

## 5. Двоичное дерево множеств (puu)

1 секунда

50 очков

Двоичное дерево множеств – это структура данных, аналогичная двоичному дереву, но позволяющая в каждой вершине хранить не один, а несколько элементов. Так же как и в случае обычного двоичного дерева, дерево множеств должно удовлетворять условию упорядоченности: элементы в каждой вершине должны быть все строго больше всех элементов всех вершин левого поддерева, и строго меньше всех элементов всех вершин правого поддерева.

Так же как и обычное двоичное дерево, дерево множеств позволяет осуществлять поиск элементов. В каждой вершине можно решить, принадлежит ли искомое значение множеству элементов этой вершины, или же нужно продолжить поиск в левом или в правом поддереве. Если считать одно такое решение за одну элементарную операцию, то для нахождения вершины, содержащей заданное значение, понадобится столько же элементарных операций, сколько уровней дерева нужно пройти от корня до этой вершины (т.е. глубина этой вершины в дереве).

В данном задании мы рассмотрим ситуацию, где из-за определенных особенностей управления памятью максимальное количество элементов, которые могут храниться в одной вершине дерева разное, и зависит от глубины вершины. А именно, для вершины на глубине  $i$  максимальное количество хранимых в ней элементов  $m_i$  задается следующим образом:

$$m_i = \begin{cases} M & \text{если } i = 1 \text{ (т.е. это корень дерева),} \\ \max(1, m_{i-1} - D_{((i-2) \bmod K)+1}) & \text{если } i > 1, \end{cases}$$

где  $M$ ,  $K$  и  $D_j$  – фиксированные целые числа, а  $(i-1) \bmod K$  означает остаток от деления числа  $i-1$  на  $K$ . Например, если  $M = 4$ ,  $K = 2$ ,  $D_1 = 1$ ,  $D_2 = 2$ , то в корне дерева может храниться не более  $m_1 = M = 4$  элементов. В обеих вершинах, непосредственно связанных с корнем не более  $m_2 = m_1 - D_1 = 4 - 1 = 3$ . В детях этих вершин, в свою очередь, не более  $m_3 = m_2 - D_2 = 3 - 2 = 1$  элементов, а в вершинах во всех остальных слоях дерева может храниться максимум  $m_i = \max(1, \dots) = 1$  элемент.

Вдобавок, известно, что в дереве придется искать разные элементы с разной вероятностью, поэтому для заданного набора элементов сбалансированное дерево не обязательно будет оптимальным в плане среднего времени поиска. Например, если в подавляющем числе случаев в дереве запрашивается поиск минимального из хранящихся там элементов, стоит расположить его в самом корне, оставив в таком случае всё левое поддерево пустым.

Задача – для заданного набора элементов и вероятностей их запросов, найти среднее ожидаемое количество элементарных операций, необходимое для осуществления таких запросов в двоичном дереве множеств оптимальной формы.

**Входные данные.** На первой строке текстового файла `puusis.txt` даны три целых числа: количество элементов  $N$  ( $1 \leq N \leq 100$ ), а также  $M$  ( $1 \leq M \leq N$ ) и  $K$  ( $1 \leq K \leq N$ ). На следующих  $K$  строках по одному целому числу: на строке  $j+1$  дано значение  $D_j$  ( $0 \leq D_j \leq M$ ). На последней строке даны  $N$  действительных чисел  $P_i$  ( $0 \leq P_i \leq 1$ ;  $\sum_{i=1}^N P_i = 1$ ): вероятности запросов каждого элемента. Вероятности упорядочены по значению соответствующего элемента (первая вероятность  $P_1$  соответствует наименьшему элементу). Обратите внимание, что сами значения элементов для решения задачи не нужны, и можно предполагать что они все различны.

**Выходные данные.** На единственной строке текстового файла `puuval.txt` вывести одно действительное число – среднее ожидаемое число элементарных операций, необходимое для осуществления поиска элемента в построенном на заданном множестве элементов оп-

тимальным образом двоичном дереве множеств. Значение может отличаться от точного ответа не более чем на  $10^{-6}$ .

**Пример.**

puusis.txt	puuval.txt
4 2 1	1.5000000
2	
0.2 0.2 0.3 0.3	

**Пример.**

puusis.txt	puuval.txt
13 4 2	1.7200000
1	
0	
0.06 0.06 0.06 0.06 0.06 0.06 0.115 0.115 0.115 0.115 0.06 0.06 0.06	

Оптимальное дерево во втором примере следующее:

