

МФТИ, ФПМИ  
Алгоритмы и структуры данных, осень 2021  
Семинар №6. Кучи

1. Определите количество вершин на глубине  $k$  в биномиальном дереве порядка  $n$ .
2. За сколько работал бы в биномиальной куче `siftDown`?
3. Вместо бинарной (двоичной) кучи можно рассматривать  $k$ -ичные кучи. В них каждая вершина (кроме листьев и, возможно, ещё одной вершины) имеет ровно  $k$  детей. Покажите, как в такой куче можно реализовать `siftUp` и `siftDown`. Реализуйте через них классические операции кучи. За сколько они работают?
4. Заранее известно, что к куче поступит  $n^a$  запросов типа `extractMin` и  $n^b$  запросов типа `insert` (в произвольном порядке). Считаем, что  $0 < a < b$ , где  $a$  и  $b$  — заданные константы. Докажите, что при каждом  $n$  можно подобрать такое  $k$ , что на все запросы можно будет ответить с помощью  $k$ -ичной кучи за  $O(n^b)$ .
5. Пусть к изначально пустой биномиальной куче поступает  $n$  запросов типа `insert`. Докажите, что она обрабатывает их за суммарное время  $O(n)$ , хотя некоторые запросы требуют  $\Omega(\log n)$  операций.
6. На основе биномиальной кучи разработайте столь же мощную структуру (то есть позволяющую отвечать на те же запросы), которая обрабатывает любой запрос `insert` за  $O(1)$ . *Указание:* рассмотрите избыточный двоичный счётчик, в каждом разряде которого можно хранить 0, 1 или 2.
7. Пусть имеется алфавит из  $k$  символов, а также текст над этим алфавитом, который нужно закодировать с помощью нулей и единиц. Известно, что  $i$ -я буква входит в текст  $n_i$  раз. Код Хаффмана (код с доказуемо самой маленькой длиной получившейся длины закодированного текста) строится следующим образом. Пусть  $n_1 \geq n_2 \geq \dots \geq n_k$ . Сначала построим код Хаффмана для букв с частотами вхождений  $n_1, n_2, \dots, n_{k-2}, n_{k-1} + n_k$ . Пусть построен код, состоящий из слов  $w_1, \dots, w_{n-1}$ . Тогда искомым код Хаффмана задаётся так:  $w_1, w_2, \dots, w_{n-1}0, w_{n-1}1$ . Иными словами, две самые редкие буквы “склеиваются”, строится более простой код, а потом к общему слову  $w_{n-1}$  приписываются разные биты. Постройте такой код за  $O(k \log k)$ .