есть полное и неполное.). У нас есть бинарный предикат сравнения, он может вернуть нам либо истину, либо ложь. Если вернет истину – идем в правого сына, если ложь – в левого (зависит от знака сравнения). Данный алгоритм не имеет никакой памяти. Единственное его состояние – это текущая вершина. Как только мы попадаем в лист, алгоритм считает, что он закончил работу. Дерево считаем правильным, если для любого неупорядоченного множества и любого набора ключей – этот набор будет отсортирован правильно. Что есть сложность для алгоритмов на дереве? – Это высота дерева – максимальная длина пути от корня до листа.   
Сложность правильных деревьев сортировки Т – Ω(NlogN).  
Всего листьев в дереве минимум N!, так как это число возможных вариантов ответа.  
T – это высота, тогда всего листьев в интервале: 2Т >= листьев >= N! -> T >= log2N! -> По формуле Стирленга: >= Nlog2N. Алгоритм такого вида неуниформен.  
Можно ли сортировать быстрее? Да, для RAM машины есть алгоритм со скоростью N log2log2N.

Сложность работы структуры данных – N \* T(время работы отдельного действия)  
Учетная (амортизированная) сложность – сложность в среднем на очень длинной цепочке операций.   
Есть два определения этой сложности:   
1) Банковский метод. Мы ввели некоторую универсальную валюту, которая символизирует собой элементарное действие. То есть за каждое элементарное действие компьютер требует 1 единицу.

Capacity – общий объем  
size – размер заполненных элементов

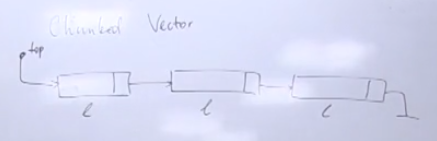
Выбор нового объема бывает:   
по аддитивной стратеги (к старому объему прибавляют константу). Добавление в уже заполненный массив стоит O(n2), так как мы расширяем массив и копируем все предыдущие элементы в новый.  
 и мультипликативная стратегия (старая умножить на некоторую константу). Тут сложность линейная. O(n).

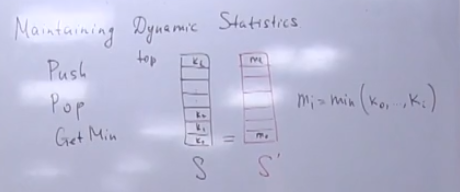
2) Физический метод (метод потенциалов).   
  
Слева – стоимости, справа – приведенные стоимости.  
Фи подбирается.  
 - формула пересчета истинных значений в приведенные.

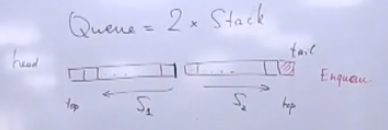
Лекция 2.

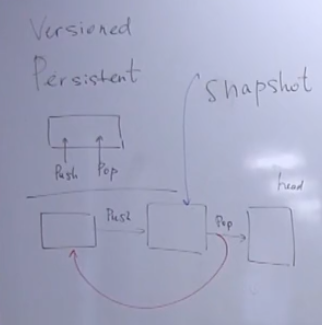
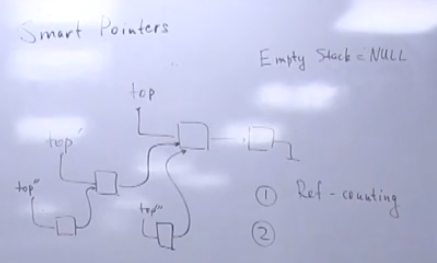
Стэки и очереди.

Хороший стэк выполняет как минимум две операции:   
push(х) – положить элемент наверх   
pop() – взять верхний элемент

Сложность операций: О(1) – учетная сложность.  
Стэк можно устроить вектором, или списком. Для списка очень большой оверхеад по памяти. На адресацию тратится в два раза больше памяти, чем на сами данные.   
Есть выход – гибрид – Chunked Vector.   


Если мы захотим к функциям стэка добавить getMin(), то для этого нужносоздать вспомогательный стэк, в котором будут храниться минимумы.  
  
То есть, каждая mi, это минимум между всеми элементами, что есть.

Очереди.  
У них есть голова и хвост.   
Операции enqueue (добавление в хвост) и dequeuer(получение из головы). Структура – циклическая очередь – это очередь, в которой элементы хранятся по индексам, которые вычисляются по некоторому модулю. При переполнении можно попробовать сделать реолокацию (увеличить размер), а при попытке вынуть что то из пустой очередь – бросать исключение.  
очередь можно сделать на основе двух стэков, который стоят друг к другу спиной.  
  
Если второй стэк пуст, то вероятно все элементы лежат в S1.

Еще одно важное свойство структур данных – Persistence (версионировани).   
Данная структура способна хранить историю своего развития и модификаций. Например, когда мы говорим стэку push и pop, то структура стэка меняется. А для такой структуры данных при каждой команде у нас создается новый экземпляр стуктуры данных. При таком подходе легко сохранить состояние этой структуры. И можно запомнить указатель на каждое состояние.  
  
Данные снэпшоты readonly. Иногда такое ограничение не присутствует.  
В данном случае можем сделать такой стэк.   
То есть мы при добавлении нового элемента создаем «новую» ветку, а при получении элемента мы просто перемещаем указатель на более старую версию.