Домашнее задание 6

Христолюбов Максим, 771

Задание 1

Добавление элемента в стек реализуется через 2 очереди следующим образом: добавляем элемент в 1-ую очередь. Извлечение элемента: перекладываем все данные кроме последнего из 1-ой очереди во 2-ую и извлекаем этот единственный последний элемент из 1-очереди. После этого перекладываем все элементы из 2-ой очереди в первую (или же теперь работаем со 2-ой как с 1-ой).

Задание 2

Троичное дерево можно хранить построчно, то есть сначала стоит корень дерева, потом его потомки слева на право, потом их потомки, причем сначала иду потомки самого левого элемента-родителя, потом среднего, потом правого. Потомки элемента $A[i]$ - $A[3i], A[3i+1], A[3i+2]$. Родитель элемента $A[i]$ - $A\lfloor \frac{i}{3} \rfloor$.

Задание 3

x- элемент, y - следующий за ним по возрастанию в дереве. Т.к. у x нет правого потомка, а $y > x$, то y был добавлен в дерево раньше x, так как в другом случае он должен был бы быть в правом поддереве x. Рассмотрим процесс добавления x в дерево с y. В тех узлах где отправлялся в левое поддерево x тоже будет отправляться в левое поддерево. Так как в дереве нет z, такого что $y > z > x$, то в тех узлах где отправлялся в правое поддерево x тоже будет отправляться в правое поддерево, так как если бы было иначе, то получилось бы что узловой элемент $u > x$, но $y < u$, то есть такое $z = u$ $y > z > x$. Значит, x пройдет по дереву тем же путем что y и отправится в его левое поддерево. После этого x не может ни в каком узле отправится в левое поддерево, так как тогда этот узловой элемент v $y > v > x$, что не возможно. Значит, x дойдет до низа дерева и там запишется, следовательно, y является самым нижним предком х, чей левый дочерний узел так же является предком x или самим x.

Задание 4

Если бы последующая за b вершина c имела бы левого потомка, тогда бы в случае, когда c находится в поддереве (а именно в правом поддереве) b, тогда левый потомок b d был бы $d < b$, $d > a$, что противоречит условию. Если b находится в левом поддереве c, то правый потомок b q такой, что $q > b$, но $q < c$ - противоречие. Если же c и b не в поддеревьях друг друга, то это значит, что существует узел e в котором c ушло в его правое поддерево, а b в левое, тогда, если у c есть левый потомок d, то $d > e > b$, а $c > d > b$, что противоречит условию. Следовательно у c нет левого потомка.

В случае когда b находится в правом поддереве a левый потомок b w такой, что $w < b$ и $w > a$ - противоречие.Если a в левом поддереве b, то если существует правый потомок a s, то $s > a$ и $s < b$, что невозможно. Если они находятся не в поддеревьях друг друга, то существует узел x, в котором a ушло в его левое поддерево, а b в правое, тогда, если существует правый потомок a s, то $s > a$ и $s < b$, что противоречит условию задачи. Следовательно у a нет правого потомка.

Задание 5

а) Если для того, чтобы узнать принадлежит ли x A достаточно выполнить $t = 1$ запросов то сделать это можно единственным образом - если запрос будет единственный запрос - принадлежит ли x A. Т.е. в таблице должно быть n строчек, каждая из которых говорила бы принадлежит ли x A. $s(m,k,1) = n$. Меньше строчек не может быть, так как для каждого элемента алгоритм должен быть способен ответить на вопрос, для этого при $t = 1$ каждому элементу должен соответствовать собственный вопрос.

б) Интуиция подсказывает, что $\log n$, но как доказать это я не знаю.

Задание 6

Если обслуживать клиентов в порядке k_1, $k_2 ... k_n$, то время обслуживания будет равно $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{i} t_{k_j} = \sum\limits_{i=1}^{n}((n+1-i)t_{k_i}) = n(n+1) - \sum\limits_{i=1}^{n} it_{k_i}$. $\sum\limits_{i=1}^{n} it_{k_i}$ будет максимальна если t_{k_i} будут идти по возрастанию. Значит, можно отсортировать клиентов быстрой сортировкой за $\Theta(nlogn)$ и получить последовательность клиентов с минимальным суммарным временем ожидания клиентов.